5.2) How to synchronise data ?

How to extract all the data for two given sensors and for a given sequence and extract the data for the second sensor acquired at the same time as the data of the 1st sensor ?

dataSynchronisation.cpp

```
#include <iostream>
#include <sstream>
#include <vector>
#include <string>

#include "containers.h"
#include "config.h"

int main(int argc, char* argv[])
{

    setlocale (LC_ALL,"C");

    //  Usage: ./DataSynchronisation sequencePath sensorAlias1 sensorAlias2
   if(argc==4)
    {
      std::string sequencePath(argv[1]);
      std::string sensor1Alias(argv[2]);
      std::string sensor2Alias(argv[3]);

      ConfigTable ctable;
      std::stringstream msg;
      msg << CONFIGFOLDERPATH << "/config_table";
      ctable.readTable(msg.str());

      std::vector<Unit> data1;
      std::string
timestamps_file_name1=ctable.table[sensor1Alias].folderName+ctable.table[sensor1Alias].su
ffix;
      std::string
sensor_data_folder1=sequencePath+"/"+ctable.table[sensor1Alias].folderName+"/";
      getReadings(ctable.table[sensor1Alias].sensorName, sensor_data_folder1,
timestamps_file_name1, ctable.table[sensor1Alias].tfm, data1, TimestampInterval());

      std::vector<Unit> data2;
      std::string
timestamps_file_name2=ctable.table[sensor2Alias].folderName+ctable.table[sensor2Alias].su
ffix;
      std::string
sensor_data_folder2=sequencePath+"/"+ctable.table[sensor2Alias].folderName+"/";
      getReadings(ctable.table[sensor2Alias].sensorName, sensor_data_folder2,
timestamps_file_name2, ctable.table[sensor2Alias].tfm, data2, TimestampInterval());

      // synchronise
      std::vector<int> synchroMap;
      synchronizePair(data1, data2, synchroMap);

      unsigned int i=0;
      while(i<synchroMap.size())
      {
            if(synchroMap[i]!=-1) // synchro OK
            {
               std::cout << "The " << i << "th data of sensor " << sensor1Alias << " is acquired
approximately at the same time as the " << synchroMap[i] << "th data of sensor " <<
sensor2Alias << std::endl;
            }
      }
    }

  return 0;
}
```

Explanation :

- All data will be stored as a Unit object (it is a generic container). Data from the sensor 1 are stored in the vector :

```
std::vector<Unit> data1;
```

- Data from the sensor 2 are stored in the vector :

```
std::vector<Unit> data2;
```

- Both data are synchronised. The data from the sensor 1 are used as the reference data i.e. a data of the 2nd sensor is seek for each data of the 1st sensor. The map contains the index of the 2nd sensor for a given index of the 1st sensor. The value is -1 if there is not any synchronised data.

```
std::vector<int> synchroMap;
synchronizePair(data1, data2, synchroMap);
```

CMakeLists.txt

```
project(DataSynchronisation)
cmake_minimum_required(VERSION 2.6)

set( CMAKE_BUILD_TYPE Release )
add_definitions( -Wall )

set(libIPDS_include_dir "/home/courbon/IPDS/IPDSCodes/IPDSLib/include/")
set(libIPDS_lib_dir "/home/courbon/IPDS/IPDSCodes/IPDSLib/build/lib/")


# IPDS library inclusion
include_directories(${libIPDS_include_dir})
link_directories(${libIPDS_lib_dir})

add_executable (${PROJECT_NAME} dataSynchronisation.cpp)

target_link_libraries(
   ${PROJECT_NAME}
   IPDS
)
```

Note that the paths to the IPDS include and library (libIPDS_include_dir and libIPDS_lib_dir) have to be modified to suit your set-up.

(the source code is in the archive DataSynchronisation.zip)