Wireless Sensor Data Collection based on ZigBee Communication

Jim Torresen and Eirik Renton
Department of Informatics
P.O. Box 1080 Blindern
N-0316 Oslo, Norway
jimtoer@ifi.uio.no

Alexander Refsum Jensenius
Department of Musicology
University of Oslo
Pb 1017, Blindern, 0315 Oslo, Norway
a.r.jensenius@imv.uio.no

ABSTRACT

This paper presents a comparison of different configurations of a wireless sensor system for capturing human motion. The systems consist of sensor elements which wirelessly transfers motion data to a receiver element. The sensor elements consist of a microcontroller, accelerometer(s) and a radio transceiver. The receiver element consists of a radio receiver connected through a microcontroller to a computer for real time sound synthesis. The wireless transmission between the sensor elements and the receiver element is based on the low rate IEEE 802.15.4/ZigBee standard.

A configuration with several accelerometers connected by wire to a wireless sensor element is compared to using multiple wireless sensor elements with only one accelerometer in each. The study shows that it would be feasable to connect 5-6 accelerometers in the given setups.

Sensor data processing can be done in either the receiver element or in the sensor element. For various reasons it can be reasonable to implement some sensor data processing in the sensor element. The paper also looks at how much time that typically would be needed for a simple pre-processing task.

Keywords

wireless communication, ZigBee, microcontroller

1. INTRODUCTION

The use of body movement for controlling interactive systems has increased in popularity over the years, and has also become a standard interaction technique in musical applications. A number of larger research projects have investigated such potentials, e.g. the MEGA project [2], Sound to Sense – Sense to Sound ($S2S^2$) [11], Gesture Controlled Audio Systems (ConGAS) [4], the Sonic Interaction Design project [3], etc. Now we also see an industrial impact of such thoughts, with Nintendo Wii and Apple's iPhone as examples of exploitation of the potential of motion sensing in interactive devices.

There are numerous challenges when it comes to developing such interactive systems, including the development of smaller, faster, cheaper and more precise sensor systems. Another important factor is to reduce the latency in the systems, to ensure that the user of an interactive system gets an immediate response to an action being carried out.

Working with wireless sensor systems adds additional challenges when it comes to handling power consumption, transmission delays, and overall latency in the system. Today the de facto standard for short range wireless transmission is Bluetooth, a technology which is currently embedded in a variety of commercial devices, ranging from computer mice to mobile phones, cameras, printers, etc.

While Bluetooth is also used in a few commercial music instruments, and in some experimental NIMEs, it has not received widespread acclaim in the computer music technology. There are a number of reasons for this, one being the stability of the connection. In our experience, Bluetooth based instruments and controllers are not reliable enough for stage use, due to challenges in connecting and pairing devices, and random dropouts. The latter we have experienced several times during concerts, where setups that worked reliably in rehearsal suddenly started behaving strangely during performance. The most possible explanation for this, we believe, may be interference from other bluetooth enabled devices such as mobile phones in the audience.

A solution to the problem of picking up other devices rather than the instrument could be to decrease the range of the system using Bluetooth Class 2 rather than Class 1. Unfortunately, this will also reduce the range of the musician using a device, and may lead to issues concerning reconnecting and pairing the device with the computer used in performance.

To overcome some of the challenges mentioned above, we have experimented with other types of wireless transmission techniques, and in this paper, we shall focus on the IEEE 802.15.4 ZigBee communication protocol. This is a simpler low rate/low power solution as compared to Bluetooth, but it also has several advantages for our applications. While ZigBee has been used in some other NIME projects, e.g. [1, 5], we have not seen so many systematic studies of its suitability for musical applications.

Our main interest has been to explore the use of accelerometer based systems, and our target is to determine how many accelerometers that are feasible to connect using a single receiver element. That is, we test a set of possible accelerometer configurations to determine how many units can be connected given the requirements of an acceptable response time.

Synchronization and limiting transmission delays are essential in order to get useful data. In our work [**?**], we have defined a maximum delay from motion to extracted motion data being available for the receiver application to be less than 10 ms [12]. In the experiments, we have also tested whether local processing in the sensor element would improve the overall performance of the system as compared to process everything in the computer. For testing, motion data has been received and processed in Max/MSP. In this paper we will not discuss the additional latency added in the sound synthesis and sound card.

The next section outlines the applied technology followed by the implemented configurations in section 3. Results are included in section 4, and section 5 concludes the paper.

2. IMPLEMENTATION TECHNOLOGY

2.1 ZigBee

ZigBee is a communication protocol for a low rate wireless personal area network (LR-WPAN) based on the IEEE 802.15.4 standard. It is designed for low rate applications where long battery life is essential, and its limited weight and size make it beneficial for on-body applications.

The ZigBee protocol has a raw transfer rate around 250 Kbps and 16 channels in the 2,4 GHz band [8], with the protocol headers and tails included. To be able to fully take advantage of this data rate, the user data payload has to contain as much data as possible. If the user data payload only contains a small number of bytes, the transfer rate will decrease accordingly.

The IEEE 802.15.4 standard only exploits the PHY layer and the MAC layer (Data Links sub layer) which are the two lowest layers of the OSI model (Open System Interconnection Reference Model). These layers define the topologies and physical characteristics for low power devices operating in a typical space of 10 m. Maximum transmission range is possible between 10 and 75 meters, and up to 1500 meters for ZigBee pro, but this is heavily dependent on the particular environment. See Table 1 for a comparison of ZigBee and Bluetooth.

Table 1: Comparison of ZigBee and Bluetooth [7]

	Bluetooth	ZigBee
Band	2.4 GHz	2.4 GHz, 868/915 MHz
Power	100 mW	30 mW
Battery life	Days – months	6 months – 2 years
Range	10–30 m	10–75 m
Data rate	1–3 Mbps	25–250 Kbps
Network	Ad hoc, P2P, star	Mesh, ad hoc, star
Security	128-bit encryption	128-bit encryption
Wake and transmit	3 s	15 ms

2.2 Accelerometers

In our setups we have used the ADXL330 from Analog Devices, a low power 3-axis accelerometer measuring acceleration within a range of ± 3 g. It outputs 3 analog voltages, one for each axis, and runs at 180 μA at 1.8 V.

2.3 Microcontrollers

The microcontroller used in our study was the Atmel AVR. This microcontroller is based on a modified Harvard 8-bit RISC architecture, which means that the program memory and the data memory are on separate busses. The AVR microcontrollers were one of the first using FLASH memory for on-chip data storage. All AVR microcontrollers have a 16 bit flash memory that can store between 1 kB to 256 kB. The integrated flash memory can be programmed with an in-system programmer (ISP), a JTAG programmer or a high voltage parallel/serial programmer. Atmel also offers development platforms such as AVR-studio for writing, programming and debugging AVR applications. This enables the user to write instructions and application in languages like C or assembly.

The implementation in this paper is based on the 64 pin ATmega1281v. This chip contains eight single ended ADC channels, which means that it can only measure two 3D accelerometers. For later studies we plan to use the 100 pin ATmega1281v with 16 single ended ADC channels, so that it is possible to connect up to five 3D accelerometers. Since these two microcontrollers have the same characteristic, we will use five accelerometers in the latency calculations.

3. IMPLEMENTED CONFIGURATIONS

This section outlines the different wireless motion capture system configurations tested. The receiver element is equal for the two setups and only *one* receiver element/microcontroller is used for all the configurations. Serial RS232 transmission was used inside the receiver since that was the only communication interface available in our system. Such slow communication is not a desirable solution compared to other communication protocols with higher transfer rates (e.g. USB), so we also include numbers on possible reduced latency by improved communication in the results section.

3.1 Multiple Accelerometers Connected to a Single Transmitter

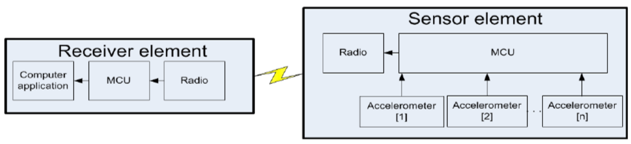

Figure 1: Multiple accelerometers connected to a single transmitter.

In this setup multiple accelerometers are connected to a microcontroller, which transmits wirelessly to a receiver connected to a computer (Figure 1). When a sensor element reads multiple sensors, wires are needed. A wired solution may not be ideal from the end user's point of view, but this will have to be balanced with the potential for a faster and more reliable motion capture solution. Further, such a wired setup can be expanded into a motion capture system where multiple microcontroller sensor elements collect movements on different parts of the body.

3.2 Single Accelerometer Connected to Each Transmitter

In this configuration, only one accelerometer is connected to each microcontroller, and each microcontroller transfers individual data streams to a receiver (Figure 2). Here each sensor element contains one accelerometer, one microcontroller and one radio transceiver. The receiver element consists of one radio transceiver and will receive data from all sensor elements. This is a *star topology* where multiple elements directly communicate with a master receiver node. Figure 3 shows an implementation of this topology where 6 sensor elements are connected to a person, and these sensor elements communicate with a receiver element.

3.3 Latency sources

Latency sources within the complete sensor system are shown in Figure 4 and include the following:

- **ADC conversion time:** The ADC conversion time will increase when adding accelerometers. This is mainly an issue for the motion capture systems in section 3.1, since the sensor element's microcontroller has to read multiple accelerometers. This will result in a larger

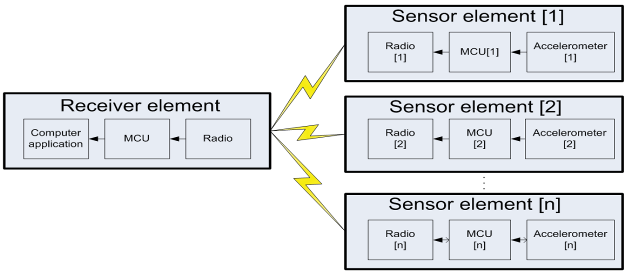

Figure 2: Single accelerometer connected to a single transmitter.

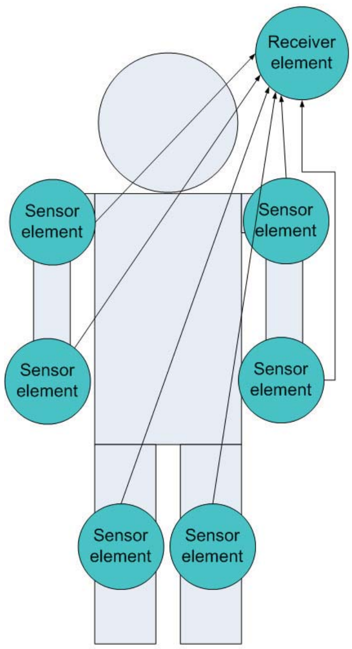

Figure 3: Multiple transmitters each containing one accelerometer.

ADC conversion latency compared to the star based setup (section 3.2) where each sensor element only measures one accelerometer.

- **Sensor data processing:** A microcontroller can be used to filter, process and extract important information from the sensor data. In cases where the data transmission is the main latency source, a sensor element can do additional processing while it is waiting for its time slot on the wireless network. This kind of processing will increase the hardware power consumption but will cause less processing in the computer application. In a motion capture system with many nodes, it will be preferable if each node minimizes their data streams.

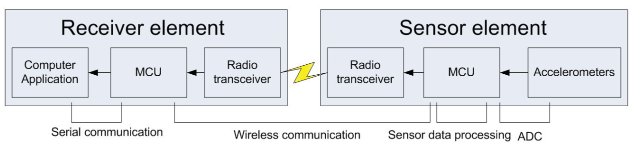

Figure 4: Contribution to latency in the system.

- **Latency sources between the sensor element and the receiver element:** The ZigBee protocol has a 250 Kbs raw transfer rate. This includes frame headers and tails, and will be reduced if the frames are not filled with the maximum amount of user data.

- **Latency sources within the receiver element:** Serial communication between the microcontroller and the computer application is based on an RS232 connection. This communication has a transfer rate that will be limited by both the computer application and the microcontroller.

We have conducted different latency tests on the above mentioned implementations, using a transfer rate measurement algorithm implemented in the receiver element. This algorithm counts the received user data payloads from the two sensor elements and calculates the transfer rate based on the amount of received user data payload per second. The latencies caused by the ADC are estimates based on the microcontroller's datasheet. The serial communication latencies are measured with a Max/MSP patch counting received frames from the serial object. Results from measurements and estimates are shown in section 4.

3.4 Data Processing Within the Transmitter Microcontroller

In addition to transferring data, the microcontroller can perform different filter algorithms and/or analyses on the sensor data. This would demand additional time spent on processing but could also reduce the amount of data to be transmitted. To test this, some simple filters (median, mean and high pass) were implemented in the transmitter elements, and were used to estimate computation time.

4. RESULTS

Table 2 shows the total latencies for the setup with multiple accelerometers connected to a single transmitter when transmitting 10-bit ADC data. The total latency is calculated from the accelerometers into Max/MSP. This shows how the serial SLIP transmission (at 76800 bps) in the receiver will influence the latency and limit the applicability for music applications as the number of accelerometers is increased.

Table 2: Latency results from multiple accelerometers connected to a single transmitter

Accelerom.	ADC	ZigBee	Serial	Total
1 (3 bytes)	312 µs	3.04 ms	1.89 ms	5.24 ms
2 (6 bytes)	624 µs	3.20 ms	3.64 ms	7.46 ms
3 (9 bytes)	936 µs	3.42 ms	5.4 ms (est.)	9.76 ms
4 (12 bytes)	1.25 ms	3.62 ms	7.14 ms (est.)	12.01 ms
5 (15 bytes)	1.56 ms	3.84 ms	8.8 ms (est.)	14.2 ms

However, using a more effective communication in the receiver we could probably be able to operate five accelerometers within the latency requirement ($\sim$ 10 ms). A larger number would not be feasible due to the number of available ADC channels in the microcontroller.

Table 3: Latency results from a single accelerometer connected to each transmitter

Accelerom.	ADC	ZigBee	Serial	Total
1	312 µs	2.73 ms	1.89 ms	4.93 ms
2	312 µs	5.23 ms	1.89 ms	7.43 ms
3	312 µs	8.18 ms (est.)	1.89 ms	10.38 ms
4	312 µs	10.03 ms (est.)	1.89 ms	12.23 ms

Table 3 shows the timing for a single accelerometer connected to each transmitter. This includes the ADC conversion, ZigBee transmission and serial SLIP transmission (at 76800 bps) in the receiver. In this case three sensor elements will be slightly above our requirement of a 10 ms latency.

As seen in Table 3, the ZigBee communication time increases almost linearly with the number of sensor elements in use. This communication time can be improved by introducing a more effective protocol [6]. Our estimations indicate that it should be possible to receive data from up to 6 sensor elements (within 10 ms) using such an optimised communication protocol [10].

4.1 Data Processing Within the Transmitter

Table 4 shows the time needed for median calculations, including the time taken for the ADC measurements and the program execution time inside the microcontroller. The latter is based on a program counter feature in AVRstudio, and are relatively small compared to the ADC conversion latencies. Therefore, such operations should be feasible to include in the transmitter units without substantially exceeding the latency requirement.

Table 4: Time needed for median calculations

Window size	f_{osc}	ADC	Processing	Total
3	4 MHz	936 µs	295.50 µs	1.23 ms
	8 MHz	936 µs	147.75 µs	1.08 ms
	16 MHz	936 µs	73.86 µs	1.01 ms
5	4 MHz	1.56 ms	749.50 µs	2.31 ms
	8 MHz	1.56 ms	374.75 µs	1.93 ms
	16 MHz	1.56 ms	187.38 µs	1.75 ms

5. CONCLUSION

The paper has presented the development and evaluation of a wireless motion capture solution for musical applications. The aim was to evaluate the possibilities of developing a solution where the total latency from sensor to received data on a computer is below 10 ms. This is what we regard as the maximum latency that would still work in the musical applications in which we want to use the system.

Two wireless motion capture systems based on accelerometers, microcontrollers and ZigBee transceivers have been tested. The first setup was based on multiple 3D accelerometers connected to a single transmitter. This system allowed for up to three accelerometers to be connected and transmit within 10 ms. Using a better communication protocol, we have estimated that it should be possible to connect five accelerometers and still be within the required latency. Concerning scalability, this setup is limited by the available number of ADC channels on the microcontroller. When it comes to other limitations, this setup requires cabling between the accelerometers and the microcontroller, which is not an ideal solution when it comes to portability and usability.

The second tested setup was based on connecting single 3D accelerometer to each transmitter. Here we also found that three accelerometers (with three separate transmitters) could be read within the latency limit. We believe that it should be possible to optimize this to six sensor elements using more efficient communication.

Future work consists of further improving our system and developing different prototypes for testing the ideas in musical practice.

6. ACKNOWLEDGMENTS

The research is funded by the Research Council of Norway through the project *Sensing Music-related Actions* (proj. no. 183180).[1]

7. REFERENCES

[1] F. Bevilacqua, F. Guédy, N. Schnell, E. Fléty, and N. Leroy. Wireless sensor interface and gesture-follower for music pedagogy. In *NIME '07: Proceedings of the 2007 International Conference on New Interfaces for Musical Expression*, New York, 2007.

[2] A. Camurri, G. De Poli, A. Friberg, M. Leman, and G. Volpe. The MEGA project: Analysis and synthesis of multisensory expressive gesture in performing art applications. *Journal of New Music Research*, 34(1):5–21, 2005.

[3] A. Ferreira. Memorandum of Understanding (MoU) for the implementation of a European Concerted Research Action designated as COST Action IC0601: Sonic Interaction Design. Technical report, EU COST, 2007.

[4] R. I. Godøy and M. Leman. *Musical Gestures: Sound, Movement, and Meaning.* Routledge, New York, 2010.

[5] R. Jacobs, M. Feldmeier, and J. Paradiso. A Mobile Music Environment Using a PD Compiler and Wireless Sensors. In *Proc. of the 8'th International Conf. on New Interfaces for Musical Expression (NIME 2008)*, 2008.

[6] Jennic. Application note: JN-AN-1035, calculating 802.15.4 data rates. In *www.jennic.com/download_file.php?supportFile=JN-AN-1035%20Calculating%20802-15-4%20Data%20Rates-1v0.pdf, 2006.*

[7] J. Kooker. Bluetooth, zigbee, and wibree: A comparison of wpan technologies. In *http://www.cse.ucsd.edu/classes/fa08/cse237a/topicresearch/*, 2008.

[8] J. Lee. Performance evaluation of IEEE802.15.4 for low-rate wireless personal area networks. *IEEE Transactions on Consumer Electronics*, 52(3):742–749, 2006.

[9] E. Renton. *Collecting Sensor Data for Active Music.* Master thesis, University of Oslo, http://www.duo.uio.no/publ/informatikk/2009/92473/Renton.pdf, 2009.

[10] E. Renton. Collecting sensor data for active music. Master's thesis, University of Oslo, 2009.

[11] X. Serra, M. Leman, and G. Widmer. A roadmap for sound and music computing. Technical report, Sound to Sense, Sense to Sound, 2007.

[12] D. L. Wessel and M. Wright. Problems and prospects for intimate musical control of computers. In *NIME '01: Proceedings of the 2001 International Conference on New Interfaces for Musical Expression*, Seattle, WA, 2001. New York: ACM Press.

[1] http://www.fourms.uio.no/projects/sma