MPI/OpenMP Project
Academic year 2017-2018

Luca Mottola
luca.mottola@polimi.it

Overview

- You are to develop a system to provide analytics over high velocity sensor data originating from a soccer game
- The data comes from a number of wireless sensors embedded in the shoes and a ball used during a soccer match
- The data spans the whole duration of the game

Data Set

- Data originates from sensors located near the players' shoes (1 sensor per leg) and in the ball (1 sensor)
- The goal keeper is equipped with two additional sensors, one at each hand
- The players' sensors produce data with 200Hz frequency
- The sensor in the ball produces data with 2000Hz frequency
- Every position event describes the position of a given sensor in a 3D reference system
- The center of the playing field is at (0, 0, 0)

Data Set

- The event schema is
  sid, ts, x, y, z, |v|, |a|, vx, vy, vz, ax, ay, az
  where
  - sid is a sensor id which produced the position event
  - ts is a timestamp in picoseconds
  - ts equal to 10753295594424116 is the start of the game and ts equal to 14879639146403495 is the end of the game
  - x, y, z describe the position of the sensor in mm
  - the origin is the middle of a full size football field
  - you may ignore the remaining fields

Data Set

- A separate data stream is provided for referee events, which includes the time when a game was paused and the time when a game was resumed
- Moreover, referee events contain the **time** and **player_ids** for substitutions
- The mapping between **player_ids** and **team_ids** as well as between **sensor_id** and **player_id** is provided in a metadata file
- The game where data was collected was played on a half-size field with teams of seven players each
- The game duration was two halves of thirty minutes

Your Task

- Using MPI and/or OpenMP you are to create a software that computes the real-time statistics of ball possession during the game
- A player is considered in possession of the ball when
  - He is the player closest to the ball
  - He is not farther than K meters from the ball
- Ball possession is undefined whenever the game was paused
- The statistics need to be output for every T time units as the game unfolds
- The statistics accumulate every T time units

Example

- Using K = 1 and T = 30
- A player is in possession of the ball when he is the closest player
  - …but he is no more than 1 meter away
- The software outputs accumulated real-time statistics for every 30 seconds of play
  - For a game lasting one hour, the software returns 120 outputs with the final one reporting statistics for the entire game

Files

- Raw sensor data for the game can be downloaded from goo.gl/BEjFNr (2.6 GB)
  - All data has been aggregated is a single file and is sorted by time stamps
- The video recording of the game (vertical view, static camera) can be downloaded from goo.gl/f6Ze99 (1st half, 1.7 GB) and goo.gl/cBSVkW (2nd half, 1.7GB)
- The metadata file that contains player's names and associated transmitter ids, detailed field coordinates etc. can be downloaded from goo.gl/fhduAx (10 kB)
- Approximate ball possession statistics can be downloaded from goo.gl/C3jqAY (10 kB)
  - These statistics have been created manually and can serve as an aid in validating your results
- The files include additional data you may ignore

Input and Outputs

- Your software needs to take as input
  - An integer value defining K, ranging from 1 to 5
  - An integer value defining T, ranging from 1 to 60
- It needs to output a string for every T time units of play, arbitrarily formatted, with the ball possession statistics
  - For each player
  - For the whole team
- The output may be directed to stdout or a file

Evaluation

- You are to turn in
  - Complete code
  - A (max) 3 page document illustrating your design
- The grade will be determined by
  - Appropriate use of MPI/OpenMP
  - The effectiveness of the parallelization techniques
  - Quality of the implementation and documentation

Note

- This project is a simplified version of the DEBS 2013 Grand Challenge
- Check goo.gl/AxbCLu for more details