Integrated Real-Time Data Stream Analysis and Sketch-Based Video Retrieval in Team Sports

Lukas Probst, Fabian Rauschenbach, Heiko Schuldt
Department of Mathematics and Computer Science
University of Basel, Switzerland
firstname.lastname@unibas.ch

Philipp Seidenschwarz, Martin Rumo
Centre for Technologies in Sports and Medicine
Bern University of Applied Sciences, Switzerland
firstname.lastname@bfh.ch

***Abstract*—Big data in sports comes with two closely related challenges: first, the online analysis of continuous data streams to identify characteristic events and second, advanced retrieval in video collections and/or event data that help game analysts to search for characteristic video scenes. For both challenges, dedicated big data stream processing and retrieval systems have been developed. However, there is no infrastructure yet that integrates retrieval and automatic online data stream analysis. In this paper, we close this gap by seamlessly combining STREAMTEAM, our real-time team sports analysis system, and SPORTSENSE, our team sports video retrieval system, to an integrated team sports analysis infrastructure that (i) automatically detects (collaborative) events and generates statistics in real-time based on a continuous stream of raw positions, (ii) visualizes the analysis results in real-time, (iii) stores the analysis results persistently for offline activities, and (iv) leverages the stored analysis results for intuitive sketch-based video retrieval.**

***Keywords*-Big Data Processing and Analysis, Complex Event Detection, Data Stream Analysis, Sketch Interfaces, Spatio-Temporal Video Retrieval, Team Sports Analysis**

I. INTRODUCTION

The last decades have shown a continuous professionalization and commercialization in sports. With the emergence of Big Data and IoT technologies, sport teams and organizations have started to gather data and analyze it to improve their performance in future competitions.

Today, many professional sport teams compile large video collections of past matches for analytical purposes. Game analysts use these collections to identify strengths and weaknesses of their own team and their competitors in order to adapt the tactics for future matches. During this process and for preparing tactical instructions, game analysts need to find characteristic video scenes. However, finding these scenes is still a tedious and labor-intensive task since it requires that video collections are manually scanned. To solve this issue, team sports (video) scene *retrieval systems* such as [1] and [2] enable game analysts to query for scenes in an intuitive way. In [3] we have presented SPORTSENSE, our team sports video retrieval system that supports four sketch-based spatio-temporal query types.

All these retrieval systems have in common that they are based on large video tag datasets including positions, events (passes, goals, etc.), and possibly also statistics (distances, etc.). *Automatic analysis systems* obviate the necessity to manually construct these datasets [4]–[6] or to buy manually generated datasets from commercial providers like Opta [7] or STATS [8]. Video-based automatic analysis systems such as [9] and [10] generate these datasets by analyzing the videos directly, i.e., extracting and processing image and audio features. Position-based automatic analysis systems detect events and generate statistics based on raw position data of the players and the ball as measured by sensors [11]–[20] or extracted from video signals [21]–[25]. One type of position-based systems analyzes complete position datasets in order to detect events and generate statistics for past matches [26]–[28]. Other systems consume the player and ball positions as data streams to detect events and generate statistics in real-time [29]–[36]. Some of these real-time analysis systems further visualize the analysis results in an online monitoring UI. In [37], we have presented STREAMTEAM, our real-time team sports analysis system that aims at analyzing the performance of individuals and the tactical behavior of the entire team.

Although specialized retrieval systems and automatic analysis systems for team sports exist already, to the best of our knowledge there is no integrated infrastructure yet that combines real-time analysis, online monitoring, and offline retrieval. In this paper, we fill this gap by presenting our integrated team sports analysis infrastructure consisting of STREAMTEAM and SPORTSENSE. This integrated infrastructure (i) automatically detects events and generates statistics in real-time out of the raw position data stream, (ii) visualizes the analysis results in an online monitoring UI, (iii) stores the analysis results persistently (as video tags) in a spatial database, and (iv) enables game analysts to retrieve characteristic video scenes intuitively in a sketch-based UI. To meet the functional requirements of such an integrated infrastructure, we have closely worked together with sport scientists. We show the applicability of our approach by reporting on the performance characteristics of the football version of our infrastructure that is able to deal with the volume, variety, and velocity of stream data that can be found in sports.

978-1-5386-5035-6/18/$31.00 ©2018 IEEE

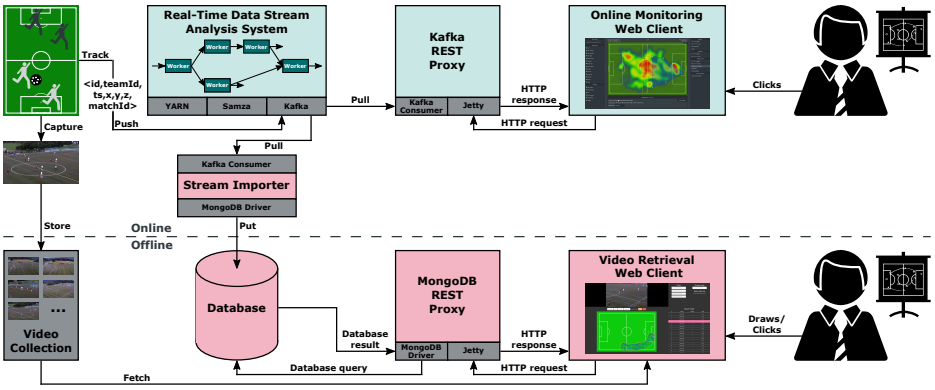


Figure 1: Overall Architecture. STREAMTEAM components colored in mint and SPORTSENSE components colored in red.

II. INFRASTRUCTURE

The sole input of our integrated team sports analysis infrastructure (depicted in Figure 1) are a data stream that contains raw positions of the players and the ball, and a single data stream element for every match that contains metadata such as the player/team names and the video path. The raw positions can either be collected from sensors or extracted from video signals.

We use Kafka [38], a widely used publish/subscribe system, for transferring data stream elements between components. We expect raw position data stream sources to push the data stream elements directly into Kafka. If sources do not support that, a proxy component (not depicted in Figure 1) which receives or pulls the elements from these sources and pushes them to Kafka has to be added.

Our *real-time data stream analysis system* consumes the raw position data stream and the metadata stream elements. Based on these data it performs analyses – mainly detections and aggregations – and emits a unified position stream as well as a multitude of event and statistic data streams.

The *online monitoring Web client* consumes the metadata stream as well as data streams that have been generated by the real-time data stream analysis system and visualizes them. Thus, it serves as a user interface for game analysts, coaches, or broadcasters who want to be informed about events and statistics in real-time. In order to enable those Web clients which are not able to directly access data stream elements from Kafka to still consume them, we have added the *Kafka REST proxy* to the infrastructure.

To make the match metadata, the positions, and the analysis results persistent for offline activities, they are stored in a MongoDB *database* instance. The process of storing the streamed data is handled by the *stream importer*. It consumes data stream elements from Kafka, transforms them to data items according to well-defined schemata, and stores these data items in the database.

The *video retrieval Web client* is a sketch-based user interface that enables game analysts to intuitively query for characteristic video scenes. It leverages the spatio-temporal data items stored in the database as video tags. The *MongoDB REST proxy* receives query type-specific API calls from the Web client and translates them to database queries it issues to the MongoDB instance.

III. STREAMTEAM

STREAMTEAM is a real-time team sports analysis system that automatically detects events and generates statistics based on a data stream containing the raw positions of the players and the ball. Moreover, it visualizes the analysis results in an online monitoring Web client. An earlier version of STREAMTEAM has been presented in [37]. However, in comparison to [37], the amount of analyses STREAMTEAM performs has increased remarkably. In our integrated infrastructure, STREAMTEAM contributes the real-time data stream analysis system, the Kafka REST proxy, and the online monitoring Web client.

A. Real-Time Data Stream Analysis System

As illustrated in Figure 1, the real-time data stream analysis system consists of multiple workers. In fact, the real-time data stream analysis system is not a single component but a multitude of loosely-coupled workers that are connected to an analysis workflow. The overall analysis task is performed stepwise by these workers.

This design choice forces a proper separation of the analysis steps and facilitates to easily adapt and extend the analysis workflow by means of modifying single and adding new workers. Moreover, it enables either performing the analysis on a single powerful server or distributing the analysis task to a cluster of computers.

The implementation of the real-time analysis workflow bases on the STREAMTEAM middleware [37]. In a nutshell, each worker is implemented in Java as a Samza [39] job using Samza's low-level API[1]. However, to ease the development of new workers and to reduce duplicated code, a Samza job is not implemented monolithically but constructed by means of combining generic and application-specific modules. The resulting Samza jobs are deployed on

[1]http://samza.apache.org/learn/documentation/0.13/api/overview.html

Category	Events/Statistics
Basic events	Ball possession changes, kicks, speed level changes
Set play events	Kickoffs, free kicks, goal kicks, corner kicks, throw-ins, penalties
Pass events	Successful passes, misplaced passes, interceptions, clearances, double passes, pass sequences
Shot events	Goals, shots off target
Non-atomic events	Duels, pressure situations, dribblings, offside lines
Statistics	Ball possession, pass, pass sequence, set play, shot, distance, speed level, dribbling, heatmap, pressing

Table I: STREAMTEAM-FOOTBALL's Events and Statistics

a YARN cluster [40] and communicate with Kafka. That is, a worker consumes its input data streams from and produces its output data streams to Kafka. In consequence, every data stream produced by a worker is made accessible to all other analysis workers and also to all other components in our infrastructure. This considerably simplifies building an integrated infrastructure as not all potential future use cases have to be considered in advance.

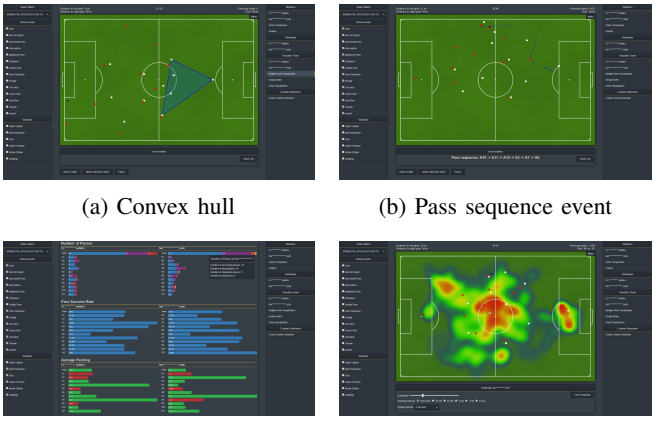
The STREAMTEAM middleware can be used to build real-time analysis workflows for arbitrary team sports such as football, ice hockey, American football, or even for non-sports scenarios like disaster management (e.g., firefighter coordination during forest fires). The analysis workflow of STREAMTEAM-FOOTBALL, our current football analysis system developed in close collaboration with sport scientists and football coaches, comprises 13 workers. The first worker standardizes the source-specific raw position data stream to a unified position stream and is thus similar to Herakles' data abstraction approach [35]. The other workers detect all events and generate the statistics listed in Table I. Since all streams (except the match metadata stream) and thus according to Samza's concept all workers are partitioned by the identifier of the match[2], STREAMTEAM-FOOTBALL is further able to analyze multiple football matches in parallel.

B. Online Monitoring Web Client

All analysis results produced in real-time by STREAMTEAM's analysis workflow are emitted as data stream elements of different streams – a Kafka topic for each type. For instance, in STREAMTEAM-FOOTBALL a successful pass event is textually represented by a data stream element in the *successfulPassEvent* stream (see Figure 3). While this is a good representation for further processing the analysis results in subsequent workers, end users cannot conceive these abstract information. Instead, they require an intuitive visualization of the analysis results.

For this purpose, STREAMTEAM comprises an online monitoring Web client which consumes the match metadata stream as well as the generated data streams and visualizes the analysis results in real-time. The Web client can differ for every sports discipline to optimally fit its characteristics. It is even possible to provide different Web clients for the same sports discipline that serve different purposes for different end user categories. For instance, game analysts might require a comprehensive Web client with access to all information while broadcasters might want to provide a lightweight Web client as a second screen application for their viewers. As all data streams are accessible, every Web client can consume the streams it requires for its visualizations.

[2]The identifier of the match is the key for the data stream elements pushed into Kafka.



(a) Convex hull (b) Pass sequence event

(c) Comparative pass statistics (d) Heatmap for player A1

Figure 2: Screenshots of STREAMTEAM-FOOTBALL's Online Monitoring Web Client (anonymized names)

Figure 2 shows screenshots of STREAMTEAM-FOOTBALL's online monitoring Web client implemented in Javascript. Amongst others, it enables the user (i) to highlight the current spatial arrangement of a set of players with a line, a bounding rectangle, or a convex hull (see Figure 2a), (ii) to get notified about detected events and to look at their visualization (see Figure 2b), (iii) to view bar charts of various statistics (see Figure 2c), and (iv) to observe the heatmaps for single players or whole teams for different time intervals and granularities (see Figure 2d).

C. Kafka REST Proxy

Web clients implemented in plain Javascript cannot access data stream elements directly from Kafka. To solve this issue, we have added the Kafka REST proxy to our infrastructure [37]. It enables Web clients to access the data stream elements via a REST API. The proxy supports retrieving the latest n data stream elements from topic t assigned to a certain key k (i.e., to a certain match). It does so by consuming data stream elements from Kafka for all topic-key-combinations, buffering them, and answering HTTP requests from Web clients. STREAMTEAM-FOOTBALL's Web client uses this proxy to consume data stream elements from Kafka while still achieving real-time performance.

IV. SPORTSENSE

SPORTSENSE is an intuitive video retrieval system that enables game analysts to issue queries for characteristic video scenes based on hand-drawn sketches. It contributes all offline components to our integrated team sports analysis infrastructure, i.e., the database, the video retrieval Web client, and the MongoDB REST proxy. Moreover, it provides the stream importer which serves as the connecting piece between SPORTSENSE and STREAMTEAM. In [3], we have presented SPORTSENSE with a primary focus on its generic database schema and its user interface. In contrast, this section focuses on the interactions inside SPORTSENSE and especially between SPORTSENSE and STREAMTEAM.

A. Database

The main idea of SPORTSENSE is to enable game analysts to intuitively query for characteristic video scenes that show situations of interest. Most situations in a match are best represented by motions (of the player or the ball) and/or by events. For instance, a counter attack is reflected by ball or player motion paths and a build-up of a team is reflected by the sequence of events (e.g., goal kicks, passes, etc.) that forms the build-up. Hence, SPORTSENSE leverages a database that stores tracking data and events as video tags.

Since tracking data as well as events contain positions that we need to query in an efficient way, we require a spatial database [41]. We have decided to use MongoDB [42] as it supports all spatial features we require for SPORTSENSE – 2D containment queries and 2D indexes – and shows suitable performance characteristics [43], [44]. SPORTSENSE's MongoDB instance contains three collections: One for storing all match metadata, one for storing all tracking data, and one for storing all events.

We have defined two generic schemata for storing the video tags in the database: A tracking data and event schema, and a match metadata schema. The bottom part of Figure 3 shows a sample of a successful pass event formatted as a video tag according to the tracking data and event schema. Both schemata have in common that they can be used to store information of arbitrary team sports as they are independent of a concrete sports discipline. The match metadata schema has a fully fixed structure as all matches of all team sports disciplines have the same metadata such as a match identifier, a date, a venue, a video path, and team/player name maps. In contrast, the other schema has a semi-fixed structure which enables a consistent access and indexes on common information. At the same time, it preserves the flexibility and thus extensibility towards new event types which is needed for data characterized by a large variety. All mandatory information are stored with a fixed structure (e.g., the corresponding video offset is always stored in the *videoTs* field). Type-specific information – the velocity for tracking data or event-specific information (e.g., the length of a successful pass) – are stored in the *additionalInfo* field which contains arbitrary key/value pairs. A more detailed description of both schemata and an additional schema for storing statistics (not queried yet in SPORTSENSE) is given in [3].

B. Stream Importer

It is possible to manually generate or buy the video tag dataset that is stored in SPORTSENSE's database. However, the former is a very tedious and time-consuming task and the latter is expensive and generates a dependency on an off-site company. With our integrated team sports analysis infrastructure, we follow a different approach by introducing the stream importer component as a connecting piece between STREAMTEAM and SPORTSENSE. In a nutshell, the stream importer consumes data stream elements generated by STREAMTEAM, transforms them into video tags according to our generic schemata, and stores them in SPORTSENSE's database.

The stream importer component is implemented in Java and runs in real-time. It consumes all elements from the match metadata stream, the unified position data stream as well as all non-intermediate atomic event data streams directly from Kafka. Data stream elements representing intermediate events are omitted since they are either only auxiliary (e.g., ball-in-area events) or too minor (e.g., ball possession changes) to represent interesting situations and are thus not relevant for retrieving characteristic video scenes. However, they could be consumed, transformed, and stored in the same way as data stream elements representing non-intermediate atomic events (e.g., successful pass events). Data stream elements representing non-atomic events (e.g., duels) are not yet considered as a single non-atomic event (e.g., a duel of player A1 and player B7) is represended by a vast number of data stream elements which are not useful for SPORTSENSE's query types. However, we plan to integrate these events in our future work.

The match metadata stream and every data stream generated by STREAMTEAM has a well-defined schema. However, this schema does not match the two schemata we use in SPORTSENSE's database. To solve this issue, every consumed data stream element is transformed to one data item.

A consumed match metadata stream element is transformed by means of a simple schema conversion. Moreover, the start timestamp *startTs* (i.e., the timestamp of the first raw position data stream element of a match) and the video offset at the start of the match (*videoOffset*) are stored for every match.

Every consumed unified position data stream element is transformed to a tracking data item and every consumed event data stream element is transformed to an event data item. Ball position data stream elements whose x or y coordinate values are not in the $[-180, 180)$ interval imposed by MongoDB's 2D index limitations[3] are omitted. Figure 3

[3]https://docs.mongodb.com/manual/tutorial/build-a-2d-index

```
topic: successfulPassEvent
offset: 445
schema: dseTs;teamId;matchId;playerKickId;
        playerKickX;playerKickY;playerKickZ;
        playerReceiveId;playerReceiveX;
        playerReceiveY;playerReceiveZ;length;
        velocity;angle;directionCategory;packing
data: 4125411;A;638809;A6;-9.27;-17.614;0.0;A2;
      -17.346;-14.95;0.0;8.504;10.172;341.744;
      forward;4
```

↓ Transform successful pass event data stream element to a successful pass event data item

```
{  "type": "successfulPassEvent",
   "matchId": "638809",
   "ts": 893024,      ← dseTs-startTs
   "videoTs": 4124,   ← videoOffset-(ts/1000)
   "xyCoords": [[-9.27,-17.614],[-17.346,-14.95]],
   "zCoords": [0.0,0.0],
   "playerIds": ["A6","A2"],
   "teamIds": ["A"],
   "additionalInfo": {
      "lenght": 8.504,
      "velocity": 10.172,
      "angle": 341.744,
      "directionCategory": "forward",
      "packing": 4.0 } }
```

Figure 3: Sample Transformation

shows a sample transformation process for a successful pass event generated by STREAMTEAM-FOOTBALL. A large part of the information can be simply extracted from the content of the data stream element and its topic. Only *ts* (i.e., milliseconds since the start of the match) and *videoTs* (i.e., video offset in seconds) have to be calculated.

Once all data stream elements retrieved in one Kafka poll call have been transformed to data items, the resulting data items are inserted in batch mode into the database.

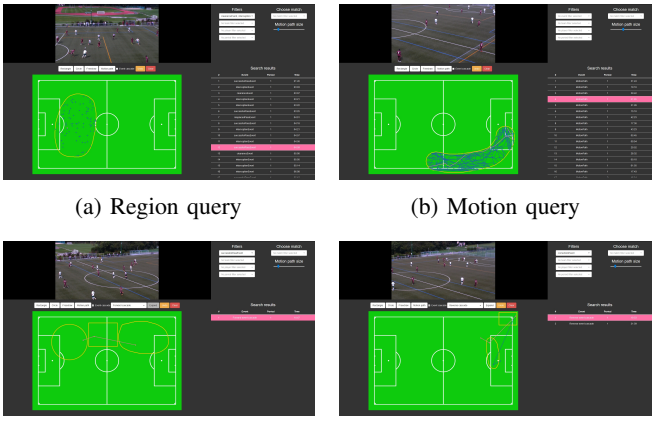
Although the database is thought to serve as a video tag dataset for SPORTSENSE, the generic schemata and the stream importer are designed in a way that the resulting database can be used as a persistent dump of the real-time data for arbitrary offline activities. For this reason, the stream importer stores also all generated statistics data stream elements (not yet used by SPORTSENSE) according to the statistics schema presented in [3]. We omit details regarding the statistics due to space restrictions.

C. Video Retrieval Web Client

In order to enable game analysts to retrieve characteristic video scenes in an intuitive way, SPORTSENSE contains a video retrieval Web client. This client may differ for every sports discipline. However, all clients share SPORTSENSE's UI concept which is to specify spatio-temporal queries by drawing sketches on the field.

Our video retrieval Web client tailored to football, SPORTSENSE-FOOTBALL, has been designed based on the findings of a user study which has compared two older prototypes [3]. It is implemented in Typescript and supports four sketch-based query types.

In order to retrieve a video scene in which a certain event has happened in a specific region, SPORTSENSE-FOOTBALL supports region queries [3], [45]. To issue a region query, the user has to draw an area – a rectangle, a circle, or a free sketch – on the field. Moreover, the user can filter for event types, matches, or involved players. Figure 4a shows the result of a region query for passes in front of the goal.



(a) Region query (b) Motion query

(c) Forward event cascade query (d) Reverse event cascade query

Figure 4: Screenshots of SPORTSENSE-FOOTBALL's Video Retrieval Web Client

Motion queries enable the user to retrieve video scenes in which the ball has moved along a certain path [3], [45]. To issue a motion query, the user has to specify the path by drawing a line which is transparently wrapped by a polygon. The result of a motion query for a path from the center along the sideline towards the right goal is given in Figure 4b.

If users are interested in retrieving videos in which a certain sequence of events is shown, they can use a forward event cascade query [3], [45]. In doing so, they have to subsequently draw the regions and set the filters for all events in the sequence. Figure 4c shows the result of a forward event cascade query consisting of a pass in the top right, a pass in the top center, and a pass in the top left.

Moreover, SPORTSENSE-FOOTBALL supports reverse event cascade queries that enable the user to discover which event sequences have lead to a certain type of event in a specific region [3], [46]. This can be used for instance to retrieve video scenes with characteristic offense patterns that typically lead to a corner kick. For specifying such a query, the user has to draw an area and specify a filter for the start event (e.g., the corner kick) and than expand step by step into the past. Optionally, the user can define a set of event types that shall be regarded during discovering past events (e.g., only the sequence of passes leading to a corner kick). Moreover, prior to each expansion the user is able to reduce the amount of sequences to expand by drawing another area. The result of a corner kick emergence query is shown in Figure 4d.

D. MongoDB REST Proxy

In this section, we present how the retrieval task specified in the video retrieval Web client is performed by the

MongoDB REST proxy. In a nutshell, the MongoDB REST proxy receives query type-specific REST API requests from the video retrieval Web client, translates them into database queries and finally returns the results to the Web client.

In case of a region query, the video retrieval Web client sends a single HTTP request to the MongoDB REST proxy when the user has finished drawing the area. Once the proxy receives a region query REST API call, it transforms this call into a single database query containing all geo (e.g., the points of the polygon defining the free sketch area) and non-geo (event type, player, etc.) filters and issues this query to the MongoDB instance. When the proxy has received the database query result from MongoDB, it constructs a JSON object and returns this as the HTTP response to the Web client which consequently visualizes the result.

Also in case of a motion query, the Web clients sends a single HTTP request. On receipt of the motion query REST API call, the proxy issues a database query for all ball tracking data in the polygon that defines the path sorted by the *matchId* and the *ts*. The returned list of tracking data items is then partitioned into paths by iterating over it and comparing the *ts* and *matchId* of subsequent items. Sufficiently long paths are added to the JSON object which is returned as the HTTP response to the Web client.

A forward event cascade query consists of several rounds of user interaction. After the user has drawn the first area, the Web client issues a REST API call that is handled as a normal region query call. When the user has drawn the second (third, etc.) area, the Web client issues another call which additionally contains the timestamps of all events returned as results in the preceding call. This is necessary to query only for those events that succeed the previous event in the resulting sequence in a limited time interval. Apart from the additional *ts* filters, the database query is very similar to the normal region query. After every interaction, the HTTP response contains the events found for the latest area. The Web client uses this information to update the visualization.

Also reverse event cascade queries encompass multiple rounds. The REST API call for the start event that is issued in the first round is handled in the same way as the first call in a forward event cascade query. Subsequently, after every expand action, the Web client sends a REST API call containing the optional event type filter set and the timestamps of the oldest events of those event sequences the user wants to further expand into the past. On receipt of these calls, the proxy performs a database query for all events that precede the given timestamps (in a given time interval) and fulfill the optional event type filter. Note that the database queries for the expand actions do not contain any geo filters. Instead, the area filter that reduces the set of event sequences is evaluated in the Web client before it sends the REST API call for the next expand action. Hence, only the timestamps of those oldest events which have happened in the area are contained in the expand request.

Query Type	Total	Call 1	Call 2	Call 3
Region	10.20 ms	-	-	-
Motion	3174.50 ms	-	-	-
Forward Cascade	17.20 ms	5.90 ms	8.50 ms	2.80 ms
Reverse Cascade	33.00 ms	3.05 ms	17.55 ms	12.40 ms

Table II: Average Retrieval Time for a Single Match

V. Evaluation

To demonstrate the applicability of our integrated team sports analysis infrastructure, we have combined the current football prototypes of our real-time analysis system and our video retrieval system, i.e., StreamTeam-Football and SportSense-Football, and used the resulting football analysis infrastructure for analyzing a test football match.

The raw position data stream for this test match has been obtained using the hybrid multimodal Inmotio system [47]. That is, the raw player positions are generated using LPM [12] and the raw ball positions are extracted from video signals [21]. As the raw ball positions contain errors due to misdetections, they have been corrected manually. The position measurements have been pushed with 45 Hz per player/ball as raw position data stream elements into Kafka.

All components except for the clients of StreamTeam-Football and SportSense-Football have been running on five cluster machines with 32 GB RAM and an Intel i7-4770. One machine is hosting the YARN resource manager, the primary and secondary HDFS name node, a Kafka instance, a Zookeeper instance, the Kafka REST proxy, the stream importer, the MongoDB REST proxy, and the MongoDB instance. The remaining four machines have run a YARN resource manager (executing Samza jobs), an HDFS data node, and a Kafka instance. Moreover, two of these machines have run Zookeeper instances. The Web clients of StreamTeam-Football and SportSense-Football have been running on a state-of-the-art commodity machine.

Using this deployment, StreamTeam-Football is able to perform all analyses and visualize all results for end users in real-time. Moreover, SportSense-Football enables game analysts to retrieve video scenes of the analyzed match at interactive speed. As the stream importer consumes and stores the analysis results in real-time, it is even possible to retrieve events with SportSense-Football that have been detected by StreamTeam-Football only seconds ago. Hence, our infrastructure supports retrieving scenes of a still ongoing match.

To evaluate SportSense's retrieval performance in more detail, we captured the REST API calls issued during the retrieval tasks depicted in Figure 4 and measured the time between issuing a call and receiving the results. The result visualization is not included in this measurement. We have measured the duration of each call 20 times using cURL (https://curl.haxx.se) and calculated the mean value. As

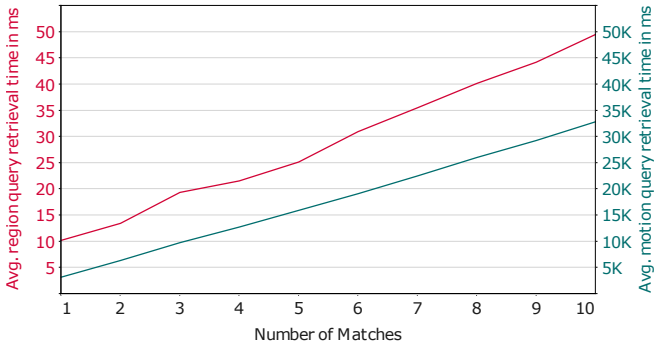


Figure 5: Scalability w.r.t. the Number of Matches

Table II shows, the REST API calls for the region and cascade queries are very fast. The call for the motion query takes significantly longer since there are much more tracking data items (2.836.885) than event data items (751). Nevertheless, the call is still fast enough to enable a pleasant user experience. A detailed user study confirming SPORTSENSE's usability and effectiveness is presented in [3].

In order to evaluate SPORTSENSE's scalability w.r.t. the data volume we have considered up to ten matches and measured the time of the region and motion query REST API call for each number of matches. As Figure 5 shows, the average retrieval time scales linearly with the number of matches for both query types. This result confirms that SPORTSENSE can be used to retrieve scenes of multiple matches each comprising millions of tracking data items and hundreds of event data items. Moreover, it indicates that our integrated infrastructure can be used to automatically detect events in and retrieve video scenes of matches of a whole or even multiple seasons.

VI. Related Work

In this section, we compare our infrastructure with other team sports analysis infrastructures. Related work for STREAMTEAM and SPORTSENSE as stand-alone systems is discussed in [37] and [3], resp.

In [48] Blobel and Lames formulate the need for a central infrastructure that manages all data of a football club. However, their first prototype only manages and visualizes health-related data. It does neither perform analyses in terms of generating statistics or detecting events, nor support the retrieval of video scenes.

The set of products for graphical video enhancement, position tracking, and manual tagging offered by Chyron-Hego [49] can also be regarded as an integrated team sports analysis infrastructure. Similarly, myDartfish Live S [5] is an infrastructure with different software for manually tagging and graphically enhancing sports videos. Moreover, Stein et al. [50] propose an infrastructure that tracks players in football videos, performs analyses based on the player positions (and manual event annotations), and enhances the videos with visualizations of the analysis results. However, these infrastructures neither contain a component that automatically detects complex events nor a component for sketch-based video retrieval. Instead, they focus on telestration.

In [51] Wilhelm et al. have presented an infrastructure that combines the collection of sensor measurements (e.g., the players' heart rate and speed), video-tracking, and position-based analyses. Hashimoto and Ozawa [52] propose an offside detection infrastructure that tracks player positions and performs position-based analyses. In contrast to our infrastructure that performs its analyses in real-time, [51] explicitly states that it operates partly offline and [52] does not provide any information. Moreover, they do not support the retrieval of video scenes based on analysis results.

STATS offers multiple products for football analysis [53] including solutions for position tracking, real-time visualization, and video scene retrieval. These products form an integrated team sports analysis infrastructure. However, this infrastructure merely calculates some basic statistics (e.g., distances and ball possession) but does not automatically detect events as our infrastructure does. Instead, STATS manually generates a live data feed which they use as a dataset for their video retrieval product.

SAP Sports One [54] is a cloud-based infrastructure for managing all data of a team including performance data, health-related data, training plans, and scouting activities. Moreover, it enables users to retrieve video scenes showing specific events in an intuitive, even though not sketch-based way. However, SAP Sports One does not generate the event dataset required for the retrieval automatically but depends on manual annotations.

VII. Conclusion

Data analysis in team sports has become increasingly prevalent. Although there are already approaches for retrieval and automatic analysis in team sports, to the best of our knowledge, there is no infrastructure yet that jointly supports the analysis of raw position data in real-time, the online monitoring of a match based on the analysis results, and the offline video scene retrieval using analysis results as video tags. In this paper, we have presented our integrated team sports analysis infrastructure that fills this gap by combining our real-time analysis system STREAMTEAM and our video retrieval system SPORTSENSE and that is tailored to the volume, variety, and velocity of the data that can be found in sports. Moreover, we have shown the applicability of our infrastructure by presenting and evaluating our current football prototype.

Acknowledgment

This work has been partly supported by the Hasler Foundation in the context of the project STREAMTEAM, contract no. 16074. We want to thank Ihab Al Kabary, Rufus Lobo, and Frederik Brix for their participation in developing previous versions of STREAMTEAM and SPORTSENSE. Some

icons used in Figure 1 are made by Freepic (Freepic icon packs at Flaticon: https://www.flaticon.com/authors/freepik).

REFERENCES

[1] L. Sha *et al.*, "Chalkboarding: A New Spatiotemporal Query Paradigm for Sports Play Retrieval," in *Proc. of IUI*, 2016.

[2] STATS, "STATS Edge," https://www.stats.com/edge, 2018.

[3] L. Probst *et al.*, "SportSense: User Interface for Sketch-Based Spatio-Temporal Team Sports Video Scene Retrieval," in *Proc. of UISTDA*, 2018.

[4] ChyronHego, "Coach Capture," https://chyronhego.com/products/broadcast-graphics/coach-capture, 2018.

[5] Dartfish, "myDartfish Live S," http://www.dartfish.com/live_S, 2018.

[6] Hudl, "Sportscode," https://www.hudl.com/elite/sportscode, 2018.

[7] Opta, "Opta Data feeds," https://www.optasports.com/services/data-feeds, 2018.

[8] STATS, "STATS Tier 6+ Definitions," http://developer.stats.com/files/Tier_6_Definitions.pdf, 2018.

[9] M. Fleischman and D. Roy, "Unsupervised Content-Based Indexing of Sports Video," in *Proc. of MIR*, 2007.

[10] S. Chen *et al.*, "Play Type Recognition in Real-World Football Video," in *Proc. of WACV*, 2014.

[11] T. von der Grün *et al.*, "A Real-Time Tracking System for Football Match and Training Analysis," in *Microelectronic Systems*, 2011.

[12] A. Stelzer *et al.*, "Concept and Application of LPM – A Novel 3-D Local Position Measurement System," *IEEE MTT*, vol. 52, no. 12, 2004.

[13] Catapult, "ClearSky T6," https://www.catapultsports.com/products/clearsky-t6, 2018.

[14] ——, "OptimEye S5," https://www.catapultsports.com/products/optimeye-s5, 2018.

[15] ——, "OptimEye X4," https://www.catapultsports.com/products/optimeye-x4, 2018.

[16] ChyronHego, "ZXY Arena," https://chyronhego.com/products/sports-tracking/zxy-arena-wearable-tracking, 2018.

[17] ——, "ZXY Go," https://chyronhego.com/products/sports-tracking/zxy-go-wearable-tracking, 2018.

[18] FieldWiz, "FieldWiz," https://www.fieldwiz.com, 2018.

[19] STATS, "STATS GPS," https://www.stats.com/gps, 2018.

[20] NBN23, "NBN23 Performance," https://www.nbn23.com/en/performance, 2018.

[21] inmotiotec, "Ball tracking technology," https://www.inmotiotec.com/en/technology/balltracking, 2018.

[22] ChyronHego, "TRACAB," https://chyronhego.com/products/sports-tracking/tracab-optical-tracking, 2018.

[23] R. Gade and T. B. Moeslund, "Thermal Tracking of Sports Players," *Sensors*, vol. 14, no. 8, 2014.

[24] ——, "Constrained multi-target tracking for team sports activities," *IPSJ CVA*, vol. 10, no. 2, 2018.

[25] STATS, "STATS SportVU football player tracking," https://www.stats.com/sportvu-football, 2018.

[26] A. A. Sangüesa *et al.*, "Identifying Basketball Plays from Sensor Data; towards a Low-Cost Automatic Extraction of Advanced Statistics," in *Proc. of ICDMW*, 2017.

[27] K. Richly *et al.*, "Recognizing Compound Events in Spatio-Temporal Football Data," in *Proc. of IoTBD*, 2016.

[28] K.-C. Wang and R. Zemel, "Classifying NBA Offensive Plays Using Neural Networks," in *Proc. of MIT Sloan Sports Analytics Conf.*, 2016.

[29] H.-A. Jacobsen *et al.*, "Grand Challenge: The Bluebay Soccer Monitoring Engine," in *Proc. of DEBS*, 2013.

[30] Y. Wu *et al.*, "Grand Challenge: SPRINT Stream Processing Engine as a Solution," in *Proc. of DEBS*, 2013.

[31] M. Jergler *et al.*, "Grand Challenge: Real-time Soccer Analytics Leveraging Low-Latency Complex Event Processing," in *Proc. of DEBS*, 2013.

[32] K. G. S. Madsen *et al.*, "Grand Challenge: MapReduce-Style Processing of Fast Sensor Data," in *Proc. of DEBS*, 2013.

[33] A. Gal *et al.*, "Grand Challenge: The TechniBall System," in *Proc. of DEBS*, 2013.

[34] S. Badiozamany *et al.*, "Grand Challenge: Implementation by Frequently Emitting Parallel Windows and User-defined Aggregate Functions," in *Proc. of DEBS*, 2013.

[35] T. Michelsen *et al.*, "Herakles: Real-time Sport Analysis using a Distributed Data Stream Management System," in *Proc. of DEBS*, 2015.

[36] L. Probst *et al.*, "PAN – Distributed Real-Time Complex Event Detection in Multiple Data Streams," in *Proc. of DAIS*, 2016.

[37] ——, "Real-Time Football Analysis with StreamTeam," in *Proc. of DEBS*, 2017.

[38] J. Kreps *et al.*, "Kafka: A distributed messaging system for log processing," in *Proc. of NetDB*, 2011.

[39] S. A. Noghabi *et al.*, "Samza: Stateful Scalable Stream Processing at LinkedIn," *Proc. of VLDB Endowment*, 2017.

[40] V. K. Vavilapalli *et al.*, "Apache Hadoop YARN: Yet Another Resource Negotiator," in *Proc. of SOCC*, 2013.

[41] R. H. Güting, "An introduction to spatial database systems," *VLDB Journal*, 1994.

[42] MongoDB, "MongoDB," https://www.mongodb.com, 2018.

[43] S. Schmid *et al.*, "Performance investigation of selected SQL and NoSQL databases," in *Proc. of AGILE*, 2015.

[44] S. Agarwal and K. Rajan, "Performance Analysis of MongoDB vs. PostGIS/PostGreSQL Databases for Line Intersection and Point Containment Spatial Queries," in *Proc. of FOSS4G*, 2015.

[45] I. Al Kabary and H. Schuldt, "Towards Sketch-based Motion Queries in Sports Videos," in *Proc. of ISM*, 2013.

[46] ——, "Enhancing Sketch-based Sport Video Retrieval by Suggesting Relevant Motion Paths," in *Proc. of SIGR*, 2014.

[47] inmotiotec, "Inmotio System (with LPM tracking technology)," https://www.inmotiotec.com/en/technology/inmotio-system, 2018.

[48] T. Blobel and M. Lames, "Information Systems for Top-Level Football with Focus on Performance Analysis and Healthy Reference Patterns," in *Proc. of IACSS*, 2017.

[49] ChyronHego, "ChyronHego Products," https://chyronhego.com/products, 2018.

[50] M. Stein *et al.*, "Bring it to the Pitch: Combining Video and Movement Data to Enhance Team Sport Analysis," *IEEE VGG*, vol. 24, no. 1, 2018.

[51] P. Wilhelm *et al.*, "An Integrated Monitoring and Analysis System for Performance Data of Indoor Sport Activities," in *Proc. of MCS*, 2010.

[52] S. Hashimoto and S. Ozawa, "A System for Automatic Judgment of Offsides in Soccer Games," in *Proc. of ICME*, 2006.

[53] STATS, "Football performance analysis solutions," https://www.stats.com/football, June 2018.

[54] SAP, "SAP Sports One," https://www.sap.com/products/sports-one.html, 2018.