DEPARTMENT OF COMPUTER SCIENCE

Automatically Correcting for a Time Drift in Accelerometers using RGBD C

Connor Williams

A dissertation submitted to the University of Bristol in accordance with the requirements of the degree of Bachelor of Science in the Faculty of Engineering.

Tuesday 5th April, 2016

Declaration

This dissertation is submitted to the University of Bristol in accordance with the requirements of the degree of BSc in the Faculty of Engineering. It has not been submitted for any other degree or diploma of any examining body. Except where specifically acknowledged, it is all the work of the Author.

Connor Williams, Tuesday 5th April, 2016

Contents

Abstract

Obesity, depression, stroke, falls, cardiovascular and musculoskeletal disease are some of the biggest health issues and fastest-rising categories of healthcare costs in the UK. The associated expenditure is widely regarded as unsustainable and the impact on quality of life is felt by millions of people in the UK each day. With a rapidly ageing population - could technology be the answer to some of these problems?

SPHERE (a Sensor Platform for HEalthcare in a Residential Environment) is developing a number of different sensors that will combine to build a picture of how we live in our homes. This information can then be used to spot issues that might indicate a medical or well-being problem.

The technology could help by:

- Predicting falls and detecting strokes so that help may be summoned.
- Analysing eating behaviour - including whether people are taking prescribed medication.
- Detecting periods of depression or anxiety.

[1]

Within the SPHERE research group, one of the sensors being developed is a wearable accelerometer. This accelerometer induces a drift between the true time and the local time on the device large enough that the wearable needs to be re-synchronised for every use.

This thesis looks to consider video data from RGBD cameras around the house, which provides data on where the person is, together with the acceleration data from the wearable sensor - and with this data it should be possible to automatically correct the time drift so that the accelerometer and the video remain synchronised.

Supporting Technologies

- Python for implementation.
- SciPy: Numpy, Matplotlib.... for help.

Notation and Acronyms

Temporal Distortion	:	No Distortion, Shift, Drift, Re-calibration, Jumps ...

Acknowledgements

Chapter 1

Contextual Background

This project could be broken down in the following steps:

1. Consider different types of temporal distortions which could take place on a device such as this one.
2. Implement a program which generates data and can apply any combination of the different types of temporal distortion.
3. Devise a method to compensate for the different temporal distortions.
4. Given a synchronised set of data from the accelerometer and the cameras, introduce artificial drifts between data sets and assess how well the methods work.
5. Apply this method to real drifted data which has already been collected in the SPHERE house in three locations; hallway, living room and kitchen.

Implicit to all of these steps will is a fair amount of data formatting and visualisation. I use Python for the implementation with SciPy which is an open source library of scientific tools for Python.

Chapter 2

Technical Background

2.1 Initial solution ideas

In a centralized system the solution is trivial; the centralized server will dictate the system time. Cristian's algorithm and the Berkeley Algorithm are some solutions to the clock synchronization problem in a centralized server environment. In a distributed system the problem takes on more complexity because a global time is not easily known. The most used clock synchronization solution on the Internet is the Network Time Protocol (NTP) which is a layered client-server architecture based on UDP message passing. Lamport timestamps and vector clocks are concepts of the logical clocks in distributed systems.

Cristian's algorithm Cristian's algorithm relies on the existence of a time server. The time server maintains its clock by using a radio clock or other accurate time source, then all other computers in the system stay synchronized with it. A time client will maintain its clock by making a procedure call to the time server. Variations of this algorithm make more precise time calculations by factoring in network radio propagation time.

Berkeley algorithm The Berkeley algorithm is suitable for systems where a radio clock is not present, this system has no way of making sure of the actual time other than by maintaining a global average time as the global time. A time server will periodically fetch the time from all the time clients, average the results, and then report back to the clients the adjustment that needs be made to their local clocks to achieve the average. This algorithm highlights the fact that internal clocks may vary not only in the time they contain but also in the clock rate.

Often, any client whose clock differs by a value outside of a given tolerance is disregarded when averaging the results. This prevents the overall system time from being drastically skewed due to one erroneous clock.

Network Time Protocol Network Time Protocol a class of mutual network synchronization protocol that allows for use-selectable policy control in the design of the time synchronization and evidence model. NTP supports single inline and meshed operating models in which a clearly defined master source of time is used ones in which no penultimate master or reference clocks are needed.

In NTP service topologies based on peering, all clocks equally participate in the synchronization of the network by exchanging their timestamps using regular beacon packets. In addition NTP supports a unicast type time transfer which provides a higher level of security. NTP performance is tunable based on its application and environmental loading as well. NTP combines a number of algorithms to robustly select and compare clocks, together with a combination of linear and decision-based control loop feedback models that allows multiple time synchronization probes to be combined over long time periods to produce high quality timing and clock drift estimates. Because NTP allows arbitrary synchronization mesh topologies, and can withstand (up to a point) both the loss of connectivity to other nodes, and ẗfalsetickersẗhat do not give consistent time, it is also robust against failure and misconfiguration of other nodes in the synchronization mesh.

NTP is highly robust, widely deployed throughout the Internet, and well tested over the years, and is generally regarded as the state of the art in distributed time synchronization protocols for unreliable networks. It can reduce synchronization offsets to times of the order of a few milliseconds over the public Internet, and to sub-millisecond levels over local area networks.

A simplified version of the NTP protocol, SNTP, can also be used as a pure single-shot stateless master-slave synchronization protocol, but lacks the sophisticated features of NTP, and thus has much lower performance and reliability levels.

Clock Sampling Mutual Network Synchronization CS-MNS is suitable for distributed and mobile applications. It has been shown to be scalable over mesh networks that include indirectly linked non-adjacent nodes, and compatible to IEEE 802.11 and similar standards. It can be accurate to the order of few microseconds, but requires direct physical wireless connectivity with negligible link delay (less than 1 microsecond) on links between adjacent nodes, limiting the distance between neighboring nodes to a few hundred meters.

Precision Time Protocol Precision Time Protocol (PTP) is a master/slave protocol for delivery of highly accurate time over local area networks

Synchronous Ethernet Synchronous Ethernet uses Ethernet in a synchronous manner such that when combined with synchronization protocols such as Precision Time Protocol in the case of the White Rabbit Project, sub-nanosecond synchronization accuracy may be achieved.

Reference broadcast synchronization The Reference Broadcast Synchronization (RBS) algorithm is often used in wireless networks and sensor networks. In this scheme, an initiator broadcasts a reference message to urge the receivers to adjust their clocks.

Reference Broadcast Infrastructure Synchronization The Reference Broadcast Infrastructure Synchronization (RBIS) protocol is a master/slave synchronization protocol based on the receiver/receiver synchronization paradigm, as RBS. It is specifically tailored to be used in IEEE 802.11 Wi-Fi networks configured in infrastructure mode (i.e., coordinated by an access point). The protocol does not require any modification to the access point.

Global Positioning System The Global Positioning System can also be used for clock synchronization. The accuracy of GPS time signals is 10 ns[7] and is second only to the atomic clocks upon which they are based.

Chapter 3

Project Execution

3.1 Data Generator

This section is about my data generator:

- Why is it necessary?
- What features does it have?
- Why does it have these features? (Linked to different temporal distortions)

3.2 Methods

This section will be about the methods I have found which solve the different types of temporal distortion.

- What are the methods?
- How well do the methods work?
- Do other temporal distortions affect the performance?
- What combination of distortions are amendanble?

Chapter 4

Critical Evaluation

4.1 SPHERE Data Tests

Evaluation of SPHERE tests here.

4.2 Results

Results here.

Chapter 5

Conclusion

Future work: Use these assessments as a means of rejecting false positive skeletons. Skeletons are the format in which the data comes from the cameras. It contains information on the location of the main joints of the person i.e. elbow, wrist, knee etc.

Bibliography

[1] SPHERE. Sphere website, 2016. [Accessed 2nd April 2016].