KTH Technology
and Health

Multi-Sensor Data Synchronization using Mobile Phones

Jonas Wåhslén

Licentiate Thesis
June 2013
TRITA-STH Report 2013:5

Department of Computer and Electrical Engineering
School of Technology and Health,
KTH (Royal institute of Technology)

Academic dissertation which with permission from Kungliga Tekniska Högskolan (Royal Institute of Technology) in Stockholm is presented for public review for passing the Licentiate of Technology degree on Thursday June 12 in Lecture hall 3-515, Alfred Nobels allé 10, Huddinge, Sweden.

TRITA-STH Report 2013:5
ISSN: 1653-3836
ISRN/KTH/STH/2013:5-SE
ISBN 978-91-7501-8

Abstract

Body sensor networking is a rapidly growing technology. Today wearable sensors are used to measure and monitor e.g. pulse, temperature, skin conductance, heart activity, and movement (through GPS or inertial measurement units). Mobile phones can act as coordinating nodes in wireless personal area networks used in home automation, healthcare, sport and wellness e.g. to measure pulse and distance. Integration of data from multiple sources sensors (data fusion) means that data from each sensor node needs to be associated with data from other sensor nodes sampled at approximately the same time. Accurate methods for time synchronization are therefore a necessary prerequisite for reliable data fusion.

This thesis studies time synchronization problems in Bluetooth piconets between multiple wireless sensor nodes connected to a mobile phone that acts as coordinating node. Three different algorithms to enable correct data fusion have been developed, implemented and evaluated. The first is a single clock solution that synchronizes multiple wireless sensor nodes based solely on the mobile phone's clock. The other two algorithms synchronize the clocks in sensor nodes to the clock in the coordinating node.

Keywords: wireless body sensor networks, wireless personal area network, sport technology, Bluetooth performance, time synchronization, data fusion, mobile phone application.

Acknowledgments

During the last years, life has dramatically changed. I now have a wife and two kids, and together we have faced the roller coaster of life. It has been a ride where hopefully the worst hills are behind us. Simultaneously I have continued my research, it would not have happen without a supporting loving family, Jeanette, Noa and Max, I Love You! Equal important to complete this thesis were my supervisor Associate Professor Thomas Lindh. You have been an invaluable resource, for input and to crystallize topic of research interest. This is also true for my research college Ibrahim Orhan, who also co-authored the papers. Our discussions have been vital for solving the challenges we faced. Thanks Martin Eriksson for presenting me to Olympic Performance Center and your vision of how future sport technology should look like, and to Dennis Sturm for seeing to my algorithms included in a real product used by elite athletes. To my colleges Anders Lindström for highlighting the importance to difference double from floats and Reine Bergström for coffee breaks where our discussion have merged technology and philosophy. To head of department, Magnus Brenning from providing me with an iPad to complete the last tests for this thesis. And I am also sincerely grateful to the former dean of KTH STH, Inge Jovik, for giving me the opportunity to do my research studies besides my ordinary lecture employment. And I am also sincerely grateful to the former dean of KTH STH, Professor Lars-Åke Brodin, for giving me the opportunity to continue my research to a licentiate degree. And final a thanks to my mother and father, Tuula and Bertil for refusing to listening to the guidelines of the study counselor that told you that 3 year technical high school were not for me. But I forgive the study counselor. It must be annoying to have a 9 grader telling you it is wrong when solving a math question on the black board. And finally a sorry to my sisters, Jenny and Jessica for all the toys I broke in the pursuit of understanding how they worked.

Table of contents

1 Introduction

Less than five years ago JavaMe and Symbian were the preferred mobile platforms to create a mobile application that should "reach the masses". Today Apple's iOS and Google's Android are the dominating mobile platforms. The most fundamental change is however the way applications are distributed. Previously, an application needed to be installed on the phone when it was sold to reach any substantial market penetration. Today this is no-longer necessary. App stores, like the one for iPhone, have completely changed the way applications for mobile phones can be distributed. It has never been easier to go from an idea to reality for mobile applications. The inevitable consequent of Moore's Law; is an increasing processing power making todays mobile phones "smart". As the mobile platforms have evolved (to mobile phones) so have the mobile network. Mobile provides a higher capacity then wired network did five years ago. Consequently, with a mobile phone acting as a coordinating node, wireless personal area networks have emerged as a feasible communication infrastructure in e.g. health monitoring and sports [1]. Samples of data from a number of wearable sensors can be used for proper feedback to an athlete during training or to patients for motor training. Significant advances have been made in the field, which makes it possible for a coach and athlete to monitor kinematic and kinetic performance parameters unobtrusively during the everyday training, i.e. outside the laboratory [2], [3] and [4].

This thesis started with a vision of a mobile platform for elite athletes that could send real-time feedback base on biomechanical movement e.g. biofeedback [5]. This vision included multiple problems that needed to be solved. The focus of this thesis has been time synchronization to enable analysis of data from multiple wirelessly connected sensor nodes.

Correctly performed data fusion is crucial for applications in sports, medicine, health and many other fields e.g. the system for flat kayak where two paddle nodes and a foot stretcher node all wireless connected to a mobile phone acting as coordinating node [6]. Our algorithm [7] has been used in the kayak system to synchronize data from the three sensor nodes, making it possible to e.g. analyze the relation between the upper and lower body movements.

Bluetooth is so far the prevailing communication technique between a mobile phone and wireless sensor nodes. Today mobile phones support low energy Bluetooth (v4.0), which extends the possible usage of connected peripheral units. Time synchronization in computer networks, as well as Bluetooth piconets, has been studied in great detail previously [8-14]. However, most of the solutions, and the low-level synchronization functions in Bluetooth, are not available on the application layer [10, 11, 13]. The only access on the application layer is sockets connection. In this thesis we focus on development of algorithm for multiple wireless sensor nodes that can be used on the application layer by software developers.

The thesis is structured as follows. Section 2 includes a short description of the different synchronization algorithms developed. Conclusions and future work are found in Section 3 and finally a summary of original work is in Section 4 follow by the appended papers.

1.1 Aim

The main purpose of this thesis work has been to develop, implement and evaluate synchronization algorithms to be used on the application level to enable integration of data from multiple sensor nodes in a Bluetooth piconet.

The implementations should take program capability on the sensor nodes, memory constrains, code complexity and network setup into account.

1.2 Contributions of the thesis

The major contributions of this thesis are:

- Development of algorithms for time synchronization of sensor nodes in a wireless personal area network.
- Analysis of Bluetooth performance with respect to time synchronization related to the specification and different vendor implementations.
- Implementation, testing and evaluation of a new synchronization method that relies solely on the clock in the coordination node, which means the timestamps form the sensor nodes, are not needed.
- Implementation, testing and evaluation of synchronization methods for Bluetooth piconets, inspired by time synchronization in distributed networks and the Internet standard Network Time Protocol.
- Implementation of a synchronization method without use of dynamic memory, which can be used on "memory constrained" wireless sensor nodes.
- Comparison and evaluation of the time synchronization algorithms in Bluetooth and Zigbee/IEEE 802.15.4 networks.

2 Time synchronization in Bluetooth piconets

In order to successfully analyse data sampled from multiple wireless sensor nodes, data synchronization between every sensor node is needed. It means that sample s_i from sensor node S_A is associated with sample s_j from sensor node S_B, if the samples' timestamps are the closest in time. Data synchronization based on timestamps when samples arrive to the mobile phone via a socket connection do not work as there is a variation between the sampling time and the time the sample gets time stamped at the mobile phone.

Our test results show that data fusion of samples based on the receiving timestamps may have synchronization errors up to as much as 50ms for 100Hz sampling frequency. It means that sample s_i sent from sensor node S_A is perceived by the mobile phone as being sent at approximately the same time as sample s_j from sensor node S_B, but should in fact be associated with a sample sent 50ms earlier or later (number s_{j-5} or s_{j+5}). Our observation shows that this error grows as the number of sensor nodes connected to the coordinating node increases.

Fig. 1 shows a block diagram of multiple wireless sensor nodes connected to a mobile phone via Bluetooth. The embedded software on the mobile phone uses the Bluetooth socket API to receive data from the sensor nodes.

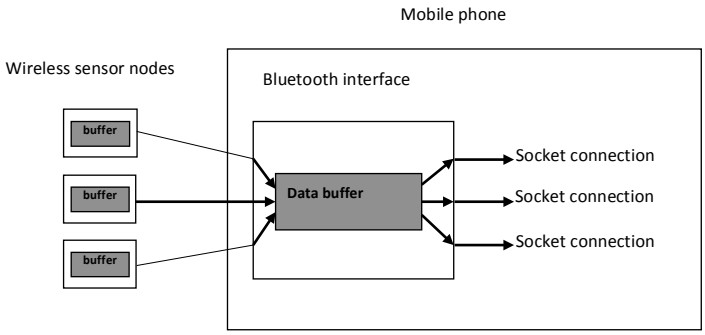

Fig. 1. A block diagram of how a mobile phone handles data input from wireless sensor nodes. Data from the sensor nodes pass two buffers; the outgoing data buffer on the sensor node, and the incoming data buffer on the mobile phone.

The application inserts a timestamp when the sample is read from the mobile phones incoming buffer. The time period, from the sampling event on the sensor node until the sample is available for an application on the mobile phone, varies randomly. This stochastic variable, Δt, is defined as the difference between the actual sample time on the sensor node and the time the sample is read by the application from the buffer on the mobile phone. The propagation time can be neglected due to the short distance between sensor nodes and mobile phone. The waiting time in the buffers, the outgoing buffer on the sensor node and the incoming buffer on the mobile phone, varies randomly. In this thesis the buffer on the wireless sensor node and buffer on the mobile phone are treated like one virtual buffer (shaded parts in Fig. 1).

2.1 Bluetooth performance issues

There are several reasons for the stochastic variable Δt. Based on trace files from a Bluetooth sniffer (Frontline [18]) two different traffic patterns that result in synchronization error were detected.

The first caused detected were due to sleep time, where a coordination node acts as Bluetooth master and a wireless sensor node (Shimmer [19]) acts as slave. The sampling time is constant but the slave transmits the samples in bursts separated by gaps in sleep time periods. The gaps between these bursts occur when the master sleeps to save energy. This occurs after having master requesting data from the slave without any response from the slave. In Bluetooth networks, slaves are only permitted to send data upon request from the master. Even though the slave has data to transmit during the idle period, it has to wait for the master to be able to transmit again. The observed idle period is around 20ms (which in this would be the idle period contribution to the time Δt).

Another reason for the stochastic time difference between the sampling time and the actual transmission time from the Bluetooth radio is the effect on inter-sending times when the number of slaves in a Bluetooth piconet increases. When additional slaves are connected to the master the transmission times become more scattered and samples are often sent in bursts, interleaved by gaps. The time delay, Δt, will vary depending on how master's scheduler is implemented by the Bluetooth vendor. If multiple slaves are connected, a master will not necessarily run a round-robin algorithm when requesting data from the slaves. Therefore, some samples will be buffered for a longer time before sent to the master.

Fig. 2 shows sequence numbers and timestamps on the mobile phone for every sample during a period of 100ms (f_s=300Hz). The triangles in Fig. 2 shows when sample where sent a Bluetooth sniffer have captured this information. Crosses represent timestamps when samples are read by the mobile phone's software. The circles indicate the last sample before a context switch. The aggregated pattern is created on the sending sensor node (the triangles). This pattern is maintained and not changed when the samples are timestamped on the mobile phone. The conclusion from Fig. 2 is that the addition buffer waiting time is created on the Bluetooth slave. The timestamps when samples arrive at the mobile phone have a Δt added to the sampling time on the sensor node. This Δt varies depending on the scheduling between the master and the slaves in the Bluetooth network. In the Section 2.2 and 2.3 we will look on different ways to handle and/or minimize Δt.

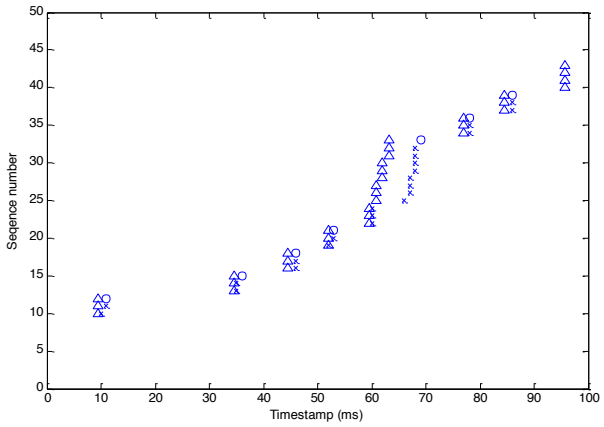

Fig. 2. The graph shows timestamps and sequence numbers for 30 samples from a sensor node connected to the mobile phone during 100ms. The triangles show timestamps when samples are sent. Crosses represent timestamps set on the mobile phone (and circles for the last sample before context switch).

2.2 Single clock synchronization

For wireless sensor nodes that have no mechanisms that support transmission of timestamps to the coordination node e.g. Sparkfun's Witilt [20], data synchronization needs to be done without clock synchronization. This is also the case, if there is a need for re-synchronization of wireless sensor nodes during data transmission. Then the single clock algorithm relying solely on the coordinating nodes clock, can be used for successfully data fusion.

The main idea with the single clock synchronization method is to identify samples with the smallest Δt, apply linear regression and use the result to recalculate the timestamps for all samples. The method can be seen as an estimate of a minimum Δt that consists of the fixed part of the delay Δt (processing time, sending time, propagation time etc.). It is not possible to measure the absolute value of Δt directly for wireless sensor nodes that are not required to have synchronized clocks. The variation of Δt, but not Δt itself, can be measured as the difference between consecutive received samples.

The technique to estimate a minimum Δt, not requiring prior knowledge of sampling rates, is to identify samples with the smallest Δt. Given that the last sample in a queue has spent the shortest time in a buffer. One can assume that an application program on the mobile phone reads samples from the connected wireless sensor nodes, and that the buffers will be emptied each time they are read by the respective thread. The last sample in the queue has spent the shortest waiting time in the buffer. These samples are marked as circles in Fig. 2 each time the thread empties the queue. The prerequisite for the algorithm is therefore that the virtual buffer in Fig. 1 will be completely emptied each time the thread reads samples from the sensor node. The proposed algorithm can be summarized in the following steps for each connected sensor node.

i. Let the samples be time stamped when read by the application program from the incoming buffer on the mobile phone.
ii. Identify a set of samples that are likely to have a minimum Δt.
iii. Apply linear regression to this set of samples to correct the timestamps of the remaining samples.

The algorithm is performed between the mobile phone and all connected sensor nodes.

2.2.1 Implementation

To implement the algorithm, a new synchronization thread is defined (see Fig. 3). Incoming samples from sensor node S_A, S_B and S_C are handled by three separate threads that send the sequence number, the sensor node ID and a timestamp to the new synchronization thread. A_i denotes sample sequence number i sent from sensor node S_A. The queue in the synchronization thread can be used to identify the last sample read by a thread before a context switch to serve a new thread occurs (bold text in Fig. 3).

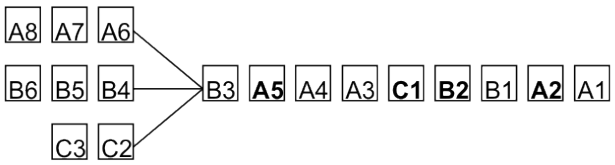

Fig. 3. Three sensor nodes (S_A, S_B and S_C) send their sensor node ID and sequence number for each received sample to the synchronization thread. Samples in bold text represent the last sample read by a thread before the context switch to new thread to serve another sensor node occurs.

A thread can be in one of three different modes; running, ready or suspended. When the thread, in running mode, has finished reading available data, it will return to suspended mode until incoming data is available, then it changes to ready. This last sample, which is read before the thread is suspended and the operating system switches to a new thread, will have the smallest Δt. These samples, in each run of a thread, can be identified in the synchronization thread (bold text in Fig. 3 and marked as circles in Fig. 2).

2.2.2 Result

The process to identify samples with the least Δt may have estimation errors. Fig. 4 shows an example of a thread in running mode, reading samples from a wireless sensor node. A thread in running mode will read all available samples from the incoming buffer. The timestamp for the sample is set when the thread reads the sample. The last sample processed by the thread in running mode, marked with a circle in Fig. 4, has an estimation error, Δt_{error}. The entire sample has not been received and is therefore not available for the thread. There are two alternatives. Firstly, the thread will be suspended since the previous sample already has been processed and there is no sample in the buffer. Once the entire sample has arrived at the buffer, it will be read by the thread next time it is in running mode. Secondly, if the thread still is in running mode as the sample arrives, it will read the sample from

the buffer. The previous sample will no longer be the last sample processed by the thread before it is suspended.

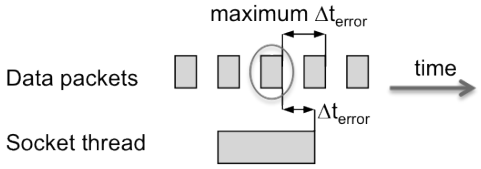

Fig. 4. A thread in running mode, reading samples from the incoming buffer. An example of Δt_{error} and the maximum Δt_{error} is shown.

In [15] modifications of the original algorithm [5] were done. Instead of using linear regression on all samples, the offset was calculated from a sub-set of samples. The synchronization error based on average offset from 50 identified "last sample" ranges from 1.1ms to 3.2ms (99^{th} percentile) depending on the numbers of sensor nodes connected. The results in [15] shows that subsets of the samples can feed the algorithm, which decreases the processing time considerably. The modification also makes it possible to use the algorithm in real-time.

In some cases the single clock synchronization approach results in significantly higher accuracy compared to synchronizing the clocks in multiple wireless sensor nodes that are connected simultaneously. However the method to synchronize the clock is far less complicated.

2.3 Synchronizing the clocks

Given that it is possible to request timestamp from the wireless sensor nodes, local clock synchronization is possible. Algorithms for synchronization of local clocks in wireless sensor networks often apply a combination of methods for distributed systems [16] and the Network Time Protocol [17] for the Internet. The main idea for this method is to synchronize the sensor nodes' (slaves') clocks to a mobile phone (master) clock in a Bluetooth piconet. Fig. 5 sketches the algorithm: The master stores a timestamp, t_1, then sends a synchronization request message, $m_{request}$, to a slave that hosts a sensor node. After receiving $m_{request}$, the sensor node inserts its local time, t_2, into the return message, m_{reply}, before transmitting it back to the master. The master generates timestamp t_3 when m_{reply} is received. The master can, based on these timestamps, determine the offset between its own clock and the slave's clock, and accordingly synchronize the two clocks. When the master during consecutive data acquisition receives a sample from the slave, including its local timestamp, it adds the estimated offset to obtain the common synchronized time. The master performs all offset calculations and implements the synchronization, slaves are passive and simply respond to requests.

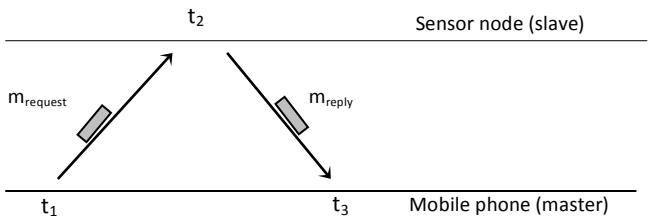

Fig. 5. The figure shows synchronization message sent between a mobile phone and a sensor node, and the timestamps set by the master and slave.

Timestamps t_1 and t_3 are set by the master and timestamp t_2 is set by the slave. Let $t_{round}=t_3-t_1$ (see Fig. 6) be the estimated round-trip time and D_{min} the minimum delay time for $m_{request}$ from the master to the slave. The earliest time $m_{request}$ can reach the slave is t_2+D_{min}, and the latest time $m_{request}$ can reach the slave is $t_2+t_{round}-D_{min}$. The maximum error in estimating the time for $m_{request}$ to reach the slave is $\pm(t_{round}/2-D_{min})$. The original idea in Cristian's algorithm [21] for probabilistic clock synchronization in distributed systems is that a server responds with its time, t_{server}, as a reply to a request from a client. The client sets its clock to $t_{server}+t_{round}/2$. Choosing $t_{round}/2$ as the delay between the readings of t_1 and t_2 minimizes the maximum error. The estimated offset, *o*, between the clocks is the difference between t_1 and t_2 plus the delay between the readings of the clocks (half of the round-trip time), as shown in Fig. 6. The master adds this offset to the local timestamp inserted by the sensor node along with the data samples.

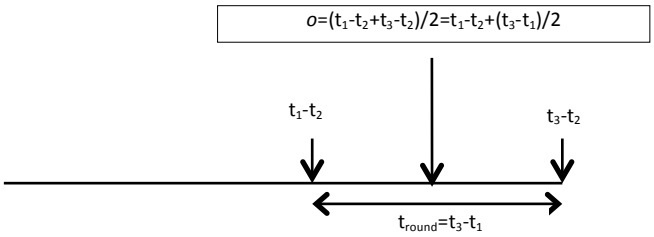

Fig. 6. The figure shows how the offset of the slave's clock relative to the master's clock and the round-trip time is estimated. The error limit for the offset is $\pm(t_{round}/2-D_{min})$.

2.3.1 Implementation

In order to increase the accuracy of the estimated offset, it is possible to request t_2 multiple times, and retrieve multiple sets of t_1, t_2 and t_3. With this subset of time stamps, two different ways to implement the calculation of offset have been examined. The first way is to calculating the average offset from gathered values.

The other is finding the min and max value. In practice it means to find the maximum t_1-t_2 and the minimum t_3-t_2 among a number of overlapping offset intervals (see Fig. 7). The benefit of using the last approach is there is no need to store more then two values, thus it is possible to implement this method without the use of dynamic memory.

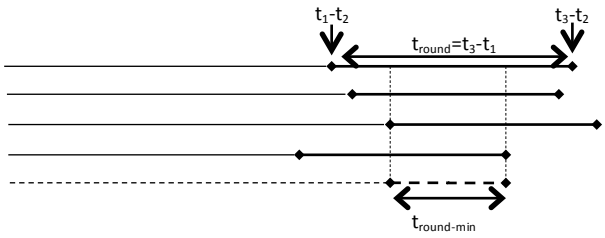

Fig. 7. The figure illustrates the algorithm to select an offset, *o*, with the smallest uncertainty interval, $o \pm t_{round\text{-}min}/2$, where *o* is the centre point of the interval $t_{round\text{-}min}$

2.3.2 Result

The algorithms described in Section 2.3.1 were evaluated by varying two parameters. Firstly, the number of connected sensor nodes, and secondly, the number of repeated loops of $m_{request}$ and m_{reply} (in Fig. 5), where each loop results in a set of timestamp t_1, t_2 and t_3. We found that no substantial improvement is gained by repeating the procedure more than 50 times, which takes less then 1 second. It can often be interrupted after fewer loops. It is also evident that the error increases as the number of connected sensor nodes increases. Two different cases were analyzed regarding the way the sensor nodes where connected i.e. that the sensor node have an established socket connection with the mobile phone. Firstly where only one sensor node was connected and secondly when all sensor nodes were connected during the synchronization. We could then observe a substantially higher synchronization error when all sensor nodes were simultaneously connected, 7ms error if six sensor nodes are connected simultaneously compared to 3.5ms error when the six sensor nodes are connected one at a time (99^{th} percentile).

This is a direct effect of the way Bluetooth controls multiple connections, where the master determines when a slave is permitted to transmit data. When a single sensor node is connected, the master only needs to check if that sensor node has data to send. However, when multiple sensor nodes are connected, the master will schedule the order that the nodes are permitted to transmit. A sensor node may have to wait before sending its sampled data. One reason is that the master will request data from every connected sensor node, regardless of whether they have data to send or not. Another reason is that the scheduling is not necessarily done in a round-robin fashion [5]. The additional error when all sensor nodes are simultaneously connected during the synchronization procedure is approximately twice the error when the sensor nodes are synchronized one at a time. For example, when 6 sensor nodes are connected the time interval between every poll request from the master to a slave will be approximately 7ms. This 7ms will be added to the synchronization error of the connected sensor nodes.

2.4 Ongoing work

In previous sections we have studied data synchronization between two or more wireless sensor nodes connected to a coordinating node (mobile phone). Those synchronization algorithms benefit from a symmetrical network setup. As a result t_{round} will be the same for all wireless sensor nodes and the estimate of the offset t_1-t_2+t_{round}/2 is valid. A scenario with one set of sampled sensor data obtained locally on the coordination node e.g. an accelerometer or a video camera on the mobile phone, and a second set of data transmitted from a wireless sensor node. Two questions arise, firstly how to detect samples with minimum delay (it is not possible to create queues as in single clock algorithm in Section 2.2)? A second question is if half of the round-trip time is a good estimation due to the delay between the reading of the timestamp on the coordinating node and the timestamp on the sensor node. An example of such of a setup is AugmentIT [22], a smart phone app that uses sensor data for augmentation of video recordings of athlete training to be used by coaches. Only off-the-shelf equipment and mobile phones are used to acquire and analyse sensor data and video recordings. AugmentIT will enable augmented reality feedback on kinematic movement to elite athletes. AugmentIT came in third place in KTH Health Technology idea competition. AugmentIT is implemented in Apple's iOS. Analysis of synchronization between two iOS units has also provided interesting input on how to improve and simplify the previous algorithms presented in the thesis.

3 Conclusion and future work

Data fusion is vital to understand and successfully interpret sensor data. In order to successfully do data fusion, samples needs to be correctly associated with each other. This thesis has presented three synchronization algorithms and prototype implementations of different synchronization approaches. The first algorithm relying solely on coordinating nodes clock to correctly do data fusion. However, it is a relatively complex method. Two clock-synchronization algorithms have also been developed that are easier to implement. One of these, clock-synchronization algorithms, can be implemented without the need for dynamic memory.

Our performance evaluations in [15] show that the synchronization error is limited to around 1ms in ZigBee/IEEE 802.15.4 networks when a time synchronization protocol or broadcast messages are used. However, for Bluetooth, which so far is the prevailing communication technique between a mobile phone and wireless sensor nodes, the errors is significant higher and increases as the number of connected sensor nodes in a Bluetooth piconet increases. A technique to improve the synchronization accuracy is by only having one sensor node at the time connected to the coordinating node during time synchronization. Our tests shows that synchronization solely based on the clock in the mobile phone perform better than synchronizing the individual clocks in the sensor nodes (which sometimes is impossible).

The thesis also examines the issues with single node synchronization in AugmentIT i.e. with the mobile phone as a coordinating node record video simultaneously retrieving data from wireless sensor nodes.

However, these are still several open questions, which can be subject for further research.

- The number of connected slaves in the new Bluetooth standard (v4.0) is no longer limited to seven slaves. What happens with the synchronization, given that there are multiple nodes connected?
- Is there a general way to find samples with minimal delay in a single node network setup?
- Is half the round-trip time a good estimation of the delay caused by transmission between a sensor node and coordinating nodes?
- Examine the best way to augment sensor data in a video. What is the best visualization to enhance understanding of sampled data?
- Given an augmented video, where sensor data can be analyzed. Would it be possible to apply machine learning on the sensor data?

4 Summary of original work

Paper I: A novel approach to multi-sensor data synchronization using mobile phones

J. Wåhslén, T. Lindh, M. Eriksson, and I. Orhan.
International Journal of Autonomous and Adaptive Communication Systems, Int. J. Autonomous and Adaptive Communication Systems. Vol. 6, No. 3, pp 289-303, 2012

This journal paper presents a novel method to synchronize samples from wireless sensor nodes without clock synchronization. The method uses the multithreaded input to the mobile phone to accurately associate samples from several wireless sensor nodes with each other. Associating data samples with each is done in post-processing, using linear regression. This makes the method able to handle clock drift and unknown sampling rate on the wireless sensor nodes. Performance issues that cause problems for data synchronization between master and slaves in Bluetooth are highlighted.

Contribution: The author in collaboration with T. Lindh was the original creator of the novel synchronization algorithm, and responsible for implementing and evaluating the algorithm. The journal was written in cooperation with the co-authors.

Paper II: Local time synchronization in Bluetooth piconets for data fusion using mobile phones

J. Wåhslén, I. Orhan and T. Lindh.
Body Sensor Networks (BSN) 2011 International Conference, May 2011, Dallas, US.

This paper presents two methods to synchronize the clocks between the master and the slaves in a Bluetooth piconet in order to perform data fusion of data from multiple wireless sensors nodes with correct timestamps. The two methods are able to perform time synchronization in real-time. The algorithm to do time synchronization is implemented on application level, thus make it possible to use for application developer. The master performs all calculation and implements the synchronization algorithm. One of the method relays on only two stored values thus it do not require dynamic memory for its implementation. Performance issues that cause problems for data synchronization between master and slaves in Bluetooth are highlighted.

Contribution: The author in collaboration with T. Lindh was the original creator of the two methods described in the papers, and the main contributor of the performance experiments and analysis of time synchronization in Bluetooth piconets. The paper was written in cooperation with the co-authors.

Paper III: Performance evaluation of time synchronization and clock drift compensation in wireless personal area networks

J. Wåhslén, I. Orhan, D. Sturm and T. Lindh.

7th International Conference on Body Area Networks, September 2012 Oslo, Norway

This paper presents the results from performance evaluations of algorithms for time synchronization based on three different approaches; one that synchronizes the local clocks on the sensor nodes, a second that uses a single clock on the receiving node, e.g. a mobile phone, and a third that uses broadcast messages. The performances of the synchronization algorithms have been evaluated in wireless personal area networks using Bluetooth and ZigBee/IEEE 802.15.4. A real-time implementation of single node synchronization from the mobile phone has been presented, implemented and tested. In addition, a new technique to compensate for clock drift to preserve accuracy in data fusion has been developed, implemented and tested.

Contribution: The author have implemented and evaluated the performance of the presented time synchronization algorithms in the paper. The author has also modified the single clock algorithm to support real-time synchronization. The paper was written in cooperation with the co-authors.

References

[1] Interconnecting Smart Objects with IP - The Next Internet, Jean-Philippe Vasseur and Adam Dunkels, Morgan Kaufmann, 2010.

[2] Baca A, Dabnichki P, Heller M, Kornfeind P. Ubiquitous computing in sports: A review and analysis. J Sports Sci. 2009 Oct; 27(12):1335-46.

[3] Bonato, P. Advances in wearable technology and applications in physical medicine and rehabilitation. Journal of NeuroEngineering and Rehabilitation, 2 (2).

[4] Knight, J. F., Bristow, H. W., Anastopoulou, S., Baber, C., Schwirtz, A., and Arvanitis, T. N. 2007. Uses of accelerometer data collected from a wearable system. Personal Ubiquitous Comput. 11, 2 (Jan. 2007), 117-132.

[5] J. Wåhslén, T. Lindh, M. Eriksson and I. Orhan, "A Novel Approach to Multi-Sensor Data Synchronization Using Mobile Phones", IJAACS, Vol. 6, No. 3. pp 289-303.

[6] D. Sturm, K. Yousaf and M. Eriksson, "A wireless, Unobtrusive Kayak Sensor Network Enabling Feedback Solutions", (BSN 2010), Singapore, June 7 - 9, 2010.

[7] J. Wåhslén, I. Orhan and T. Lindh, "Local Time Synchronization in Bluetooth for Data Fusion Using Mobile Phones", International Conference on Body Sensor Networks (BSN 2011), Dallas, May, 2011.

[8] Elson, J and Römer, K. Wireless Sensor Networks: A New Regime for Time Synchronization. ACM SIGCOMM Computer.

[9] N. Kaempchen and K. C. J. Dietmayer, "Data synchronization strategies for multi-sensor fusion," in Proceedings of ITS 2003, 10th World Congress on Intelligent Transportation Systems.

[10] L. Diduch, A. Fillinger, I. Hamchi, M. Hoarau, V. Stanford, Synchronization of data streams in distributed realtime multimodal signal processing environments using commodity hardware, ICEM 08.

[11] M. Jadliwala, M.,Duan, Q.,Upadhaya, S., Xu and X, J. Towards a Theory for Securing Time Synchronization in Wirless Sensor Networks. WiSec '09.

[12] Ringwald, M. and Römer, K. Practical Time Synchronization for Bluetooth Scatternets. BROADNETS ´07.

[13] Ashraf, I., Gkelias, A., Dohler, M., and Aghvami, A. Time-synchronised multi-piconet Bluetooth environments. IEEE Communications 153(3) 445-452.

[14] K. Römer, P. Blum and L. Meier, "Time Synchronization and Calibration in Wireless Sensor Networks", in Ivan Stojmenovic (Ed.): Handbook of Sensor Networks: Algorithms and Architectures, pp. 199-237, September, 2005.

[15] J. Wåhslén, I. Orhan, D. Sturm and T. Lindh, "Performance evaluation of time syncronization clock drift compensation in wireless personal area network", 7th International Conference on Body Area Network, Olso, September, 2012

[16] G. Coulouris, J. Dollimore and T. Kindberg, "Distributed Systems: Concepts and Design", Chapllpleter 11, 4th edition, Addison-Wesley, 2006.

[17] D. Mills, J. Martin, J. Burbank and W. Kasch, "Network Time Protocol Version 4: Protocol and Algorithms Specification", RFC 5905, Internet Engineering Task Force (IETF), June, 2010.

[18] Frontline, http://www.fte.com

[19] Shimmer Research, http://www.shimmer-research.com

[20] Spark Fun, http://www.sparkfun.com

[21] F. Cristian, "Probabilistic clock synchronization", Distributed computing (1989): pp. 146-158.

[22] J. Wåhslén and D. Sturm, "Executive Summary of AugmentIT", Third place in KTH Health Technology idea competion

[23] Apple's GameKIT API – http://developer.apple.com/

List of papers and journals

Paper I

J. Wåhslén, T. Lindh, M. Eriksson, and I. Orhan, “A Novel approach to multi-sensor data synchronisation using mobile phones“, International Journal of Autonomous and Adaptive Communications Systems, Int. J. Autonomous and Adaptive Communications Systems, Vol. 6, No. 3, pp 289-303, 2012

Paper II

J. Wåhslén, I. Orhan and T. Lindh, "Local Time Synchronization in Bluetooth Piconets for Data Fusion Using Mobile Phones", BSN 2011, May 2011, Dallas.

Paper III

J. Wåhslén, I Orhan, D. Sturm and T. Lindh, ”Performance Evaluation of Time Synchronization and Clock Drift Compensation in Wireless Personal Area Networks”, 7th International Conference on Body Area Networks, September 2012, Oslo, Norway.

Other Events

J. Wåhslén and D. Sturm, ”Executive Summary of AugmentIT”, Third place in KTH Health Tehnology idea competition.

Other scientific contributions not included in this thesis

Paper IV: A novel approach to multi-sensor data synchronization using mobile phones

J. Wåhslén, T. Lindh and and M. Eriksson, 5th International Conference on Body Area Network, September 2010, Corfu, Greece