The Chinese University of Hong Kong, HKSAR, China. Jun 1-3, 2008

Compressing Inertial Motion Data in Wireless Sensing Systems – An Initial Experiment

Lawrence Cheng, Stephen Hailes, Zhen Cheng, Fu-Yi Fan, Denis Hang, Yang Yang, *Member, IEEE*

***Abstract*—The use of wireless inertial motion sensors, such as accelerometers, for supporting medical care and sport's training, has been under investigation in recent years. As the number of sensors (or their sampling rates) increases, compressing data at source(s) (i.e. at the sensors), i.e. reducing the quantity of data that needs to be transmitted between the on-body sensors and the remote repository, would be essential especially in a bandwidth-limited wireless environment. This paper presents a set of compression experiment results on a set of inertial motion data collected during running exercises. As a starting point, we selected a set of common compression algorithms to experiment with. Our results show that, conventional lossy compression algorithms would achieve a desirable compression ratio with an acceptable time delay. The results also show that the quality of the decompressed data is within acceptable range.**

I. Introduction

In recent years, (human) body motion sensing for supporting sport's training [1][2][8], medical care [3][4], computer graphic generation [5][6], and more, has been under investigation. Body motion sensing can be carried out in two distinctive ways: using either optical motion sensing systems [7] or Inertial Motion Sensing (IMS) systems. In IMS systems, in which multiple, lightweight inertial motion sensors [14], such as accelerometers, are attached to different parts of a (human) body. These devices capture (different types of) motion data of different segments of a moving body, and deliver the collected data (usually through wireless channels) from the sensors to a remote repository, at which data analysis and (long-term) data storage take place. As the number of on-body sensors (or their sampling rate) increases, the size of the collected motion data will increase proportionally, and will eventually become too large for efficient storage and/or for (wireless) transmission [12]. Using a commercially available system, the MTx sensor from xSens, as a reference [11]: the data rate of each sensor can reach 240Kbps[1]. Compressing the data, either at the source(s) (i.e. at the sensors in IMS systems) or at the repository, therefore becomes important. In IMS systems, compression reduces the quantity of data that needs to be transmitted wirelessly between the sensors and remote repository, which is important for bandwidth-limited wireless environments.

Compression of optical motion data has been investigated in previous research. In [9], a data-driven compression algorithm designed to compress motion data captured by optical sensing systems was described. In [10], another dedicatedly designed compression algorithm was described. However, compression of inertial motion data in IMS systems (which involves different data types from the displacement data of optical sensing systems) is yet to be explored. In this paper, we shall present a set of experiment results on inertial data compression, using the acceleration data that we have collected during several running sessions. The reason for using running data is because these are the most rapidly changing data of human motion, which contains more noise[2]; which we believe are more interesting than conventional walking steps data.

II. Background

A. Inertial Motion Sensor Prototype

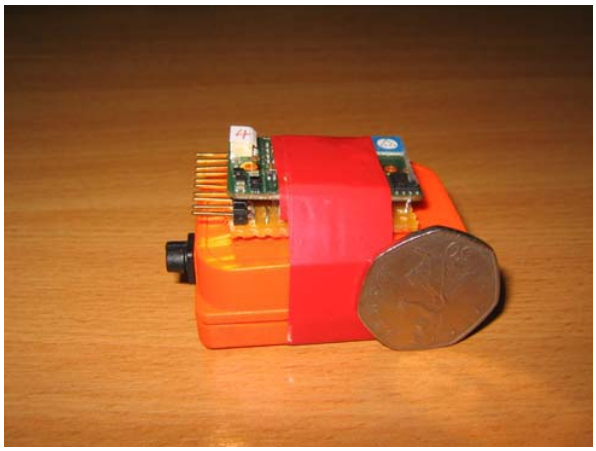

Figure 1 – A battery-powered MTx sensor connected to a connectBlue WiFi module via RS232

We have developed a wireless inertial sensor to collect motion data; the sensing unit contains an MTx sensor that is connected via a RS232 interface to an 802.11 wireless interface (a connectBlue OWSPA311g module) (Figure 1). Figure 2 shows the changes in acceleration data output from our sensor. Note that acceleration data oscillates rapidly over time.

[1] The maximum sampling rate of a MTx is 500Hz; each sample is 60 bytes. Thus a data rate of 240Kbps per sensor. Multiple sensors are needed per subject because each sensor can only sense one segment of the body.

[2] Running movements are more rapid and harder to detect (due to much shorter moments of impact with ground); running data also contain more noise due to the small frictional movements among the contacting surfaces of the sensor and the subject [15].

978-1-4244-2253-1/08/$25 ©2008 IEEE

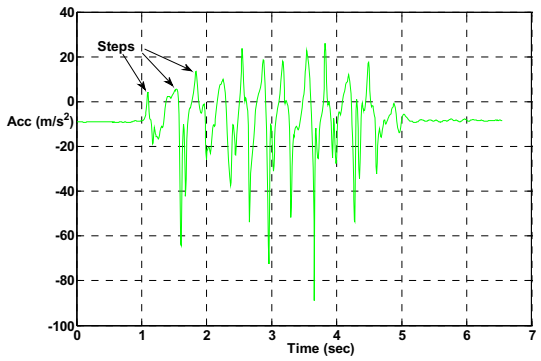

Figure 2 – Changes in vertical acceleration of running data over time

B. *Existing Compression Algorithm Evaluation Criteria*

In lossless compression, the compression ratio and block size are the common comparison parameters. There are two ways to determine the accuracy of uncompressed data when lossy compression algorithms are used: a generic approach would use mathematical methods such as Mean Square Error (MSE); alternatively, the application (e.g. such as a body motion kinematics analysis) can be executed on the original (uncompressed) data and the decompressed data respectively. In this paper, we used the generic approach because it does not require understanding of body kinematics analysis (which is out-of-scope of this paper). Another application-specific factor is the time delay: this depends on how long an application can tolerate between the time of data collection and result computations; this value is user/application-specific, and depends on the time allowance between when an event happens and when the data is needed by the user. For example, the time delay toleratable for a long-term body motion medical analysis would be much longer than the time delay acceptable for a real-time 100m-run monitoring system (which would be ~10 seconds)[3].

III. Common Compression Schemes

A. *Assumptions and Experiment Setup*

Our investigation focuses on the compression ratio and the accuracy of the decompressed data: power consumption is out of scope of this paper. We assume that a reliable wireless transport protocol, such as TCP, is used to deliver the compressed data to a remote repository for decompression. This assumption is made because there is a need to separate the cause of lost in accuracy due to the lossy compression/decompression process, and the lost in accuracy due to network (i.e. packet) loss (see future work). The wireless sensor is placed on the back of the subject for 100m runs. The compression algorithm is run on a laptop with an Intel Core 2 CPU 1.83GHz and 1GB of RAM.

B. *Results and Analysis*

Figure 3 and Figure 4 show the results when compressing different number of blocks of acceleration data[4]. Both lossless and lossy algorithms were used: bzip2, zlib, LZW, and lossy zlib. We chose these algorithms as our starting point of investigation because they are among the most popular candidates for data compression.

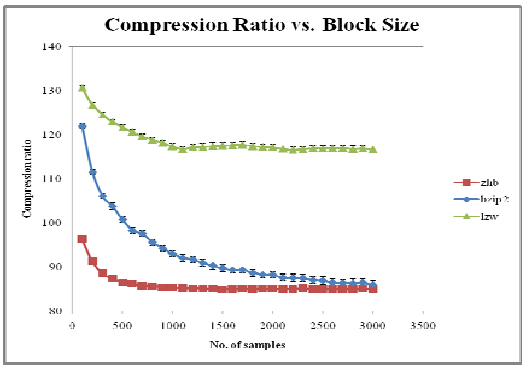

Figure 3 – Compression ratio Vs block size (lossless compression)

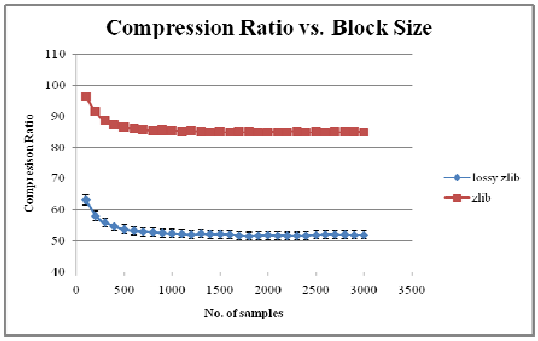

Figure 4 – Compression ratio Vs block size (lossy compression)

The results in Figure 3 and Figure 4 show that the (average) compression ratio improves when more data are being compressed (i.e. using a larger block size). In Figure 3, the best compression ratio was achieved when compressing 500-1200 samples in one block using (lossless) zlib. This would yield a time delay between 4.2 to 10 seconds (at a sampling rate of 120Hz). However, the compression ratio is far from optimal (~87%). Furthermore, some algorithms, such as LZW, generates worst-off compression ratio when compressing data in small blocks. This is because LZW is a dictionary-based algorithm that is designed to encode new data based on previously encountered data. However, in small blocks of rapidly changing acceleration data (Figure 2), such feature cannot be explored. A better compression ratio result is achieved when lossy zlib was used. A compression ratio ~53% was archived when 800 samples are compressed at once. This yields a time delay of ~6.67 seconds. We calculated the MSE value when lossy zlib was used was ~0.0048 (±0.001); thus, the errors are marginal. However, because the time delay is less than optimal (i.e. optimal compression ratio is only achieved when compressing larger blocks of data), other compression algorithms should be investigated.

[3] Our interviews with coaches suggest that they would rather to observe the athlete's movement directly whilst they are running. However, the coaches would like to visualise the computed results, such as stride length, stride frequencies, etc., as soon as the stride is finished. Thus, for a 100m run, a maximum time buffer of 10 sec can therefore be assumed.

[4] We used the default sampling rate of the MTx unit for these experiments i.e. 120Hz (120 samples per second).

IV. Data Compression Using Wavelets

A. Algorithm Design

Figure 5 – Block diagram of the wavelet algorithm

In this section, we present how the compression ratio would be improved should wavelet transforms are used [13]. We chose wavelets as our compression algorithm for the following experiments because they are a standard form of compression algorithm. The wavelet algorithm we used had a series of filters and methods (Figure 5). The signals are low-pass filtered by a Butterworth filter of order 4 (with a cut-off frequency of 20Hz) in order to reduce noise. The resultant data set is then subject to different wavelet transforms respectively (i.e. Haar, Linear Interpolation, Daubechies D4) to create different sets of coefficients. The number of coefficient is equivalent to the number of values in the data set, so that a few large(r) values remain to represent the magnitude of the original data; and small(er) values can be stored with fewer bytes. The last step is compression, by using the different sets of coefficients derived from the previous step.

B. Results & Analysis

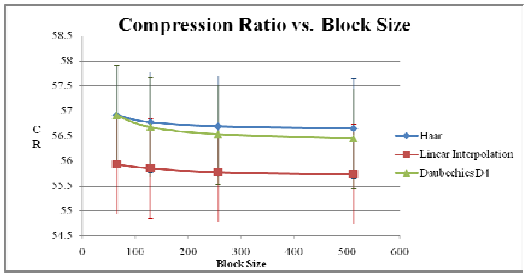

Figure 6 – Compression ratio Vs block size

Figure 6 shows the average compression ratios when running acceleration data are compressed using different wavelet transformations. The results are more optimal than the ones when compressing samples using lossless algorithms (note that the maximum number of blocks in Figure 6 is different from Figure 4). The differences between different transform algorithms are not obvious. In addition, the block size has little impact on the compression ratio. This gives the wavelets approach an advantage over the lossless approach we presented in the last section in term of a smaller time delay. The MSE in the signals are within acceptable range: 0.036 (±0.001), 0.0375 (±0.001), 0.037 (±0.001) respectively for the Haar, Linear Interpolation, Daubechies D4 transform respectively.

The reasons why wavelets generate a better compression ratio is that they handle this type of data better: note that, acceleration data oscillates over time (i.e. similar to a sin/cosine wave pattern when plotted against time, Figure 2). This suggests that, the relationship between adjacent samples (of the same data type) should be – to some extend - predictable[5]. Wavelet transforms fit into our scenario better because they assume finite-length, oscillating waveforms: body motion, such as ground contact of the foot, generates finite waves that propagate through the body (i.e. movements can only be measured within a finite amount of time). In addition, wavelets handle sharp discontinuities (i.e. sharp peaks), such as sudden movement or sudden stoppage (such as when the toe touches ground, acceleration and gyros readings change rapidly, see Figure 2), better than Fourier transform.

V. Data Compression Using ADPCM

A. Algorithm Design

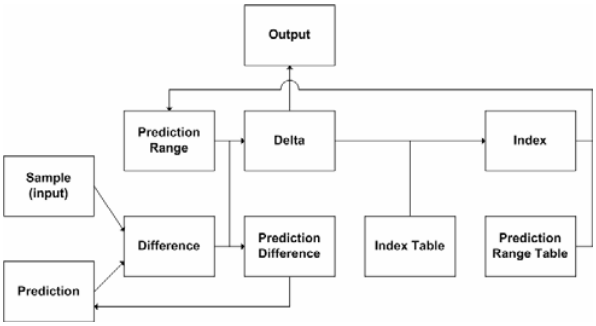

Figure 7 – Encoder block schematic

Delta	Prediction Difference	Condition
0	PredictRange/8	0<Difference<PredictRange/4
1	(3/8)*PredictRange	PredictRange/4<=Difference<(1/2)*PredictRange
2	(5/8)*PredictRange	(1/2)*PredictRange<=Difference<(3/4)*PredictRange

Table 1 – Prediction difference and delta

From Figure 2, some types of motions – such as acceleration data - oscillates over time (i.e. repeated movements). This feature is similar to speech waves. Thus, we chose Adaptive Differential Pulse Code Modulation (ADPCM) as our next compression algorithm to elaborate further this feature [6]. To improve the efficiency of ADPCM, we implemented a simplified version of ADPCM that is based on the G.721 standard. 4 bits are used to code one (acceleration) sample. One sample is used to predict the latter one. Figure 7 shows the encoder block schematic: the input is a sample value; the difference is the sample value minuses prediction; the prediction range is the range within which current prediction can change. Table 1 shows a shortlist of the methods to determine the prediction difference and delta.

B. Results and Analysis

Table 2 shows the results when compressing different number of acceleration samples using ADPCM with error control under 1%. For completeness, two different sets of data were used: running data and sprinting data (i.e. the latter involves more rapid movements). The results show that the compression ratio is more optimal (i.e. <45% for the acceleration data). The quantity of samples being compressed at once has little impact on the results; this would be advantageous for real-time (or near-real-time)

[5] Consider the acceleration of the foot: the same (or very similar) movement would be repeated recursively throughout the run; thus, the acceleration data should be to a certain extend, predictable (for each cycle).

[6] Another candidate is Differential PCM (DPCM), which encodes the differences between adjacent samples. ADPCM is used in this case because, as we have explained in an earlier section, we intend to determine whether the data is predictable [13].

applications. We have also calculated the corresponding MSE, which is within acceptable range (i.e. <0.003 (±0.001).

# of samples	Running Data Set			Sprinting Data Set		
	Min	Max	Average	Min	Max	Average
50	20.5%	100%	44.9%	20.5%	100%	47.6 %
100	18.5%	97.2%	42.8%	18.5%	100%	45.7 %
200	17.5%	92.7%	41.3%	17.0%	100%	44.1 %

Table 2 – ADPCM compression ratio on acc. data (with error control <1%)

VI. Conclusion & Future Work

On-body wireless inertial sensing for supporting medical care and sports training have been investigated in recent years. As the number of sensors increase, there is a need to identify a suitable compression approach in order to compress data at source(s) or at the repository in order to reduce overheads in (wireless) transmission and/or data storage. This is particularly important in bandwidth limited environments where multiple sensors are deployed in close physical proximity. In this paper, we have presented a set of compression experiment results on the inertial motion data that we have collected using our wireless MTx sensor. Our observations are that, existing lossy compression algorithms can be used to compress inertial data with optimal compression ratio and acceptable errors. We have presented in this paper a set of initial experiment results on inertial motion data compression, we believe our work provides some insights for designing more efficient IMS systems through compression.

As part of our future work, we intend to investigate the effect of packet lost in the quality of the decompressed data when an unreliable transmission protocol, such as UDP, is used. Should the results are still within an acceptable range, using UDP would generate less transmission overhead (compared to TCP), which would be advantageous in a wireless environment and in real-time systems. However, should the results are not optimal, applying a suitable Forward Error Correction (FEC) scheme would be a potential solution.

Acknowledgements

This paper describes work undertaken in the context of the EPSRC-funded SEnsing for Sport And Managed Exercise (SESAME) project (EP/D076943).

References

[1] The SEnsing for Sport And Managed Exercise (SESAME) project, http://www.sesame.ucl.ac.uk

[2] S. Armstrong, "Wireless Connectivity for Health and Sports Monitoring: a review", in Proceedings of the British Journal of Sports Medicine 2007, pp. 285-289.

[3] K. Lorincz, D. Malan, T. Fulford-Jones, A. Nawoj, A. Clavel, V. Shnayder, G. Mainland, S. Moulton, M. Welsh, "Sensor Networks for Emergency Reponse: Challeneges and Opportunities", in Special Issue on Pervasive Computing for First Response, IEEE Pervasice Computing, Oct-Dec 2004.

[4] T. Tamura, "Wearable Accelerometer in Clinical Use", in Proceedings of the 27th IEEE Internal Conference of the Engineering in Medicine and Biology Society (EMBS), Shanghai, China, Sep 2005, pp. 7165-7166.

[5] K. Grochow, S. Martin, A. Hertzmann, Z. Popovic, "Style-based Inverse Kinematics", in Proceedings of ACM Transactions on Graphics (SIGGRAPH), August 2004, pp. 522-531.

[6] A. safonova, J. Hodgins, N. Pollard, "Synthesizing Physically Realistic Human Motion in Low-dimensional, Behaviour-specific Spaces", in Proceedings of ACM Transactions on Graphics (SIGGRAPH), August 2004, pp. 514-521.

[7] The Qualisys Motion Capture Systems, http://www.qualisys.com/

[8] L. Cheng, S. Hailes, "An Experimental Study on a Motion Sensing System for Sports Training", short paper in the Proceedings of the 5th European Confernece on Wireless Sensor Networks (EWSN), Bologna, Italy, Feb 2008.

[9] G. Liu, L. McMillan, "Compression of Human Motion Data Sequences", in Proceedings of the 3rd International Symposium on 3D Data Processing, Visualsation, and Transmission (3DPVT), 2006, pp. 248-255.

[10] H. Ye, J. Gong, "Motion Data Management of 3D Moving Objects", in Proceedings of IEEE Geoscience and Remote Sensing Symposium (IGARSS), July 2003, pp. 3736-3738.

[11] The MTx System, xSens Motion Technologies, http://www.xsens.com

[12] N. Kimura, S. Latifi, "A Survey on Data Compression in Wireless Sensor Networks", in Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC), April 2005, pp. 8-13.

[13] S. Nalatwad, M. Devetsikiotis, "A Framework for Adaptive Wavelet Prediction in Self-Sizing Networks", in Proceedings of the 39th Annual Symposium on Simulation (ANSS), 2006, pp. 10-17.

[14] A. Christian, J. Healey, "Gathering Motion Data Using Featherweight Sensors and TCP/IP over 802.15.4", HP Technical Report, HPL-2005-188, Oct 2005.

[15] S. Nakazawa, T. Ishihara, H. Inooka, "Real-time Algorithms for Estimating Jerk Signals from Noisy Acceleration Data", in International Journal of Applied Electromagnetics and Mechanics, Vol. 18 (2003), pp. 149-163.