The DEBS 2013 Grand Challenge

Christopher Mutschler
University of
Erlangen-Nuremberg
and
Fraunhofer Institute for
Integrated Circuits IIS
91058 Erlangen, Germany
Christopher.Mutschler@fau.de

Holger Ziekow
AGT International
Hilpertstr. 35
64295 Darmstadt, Germany
HZiekow@agtinternational.com

Zbigniew Jerzak
SAP AG
Chemnitzer Straße 48
01187 Dresden, Germany
Zbigniew.Jerzak@sap.com

ABSTRACT

The ACM DEBS 2013 Grand Challenge is the third in a series of challenges which seek to provide a common ground and evaluation criteria for a competition aimed at both research and industrial event-based systems. The goal of the Grand Challenge competition is to implement a solution to a real-world problem provided by the Grand Challenge organizers. The 2013 edition of the Grand Challenge focuses on real-time, event-based sports analytics. The 2013 Grand Challenge data set was collected during a football match carried out at a Nuremberg Stadium in Germany and is complemented with a set of continuous analytical queries which provide detailed insight into the match statistics for both team managers and spectators.

Categories and Subject Descriptors

C.2.4 [**Distributed Systems**]: Distributed applications

General Terms

Performance, Experimentation

Keywords

event processing, streaming, cep

1. INTRODUCTION

Event-based systems are increasing their footprint in different sectors of our economy. Typical use cases include algorithmic trading [5], information integration [2], and monitoring of key performance indicators in manufacturing domain [4]. With this edition of the DEBS 2013 Grand Challenge we want to explore a slightly different industry branch – sports and entertainment.

The goal of this year's Grand Challenge is to demonstrate the applicability of event based systems for providing real-time, continuous analytics for both managers of sport teams as well as spectators of sports events. The explicit focus of the 2013 Grand Challenge is on analytics for football games. To that end we provide a data set collected during a real football game and couple it with analytical queries. The goal of the analytical queries is twofold. On one hand side they provide the competitive edge to team managers allowing them to take better more insightful decisions in real time. Examples of such decisions include, e.g., substitution of underperforming players or monitoring of the weaknesses of the opponent team. On the other hand side the ability to operate on real time data allows for new and enriched experience for spectators both in stadiums as well as in front of TV screens. Typical examples of such improved experience include live, interactive in game statistics and analysis.

We hope that both data and queries presented in the DEBS 2013 Grand Challenge will drive stronger adoption of real-time event-based systems. We also hope that both data and queries will become helpful tool for evaluation of event based systems which goes beyond the scope of the DEBS 2013 Grand Challenge.

The remainder of this paper is structured as follows: in Section 2 we present the technical details of the real-time location system RedFIR which was used for collecting the raw data. In Section 3 we provide a detailed description of the recorded raw data. In Section 4 we provide a description of continuous queries to be executed on top of the recorded data. We conclude with Section 5.

2. REDFIR SYSTEM

The RedFIR tracking system is a Real-Time Locating System (RTLS) based on time-of-flight measurements, where small transmitter Integrated Circuits emit burst signals [3]. A centralized unit processes this microwave signals and extracts time of arrival (ToA) values. ToA values are the basis for time difference of arrival (TDoA) values, from which x, y, and z coordinates are derived. In the following, we describe the technical aspects of the RedFIR system installed in the main soccer stadium in Nuremberg, Germany, where the data for this challenge has been recorded.

The RedFIR system operates in the globally license-free ISM (industrial, scientific, and medical) band of 2.4 GHz and allows a usage of around 80 MHz. Miniaturized transmitters ($61x38x9$ mm) use this available bandwidth to generate short broadband signal bursts together with identification sequences. The locating system is able to receive an overall of $50,000$ of those signal bursts per second.

Figure 1 illustrates the signal processing chain of the Red-

DEBS'13, June 29–July 3, 2013, Arlington, Texas, USA.

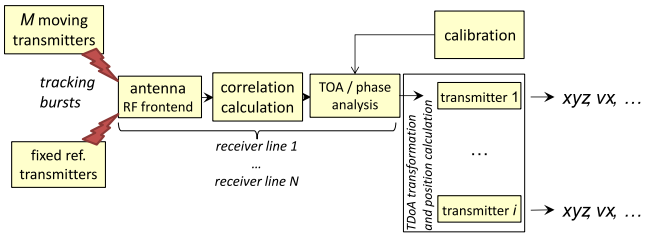

Figure 1: The signal processing chain of the RedFIR system

FIR system. The system distinguishes fixed reference transmitters for configuration purposes, i.e., six transmitters located at the corners and the lateral ends of the midline, from M moving transmitters, i.e, the balls or the mobile transmitters carried by the players. All transmitters emit tracking burst signals, which are received by N receiving antennas that are located on top of the flood light poles and the roofs of the stands. The installation in the soccer stadium in Nuremberg provides 12 antennas that receive signals from up to 144 different transmitters. Balls emit around 2000 tracking bursts per second whereas the remaining transmitters emit around 200 tracking bursts per second.

For each of the 12 receiver lines dedicated FPGAs correlate the individual burst sequences, and a CPU analyzes the resulting ToA and signal phase measurements. The receivers are synchronized so that the RedFIR system can determine time difference of arrival (TDoA) values out of the ToA values for each particular tracking burst between pairs of receivers. This allows the RedFIR system to avoid taking the time of transmission into account and thus alleviates the need to synchronize the transmitters. The RedFIR system can calculate the positions using only the timing information from the receivers. With the given number of receiving units N=12, the system derives N-1=11 TDoA-value streams.

Burst signals can be reflected by metallic surfaces as well as the playing field. This means that the receiving antennas might receive a given signal on multiple paths, which might negatively impact the quality of measurements. To avoid using the wrong propagation path an unscented Kalman filter (UKF) monitors several propagation paths at the same time and chooses a reliable line-of-sight (LOS) path for the tracking signal. Hence, at any time the most likely propagation path is used for positioning and ToA calculation [6].

In a final step the TDoA streams and the carrier phase signals serve as inputs to an extended Kalman filter (EKF), one per transmitter, to calculate the x, y, and z coordinates as well as other statistical information – see Section 3. The EKF uses hyperbolic positioning in order to calculate intersection points between measurements. The position information is further improved by using measurements of the carrier phase signal together with the ToA measurements - this allows to increase the accuracy for measurements of relative movements.

The miniature transmitters themselves are splash-proof (in case of the player transmitters) or integrated into the football. Hence, they are charged by an inductive principle. Figure 2 shows the charging cradle for a ball (charging units for the small transmitters are only a few centimeters long) and the suspension of the transmitter within the ball. Both ball and player transmitters can be fully charged within a few hours. The batteries run for over 3 hours, i.e., more than enough for a game with interruptions, stoppage time, and extra time. Moreover, the transmitter are application-specific integrated circuits (ASIC) and can be used as arbitrary waveform generators (AWG) operating at 2.4 GHz, i.e., they are re-programmable.

Figure 2: A glass model of the ball's transmitter and inductive charger

3. DATA

The DEBS 2013 Grand Challenge data set was recorded during a training game between two teams of 8 players. The game, during which the data has been collected, was played on a half-size field. The game duration was two halves of thirty minutes each. Figure 3 shows a schematic view of the playing field. Each corner of the field is marked with its coordinates used throughout the DENS 2013 Grand Challenge. The goal areas of both teams are marked with dashed lines. Since the playing field is not a perfect rectangle it is possible to approximate the field as a rectangle using following coordinates: $(0, 33965)$, $(0, -33960)$, $(52483, 33965)$, and $(52483, -33960)$.

Each player and a referee had two sensors embedded in their shin guards. Goalkeepers were additionally equipped

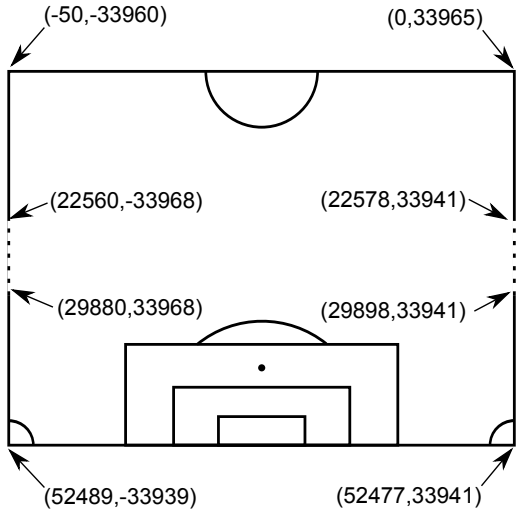

Figure 3: Playing field dimensions and goal areas (dashed lines)

with two sensors located in their gloves. Each ball used during the game had one sensor located approximately in the middle of the ball. Listing 1 gives an overview of the sensor to player assignment. Each sensor is identified by a unique number – its id. The sensor embedded in the left shin guard of a referee has an id of 105.

```
Referee
   Left Leg: 105, Right Leg: 106

Balls
   1st Half: 4, 8, 10
   2nd Half: 4, 8, 10, 12

Team A
   Goalkeeper A ( Left Leg: 13, Right Leg: 14,
                  Left Arm: 97, Right Arm: 98)
   Player A1 (Left Leg: 47, Right Leg:16)
   Player A2 (Left Leg: 49, Right Leg: 88)
   Player A3 (Left Leg: 19, Right Leg: 52)
   Player A4 (Left Leg: 53, Right Leg: 54)
   Player A5 (Left Leg: 23, Right Leg: 24)
   Player A6 (Left Leg: 57, Right Leg: 58)
   Player A7 (Left Leg: 59, Right Leg: 28)

Team B
   Goalkeeper B ( Left Leg: 61, Right Leg: 62,
                  Left Arm: 99, Right Arm: 100)
   Player B1 (Left Leg: 63, Right Leg: 64)
   Player B2 (Left Leg: 65, Right Leg: 66)
   Player B3 (Left Leg: 67, Right Leg: 68)
   Player B4 (Left Leg: 69, Right Leg: 38)
   Player B5 (Left Leg: 71, Right Leg: 40)
   Player B6 (Left Leg: 73, Right Leg: 74)
   Player B7 (Left Leg: 75, Right Leg: 44)
```

Listing 1: Assignment of sensors (ids) to players and balls

Sensors in the shin guards and gloves produce data with 200Hz frequency. The sensor in the ball produces data with 2000Hz frequency. The total data rate during the game reaches roughly 15,000 position events per second. Listing 2 shows the schema of the raw data set provided as input for the DEBS 2013 Grand Challenge.

```
sid,          //sensor id
ts,           //time stamp
x, y, z,      //sensor coordinates
|v|,         //velocity
|a|,         //acceleration
vx, vy, vz, //direction vector
ax, ay, az   //acceleration vector
```

Listing 2: The input data schema

sid is a unique identifier of the sensor which produced a signal used for the calculation of the position event by the RedFIR signal processing chain – see also Listing 1. *ts* is a time stamp of the given measurement in picoseconds. Time stamp with value 10,753,295,594,424,116 designates the start of the match. Time stamp 14,879,639,146,403,495 marks the end of the match. The first half of the game ends at 12,557,295,594,424,116. The second half begins at 13,086,639,146,403,495. x, y, and z describe the position of the sensor in mm – coordinates (0,0,0) are located at the center of a full size football field. $|v|$ describes is the velocity of the sensor in m/s. vx, vy, and vz represent directed velocity vectors with a normalized size of 10,000. The speed of the sensor in the x-direction v_x in SI-units (m/s) can be calculated as:

$$v_x = |v| \cdot vx \cdot 10^{-4} \cdot 10^{-6} \quad (1)$$

$|a|$ represents the acceleration of the sensor in m/s^2. ax, ay, and az represent directed acceleration vectors with a normalized size of 10,000. The acceleration of the sensor in the x-direction in m/s^2 is calculated analogously to that of the velocity:

$$a_x = |a| \cdot ax \cdot 10^{-4} \cdot 10^{-6} \quad (2)$$

The value of acceleration $|a|$ is zero when the ball is not moving. Due to the geometry of the football field, and hence the antenna placements over the available areas, the absolute and relative precision in the x and y horizontal plane is highly accurate, i.e., positions only deviate a few centimeters. Relative movements in the vertical plane (z) are also resolved with high accuracy. However, due to the limited capabilities of antenna placement in the vertical axis the resolution of the vertical plane cannot compete with that of the horizontal plane. Specifically, it is possible that values for the z coordinate become negative.

Raw sensor data for the whole game can be downloaded from the following link: http://lafayette.tosm.ttu.edu/debs2013/grandchallenge/full-game.gz. The total file size is 2.6 GB and it contains a total of 49,576,080 position events. The provided data file is sorted according to the time stamp of the events. The meta data file containing all game interruptions can be downloaded from the following link: http://lafayette.tosm.ttu.edu/debs2013/grandchallenge/referee-events.tar.gz.

For reference purposes we also provide a video recording of the whole game. The video has been recorded using a static camera showing a vertical view of the game. We provide two separate files, one for each half of the game. Both files are 1.7GB in size and are encoded using a standard H.264/MPEG-4 AVC standard. The file containing the footage of the first half of the game can be downloaded here: http://lafayette.tosm.ttu.edu/debs2013/grandchallenge/RedFIR_2012_1.mov, the file containing the footage of the

second half is available here: http://lafayette.tosm.ttu.edu/debs2013/grandchallenge/RedFIR_2012_2.mov.

Please note that the video is not fully aligned with the game as recorded by the RedFIR system. The sensor recordings start approximately 5 seconds earlier.

4. QUERIES

In this section we provide a description of four analytical queries which should be implemented as a part of the DEBS 2013 Grand Challenge. The goal of the queries is to provide continuous analysis on top of streaming data [1]. Therefore, in the remainder of the paper we assume that all queries are standing queries consuming the raw input data as defined in Section 3.

Each query, unless explicitly stated otherwise, should provide its results as a continuous stream of events. Result events, unless specified differently in the query, should be produced immediately upon the update from the input event. Results can be provided by simply writing to stdout or to a set of files, one for each result stream. It is also possible to use single output file, prefixing each result stream with a unique id.

4.1 Running Analysis

The goal of the running analysis query is to quantify and track how well each of the players moves on the playing field. The major indicators are the speed and distance covered by each player as a function of time. Each run undertaken by a player can be assigned to one of predefined intensity categories, according to its speed. We define the following intensities: standing (till $1km/h$), trot (till $11km/h$), low speed run (till $14km/h$), medium speed run (till $17km/h$), high speed run (till $24km/h$), and sprint (faster than $24km/h$). The goal of the query is to quantify, on a per player basis, the distribution of run intensities and corresponding distances.

To that end the running analysis query should return two classes of results: (1) current running statistics and (2) aggregate running statistics. The current running statistics report the current running intensity of the player as well as the distance covered with that intensity. The result schema for the current running statistics is given in Listing 3.

```
1 ts_start,
2 ts_last,
3 player_id,
4 intensity,
5 distance,
6 speed
```

Listing 3: Current running statistics output stream schema

Where *ts_start* represents the start of the run with the given intensity, *ts_stop* represents the time stamp of the last event which updated the given result. *player_id* is the identifier of a player for which the measurement is made, and *intensity* describes the intensity of the run. Running intensities can be returned as integers with 0 corresponding to standing and 5 corresponding to sprint. *distance* is the length of the run with the given intensity. Distance should be calculated in meters in the horizontal plane only. *speed* is the average speed of the given intensity run. The current running statistics result stream should be returned with a frequency of at most $50Hz$. Intermediate values can be aggregated using an average function.

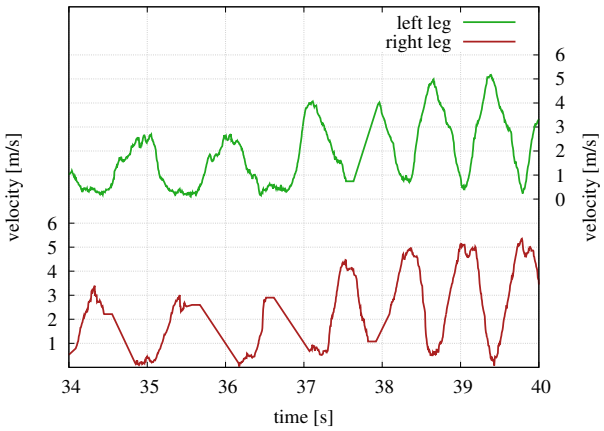

Figure 4: Velocity of the left and right leg of a player

Aggregate running statistics provides a cumulative view on the running performance of a given player. It contains information on all intensities a player has been running with in the past, as well as distances covered for each intensity. The aggregate running statics should be calculated for different window lengths, thus allowing to compare the current as well as past performance of the given player. The result schema for the aggregate running statistics is given in Listing 4.

```
1 ts,
2 player_id,
3 standing_time, standing_distance,
4 trot_time, trot_distance,
5 low_time, low_distance,
6 medium_time, medium_distance,
7 high_time, high_distance,
8 sprint_time, sprint_distance
```

Listing 4: Current running statistics output stream schema

where *ts* represent the latest time stamp which updated the statistics, the *player_id* is the player identifier, *intensity_time* is the time a given player spent in the given intensity (in milliseconds), *intensity_distance* is the distance covered with the given intensity (in meters). The aggregate running statistics must be calculated using following time windows: 1 minute, 5 minutes, 10 minutes, 20 minutes and the whole game duration. Each window will provide its results in a separate stream – the aggregate running statistic query will produce 5 result streams in total. Each result stream should be updated with a frequency of at most $50Hz$.

Every running intensity, which has been active for less than a second must be counted on top of the next intensity with a duration longer than 1 second. For example, if a player is in a trot state for a 2 seconds, then in a low speed run state for 0.8 seconds, and then in a medium speed run state for 2 seconds, the time of the low speed run must be counted on top of the medium speed run. Please note that the requirement to count only intensities active for at least one second requires to delay the output until a reliable measurement has been made. Specifically, this might imply longer delays in output if a player changes frequently between run intensities – e.g. oscillating between trot and low intensity run.

In order to accommodate for the noise in the raw velocity

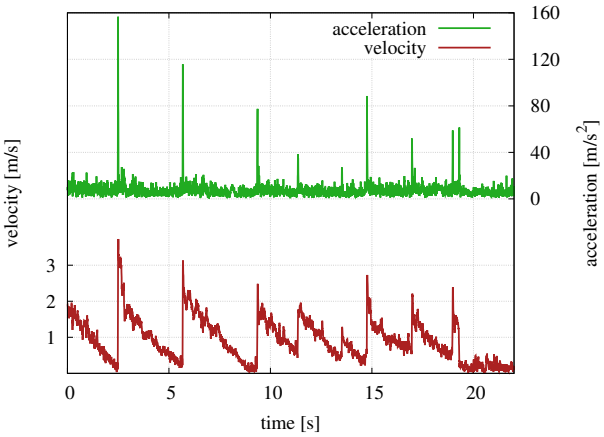

Figure 5: Velocity and acceleration of the ball

measurements as well as to accommodate for the location of the sensors, the actual speed of the run should be computed using both sensors. Figure 4 shows that the speed of individual sensors does not reflect the actual speed of the player. Moreover, it has to be noted that due to interference it is possible that the system has not received certain position events – see for example the right leg velocity between 35.5 and 36.2 seconds. A good approximation of the player speed is the average speed of both left and right sensor over the last second.

4.2 Ball Possession

The goal of the ball possession query is to calculate the ball possession for each of the players as well as for each team. A player (and thereby his respective team) can obtain the ball whenever the ball is in his proximity and he hits it. A ball is in proximity of the player when it is less than one meter away from him. The distance of one meter applies to the distance between the sensor within the ball and any of the two sensors in the player's shin guards. A ball is hit whenever its acceleration or velocity peaks.

A ball will stay in the possession of a given player until another player hits it, the ball leaves the field, or the game is stopped. Specifically, a ball may leave the player's proximity and will still remain in his possession. The ball possession is calculated as time between the first ball contact (hit) and the last ball contact (hit).

Reliable detection of a ball hit is difficult. For the purpose of the Grand Challenge two approaches can be used. The first approach relies on detection of the acceleration threshold of 55 m/s^2. This value depends heavily on the fitness of the players as well as hit type. In professional games values of up to 100 m/s^2 are common. The second approach relies on monitoring of the changes in the ball velocity whenever the observed velocity changes by more than 100% within 15 milliseconds a ball hit may be assumed. Figure 5 illustrates both the acceleration as well as velocity values for a ball in play over the period of 22 seconds.

The ball position query should return two classes of results: (1) per player ball possession stream and (2) per team ball possession. The per player ball possession stream should contain following information – see Listing 5.

```
1 ts, player_id, time, hits
```

Listing 5: Per player ball possession output stream schema

where *ts* is the latest time stamp of the event which lead to the update of the ball possession, *player_id* is the identifier of the player owing the ball, *time* is the total time of the ball possession for a given player, and *hits* is the total number of ball contacts for a given player. Per player ball possession result stream should be updated with the frequency of at most 50Hz.

The per team ball possession result stream must contain following statistics – see Listing 6.

```
1 ts, team_id, time, time_percent
```

Listing 6: Team ball possession output stream schema

where *ts* is the latest time stamp of the event which lead to the update of the team's ball possession, *team_id* is the team identifier. A team should be identified either as "A" or "B". *time* is the total time of the ball possession for a given team, *time_percent* is a percentage of the ball possession time for a given team w.r.t. the total ball possession time of both teams. The per team ball possession should be calculated using five different time windows: 1 minute, 5 minutes, 10 minutes, 20 minutes and the whole game duration. In total five team ball possession result streams should be returned. Each of the team ball possession result streams should be updated with the frequency of at most 50Hz.

4.3 Heat Map

The goal of the heat map query is to calculate statistics about the presence of each player in a given region of the playing field. For this purpose we define a fixed size grid with X columns along the x-axis and Y rows along the y-axis. Each row and column is of equal size. The parameters X and Y should be implemented with following values 8 and 13 (a grid of 104 cells), 16 and 25 (a grid of 400 cells), 32 and 50 (a grid of 1600 cells), 64 and 100 (a grid of 6,400 cells), respectively. The goal of the heat map query is to calculate how long each player spent in the given cell. The system should return results for all grid sizes in parallel using a separate result stream for each grid configuration.

The system must calculate for each cell and each player the percentage of time that the given player spent in the respective cell. The statistics should be calculated using five different time windows: 1 minute, 5 minutes, 10 minutes, 20 minutes and the whole game duration. In total 20 result streams must be returned by the system. Each result stream must be updated once per second and contain the following information – see Listing 7

```
1 ts, player_id,
2 cell_x1, cell_y1, cell_x2, cell_y2,
3 p_time
```

Listing 7: Heat map output stream schema

ts represent the time stamp of the latest event updating the heat map statistics. *player_id* is the player identifier for which the heat map is returned. *cell_x*1, *cell_y*1, *cell_x*2, *cell_y*2 are the coordinates of the lower left and upper right corner of the cell, respectively. *p_time* is the percentage of

time that given player spent in the cell during the period specific to the result stream.

Given the large total number of cells in the result streams following optimizations are possible. Instead of returning one event per cell, it is possible to return one event per whole heat map. The event schema should be the following – see Listing 8.

```
ts, player_id,
cell1_x1, cell1_y1, cell1_x2, cell1_y2, p_time1,
cell2_x1, cell2_y1, cell2_x2, cell2_y2, p_time2,
cell3_x1, cell3_y1, cell3_x2, cell3_y2, p_time3,
...
```

Listing 8: Combined heat map output stream schema

All fields are equivalent to the ones specified in Listing 7. The major difference is the fact that this event schema contains all cells for a given heat map resolution. A further optimization is possible where only fields with changed values (since the last update event) are included.

4.4 Shot on Goal

The aim of the shoat on goal query is to detect when a player hits the ball in an attempt to score a goal. A shot on the goal is defined as any shot that would hit (or closely miss) the goal of the opposing team. Shots on goal include unsuccessful attempts that are, e.g., blocked by a defending player or saved by the goal keeper.

A shot is detected if a player with hits the ball with a minimal acceleration of 55 m/s^2, and the projected movement of the ball would cross the goal area within 1.5 seconds after the hit. The goal areas are defined as rectangles with the following coordinates:

- Goal area 1: $x > 22578.5$ and $x < 29898.5$ with $y = 33941.0$ and $z < 2440.0$
- Goal area 2: $x > 22560.0$ and $x < 29880.0$ with $y = -33968.0$ and $z < 2440.0$

A hit distorts the speed values of the ball. Therefore, data is preprocessed by a Kalman-filter and stabilizes over time – see Figure 5. To allow for corrective measures we do not require shot detection until the ball has moved 1 meter away from the hit location.

It is open to the challenge participants to decide to which degree the ball movement projection should considers the physics of a flying ball. A base-line solution extrapolating the motion vector is acceptable. Extended solutions taking into account, e.g., the curvature of the ball trajectory (bent balls) are also possible.

For the duration of the shot the result stream should be updated with motion values of the ball and the ID of the shooting player according to the following schema – see Listing 9.

```
ts, player_id,
x, y, z,
|v|, vx, vy, vz,
|a|, ax, ay, az
```

Listing 9: Shot on goal output stream schema

ts represent the time stamp of the latest event updating the shot on goal statistics. *player_id* is the identifier of the player trying to score a goal. x, y, and z are the current coordinates of the ball. $|v|$, vx, vy, and vz represent the current velocity and velocity vectors of the ball. $|a|$, ax, ay, and az represent the current acceleration and acceleration vectors of the ball.

The result stream should be updated with the frequency of the sensor data until an exit condition occurs. Exit conditions are: (1) the ball leaves the field, or (2) the direction of the ball movement changes so that (the proximity of) the goal area would no longer be hit.

5. SUMMARY

In this paper we present the ACM DEBS 2013 Grand Challenge. We provide a detailed description of the Grand Challenge data set along with a description of four analytical queries. The goal of this paper is to provide a long living reference for all willing to use the data and queries for benchmarking and evaluation of event based systems beyond the scope of the ACM DEBS 2013 Grand Challenge. To that end the authors of this paper explicitly encourage comments or questions regarding both the data as well as queries specified in this paper.

6. ACKNOWLEDGMENTS

This work is supported by the Fraunhofer Institute for Integrated Circuits IIS. We would like to thank the researchers at the Fraunhofer IIS for sharing their work and RedFIR system with us. This work is also is partially sponsored by European Commission's Seventh Framework Program under grant agreement No. 257843 — project SRT-15 (http://srt-15.eu). Finally, we would like to thank all ACM DEBS 2013 Grand Challenge participants who provided us with continuous feedback and improvements regarding the description of the DEBS Grand Challenge. Thank You!

7. REFERENCES

[1] A. Arasu, S. Babu, and J. Widom. The cql continuous query language: semantic foundations and query execution. *VLDB*, 15(2):121–142, 2006.

[2] S. Chaudhuri, U. Dayal, and V. Narasayya. An overview of business intelligence technology. *Communications of the ACM*, 54:88–98, 2011.

[3] T. v. d. Grün, N. Franke, D. Wolf, N. Witt, and A. Eidloth. A real-time tracking system for football match and training analysis. In *Microelectronic Systems*, pages 199–212. Springer Berlin, 2011.

[4] Z. Jerzak, T. Heinze, M. Fehr, D. Gröber, R. Hartung, and N. Stojanovic. The DEBS 2012 Grand Challenge. In *DEBS 2012: 6th ACM International Conference on Distributed Event-Based Systems*, Berlin, Germany, July 2012. ACM.

[5] N. Leavitt. Complex-event processing poised for growth. *Computer*, 42(4):17–20, 2009.

[6] T. Nowak and A. Eidloth. Dynamic multipath mitigation applying unscented kalman filters in local positioning systems. *International Journal of Microwave and Wireless Technologies*, 3:365–372, 2011.