An FPGA-based Spectral Anomaly Detection System

Duncan J.M. Moss, Zhe Zhang, Nicholas J. Fraser and Philip H.W. Leong
School of Electrical and Information Engineering
Building J03, The University of Sydney, 2006, Australia

***Abstract*—Anomaly detection based on spectral features is applicable to a diverse range of problems including prognostic and health management, vibration analysis, astronomy, biomedical engineering and computational finance. The input data could be regularly sampled, as in the case of a standard analogue to digital converter sampling a bandlimited signal at above the Nyquist rate, or irregularly sampled, as in the case of stock quotes or astronomical data. In this paper, we present new online algorithms for the computation of power spectra for regularly or irregularly sampled data, and performing anomaly detection on time series data. Both algorithms allow hardware implementations with $O(1)$ time complexity, this being the minimum for any system that considers all the samples. We combine the two algorithms to form a power Spectrum-based Anomaly Detector (SAD). We also describe an implementation of SAD which has minimal hardware requirements, and achieves one to two orders of magnitude improvement in speed, latency, power and energy over a traditional processor-based design.**

I. Introduction

Time is of the essence when processing data streams. Efficient processing can mean the difference between good and bad decisions. Real-time data mining and machine learning techniques have long been applied to fields ranging from forecasting financial markets to autonomous vehicles to adaptive processing and machine prognostics. For any application, the premise of machine learning is the same: a computer system will learn to model future outcomes based on previously acquired data.

Although there has been considerable recent progress in addressing how to scale offline systems to match the rapidly increasing quantities of data, real-time learning remains a challenge. New algorithms and computer architectures are needed as current implementations based on general-purpose computing are not able to process data with sufficiently low latency or high throughput. Such systems require *online learning* in which the machine learning model can be updated upon receiving new data without extensive recomputation.

Field Programmable Gate Array (FPGA) technology has a distinct advantage in such applications as it offers a more direct way to implement a given algorithm than interpreting an instruction stream, as done on multicore, cluster and Graphics Processing Unit (GPU) based systems. Higher degrees of parallelism can be achieved through replication of computing units, pipelining and reducing instruction execution overhead. Moreover, FPGAs offer the potential to more tightly integrate machine learning with the lower level data acquisition and/or networking hardware, reducing buffer sizes and better optimising latency.

The power spectrum is often used as a feature in applications including machine prognostics [1], astronomy [2] and computational finance [3]. In some applications, it is either impossible to sample at uniform intervals, e.g. astronomical observations [2], or data points are sampled at irregularly spaced intervals e.g. financial tick data [3]. In this paper, we address the problem of extracting power spectra in a hardware and time efficient manner, for both uniformly sampled and irregularly sampled data.

Anomaly detection is the problem of identifying data which is not in accordance with expected behaviour, and is used in fields such as telecommunications, epidemiology, molecular biology, astronomy, quality control, finance and reliability. Common applications include intrusion detection, credit card fraud detection, medical and public health data analysis, industrial damage detection, image processing, email spam filtering, and sensor networks [4].

In this paper, we present an FPGA-based, unsupervised, online spectral based anomaly detection system for time series data. In order to minimise latency and maximise throughput, the system is designed to update its state in time complexity $O(1)$ when given a new sample, this being the asymptotic minimum complexity of any technique which considers all the input data. The contributions of this work are:

- An online power spectrum computation technique which dramatically reduces latency compared to one using the Discrete Fourier Transform (DFT) or Fast Fourier Transform (FFT), for both regularly and irregularly-sampled time series data.
- An efficient algorithm for anomaly detection on time series data.
- The first integrated power Spectral Anomaly Detection (SAD) system which is optimised for efficient FPGA implementation, and intended for real-time embedded and large-scale streaming data mining applications.
- To the best of our knowledge, this is the first paper demonstrating the feasibility of high-speed SAD applications.

The remainder of the paper is organised as follows. In Section II, previous work in spectral estimation and online anomaly detection is reviewed. In Section III, we propose a new hardware-efficient algorithms for extracting the power spectrum from a multivariate time series, and performing anomaly detection. The FPGA implementation of the algorithm is described in Section IV and results are presented in Section V. Finally, conclusions are given in Section VI.

II. Background

A. Power Spectra of Irregularly Sampled Time Series

The standard tool for calculating power spectral density is the DFT. For an N-point time series $x_n, n = \{0, 1, \ldots, N-1\}$ sampled at uniformly spaced time points, the DFT is defined as:

$$X_k = \sum_{n=0}^{N-1} x_n e^{-j2\pi kn/N} \tag{1}$$

Though computing Eq. (1) for frequency bins $k = \{0, 1, \ldots, N-1\}$ is most common, the frequency resolution can be arbitrarily chosen, particularly useful for the purpose of estimating power spectra. We define a frequency domain resolution of M bins uniformly distributed across the frequency range of $\omega \in (-\pi, \pi)$ where $\omega = 2\pi k/N($ in radians per sample).

As a function of ω, Eq. (1) can be rewritten as:

$$X(\omega) = \sum_{n=0}^{N-1} x_n e^{-j\omega n} \tag{2}$$

The periodogram or normalised power for frequency ω is commonly computed from the squared magnitude of $X(\omega)$:

$$P(\omega) = |\frac{1}{N} X(\omega)|^2 \tag{3}$$

For the generalised univariate time series $x_n = x(t_n)$ with arbitrarily spaced but strictly increasing sampling times t_n, Eq. (2) may be used by replacing the time index n in complex exponent by the samples observation time t_n:

$$X(\omega) = \sum_{n=0}^{N-1} x_n e^{-j\omega t_n} \tag{4}$$

and ω is in radians per unit time t_n. We compute over uniformly spaced frequency bands $\omega = 2\pi j/M$, and $j = \{0, 1, \ldots, (M-1)\}$. The power based on (4) is known as the *classical Fourier periodogram.*

Compared with the classical periodogram, Least-Squares Spectral Analysis (LSSA), otherwise known as the Lomb-Scargle periodogram, generally produces better and more accurate power spectra by including compensation terms which reduce the effect of global aliasing due to non-uniform sampling [2], [5]. However, Scargle also stated that the Lomb-Scargle periodgram typically does not differ significantly from the classical periodgram [2]. Since the full Lomb-Scargle periodogram is approximately 3-4 times more expensive to compute, and its output is subsequently discretised into a small number of symbols, it is safe to assume that the classical periodogram will be a sufficiently good approximation.

To further reduce computational and book keeping costs, an exponentially decaying weight is applied, replacing the fixed-width sliding window.

$$X(\omega) = (1-\gamma) \sum_{n=0}^{N-1} \gamma^{N-1-n} x_n e^{-j\omega t_n} \tag{5}$$

where γ is the exponential weighting factor $\gamma \in (0, 1)$. Given a particular choice of sample half-life λ (in number of samples), γ is related to λ by $\gamma^{\lambda} = \frac{1}{2}$.

As the number of samples N increases towards infinity, the sum of exponential weights will converge to:

$$\lim_{N\to\infty} \sum_{n=0}^{N} \gamma^{N-n} = \frac{log_e \gamma}{\gamma - 1} \int_0^{\infty} \gamma^n dn = \frac{1}{1-\gamma}$$

and hence the normalisation by its inverse $(1-\gamma)$.

B. Anomaly Detection

A number of comprehensive surveys, including those of Chandola et. al [4], Patcha and Park [6], Agyemang et. al. [7], and Hodge and Austin [8], have been published. In this subsection, we review FPGA-based anomaly detection systems.

Perhaps the best known systems in the reconfigurable computing literature is network intrusion detection. This requires operation at network line speeds, where the advantages of FPGAs are clear. Das et. al [9] proposed a system based on feature extraction and Principle Component Analysis (PCA) to identify anomalies. Their architecture could support data at over 20 Gbps. The PCA part of the system was performed in an offline manner.

Carter et. al. [10] has proposed an Exponentially Weighted Moving Average (EWMA) method where an anomaly is detected when the absolute value of the difference between the local mean and the input data sample exceeds the estimated standard deviation times a constant multiplier. This results in an effective and robust method that quickly adapts to distributional data shifts. It has an update time complexity $O(1)$, making it suitable for a low latency, high throughput implementation. Unfortunately, it only considers the simple mean value and cannot track changing sequences of patterns in the data.

III. Algorithms

In this section, we present our reformulations of existing power spectra and anomaly detection algorithms to facilitate efficient FPGA-based SAD implementations.

A. Power Spectra of Irregularly Sampled Time Series

While the FFT is commonly used as an efficient algorithm to compute the DFT, it is not optimal for real-time applications which require low latency and/or fast output update rates. In the case of computing Eq. (5), it is desirable to have the power updated as soon as a new sample is available, with the least amount of computation possible. We observe that the DFT at the time of sample number N with adjusted time reference is:

$$X(\omega, N) = (1-\gamma) \sum_{i=0}^{N} \gamma^{N-i} x_i e^{-j\omega(t_i - t_N)} \tag{6}$$

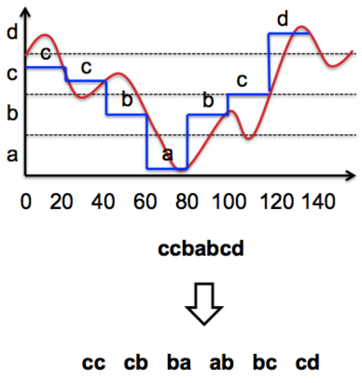
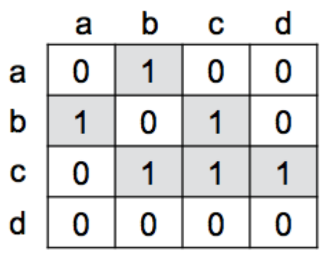


Fig. 1. Signal transformation: Quantisation



	a	b	c	d
a	0	1	0	0
b	1	0	1	0
c	0	1	1	1
d	0	0	0	0

Fig. 2. Signal transformation: generation of bitmap for input *edcbcbacff*.

Using the latest sample time t_N as the zero time reference is intuitive for online data streaming and this phase shift has no effect on the magnitude of power.

Then it is quite simple to manipulate and express Eq. (6) as a recursive function of its previous value at $N-1$:

$$\begin{aligned} X(\omega, N) &= (1-\gamma)\sum_{n=0}^{N-1}\gamma^{N-i}x_n e^{-j\omega(t_n-t_N)} + \\ &\quad (1-\gamma)x_N e^{-j\omega(t_N-t_N)} \\ &= X(\omega, N-1)\,\gamma\, e^{j\omega(t_N-t_{N-1})} + \\ &\quad (1-\gamma)x_N \end{aligned} \tag{7}$$

The phase shift is based on the time lapsed after the previous update, $\Delta t = t_N - t_{N-1}$ and does not depend on any absolute time reference. Updates require $O(1)$ computation.

The power is then computed by:

$$P(\omega, N) = |X(\omega, N)|^2 \tag{8}$$

Note that while we choose to use M evenly spaced frequencies in this work, the algorithm allows for the selection for an arbitrary number frequencies. Non-uniform spacings may be advantageous in some applications.

B. Anomaly Detection

The anomaly detector employed in this work is based on an algorithm proposed by Kumar et. al. [11]. The numerical input is first quantised to a discrete representation. While Kumar et. al. used Symbolic Aggregate approXimation (SAX) [12], which generates symbols which are approximately equiprobable, this requires two passes through the data. Our implementation processes the data in a single pass by taking the most significant b-bits of the data. This process is illustrated in Fig. 1, using an 4 symbol alphabet, represented by the symbols a-d. The signal string output is *ccbabcd*.

Bitmaps are then constructed from the time series of symbols in a window of size W. The frequency of all contiguous sequences of length-d symbols (i.e. all d-grams) is calculated, and used as entries in a $\sqrt{(b^d)} \times \sqrt{(b^d)}$ array. This can be considered a bitmap with colours representing the different counts, or a single dimensional array of size b^d. For example, considering the time series in Fig. 1 and $d = 2$, Fig. 2 illustrates the resulting bitmap for all adjacent pairs, the entries containing their frequency.

To form detector and reference bitmaps, BM_D and BM_R, time series bitmaps are calculated for windows of two different sizes, W_D and W_R respectively. Typically, W_R is much larger than W_D. These are then compared by computing the sum of squared differences between these two bitmaps considered as one dimensional arrays, to arrive at a final score i.e.

$$s = \sum_{i=0}^{b^d-1}\left(\frac{BM_R[i]}{W_R-1} - \frac{BM_D[i]}{W_D-1}\right)^2. \tag{9}$$

Each frequency count is normalised by dividing by $W-1$ to allows for the two differently sized windows to be compared.

The anomaly detection is conducted over M channels, resulting in M scores s_j ($j = 0 \ldots (M-1)$). The consolidated anomaly score is simply

$$a = \frac{1}{M}\sum_j s_j \tag{10}$$

and an anomaly is detected if

$$anomaly = \begin{cases} \text{TRUE} & a > l \\ \text{FALSE} & otherwise \end{cases} \tag{11}$$

where l is a user-defined threshold.

While summation was used to aggregate anomaly scores over the channels, many alternative techniques are available. For example, the max function could result in more sensitive detection. This will be explored further in future work.

IV. Implementation

Fig. 3 is a block diagram illustrating the implementation. The Recursive Non-Uniform Discrete Fourier transform (RD) takes the input time series and generates M time series of power levels at different frequencies. The individual channels are passed to M parallel anomaly detectors which perform signal discretization (SD), bitmap generation (BM) and score calculation (C).

The implementation is described in Alg. 1 and parameterised according to Tab. IV, any of which can be modified at compile time. Since the M frequency components can

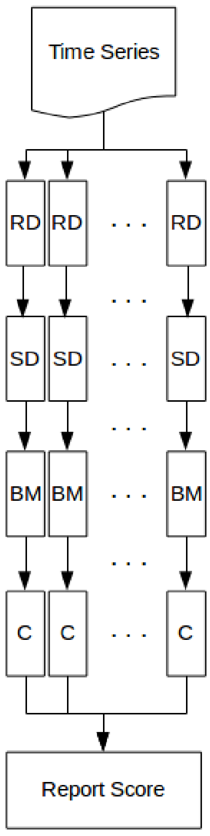


Fig. 3. Block diagram of power spectra anomaly detection system implementation.

be computed independently, the computation complexity can range from $O(1)$ in which all channels are processed in parallel, to $O(M)$ when they are processed one channel at a time.

In our implementation, several further optimisations are applied:

1) Computation of the exponential $e^{j\omega_j \Delta t}$ avoided by using a 512-entry lookup table and all computation is done in fixed point with wordlength L.
2) The score computation in the last line of Alg. 1 is computed incrementally as follows. For each iteration, the only entries that have changed in BM_R are those for d_R and i_D. Similarly for BM_D, the only entries that have changed are d_D and i_D. Since the remaining entries have been unchanged, the score can be updated incrementally from these values by subtracting the old squared difference term and adding back the new one. This computation requires six addition/subtractions and six multiplications for each of the four changed indices.
3) We choose combinations of the L, b and d parameters so that the number of bits of memory, $L \times b^d$, is less than the size of a block memory on the FPGA.

TABLE I
DEFAULT PARAMETERS FOR THE SAD IMPLEMENTATION.

Parameter	Description	Default
L	Global wordlength	16
γ	Recursive DFT decay rate	0.995
W_D	Detector window length	9
W_R	Reference window length	33
M	Number of frequency channels	4-32
b	Number of symbols in alphabet	8
l	Anomaly threshold	0.5
d	Bitmap level	2

V. RESULTS

This section describes the resource utilisation, performance and accuracy of the implementation. The implementation was created in Xilinx Vivado HLS 2013.4. The target device was a Xilinx Virtex 7 XC7VX690TFFG1930-3.

In an effort to promote reproducible research, our implementation is available from reference [13]. The University of California Riverside (UCR) datasets used in our paper are available from reference [14].

A. Resource Utilisation and Performance

Default parameters as summarised in Tab. I were used to create an $M = 1$ design. This was synthesised to obtain the FPGA resource utilisation summarised in Tab. III. Vivado reported an initiation interval of 10 clock cycles, a latency of 17 cycles and a maximum frequency 250 MHz. Projections for the performance of multiple channels are given in Tab. II.

The datapath for the design requires a total of 14 L-bit fixed-point multpliers, 3 subtractors, and 10 adders, and is independent of the other parameters. The total memory in bits required for the design is $2MLb^d + L(W_R + W_N)$. The first term accounts for the reference and detection bitmaps, and the second for window buffers.

As can be seen from Tab. III, DSP resources restrict the maximum number of parallel designs to approximately 256. However, designs can go beyond this value using lookup table (LUT) resources. The next limit are BlockRAMs which limit M to approximately 1000.

The same C implementation used to synthesise the FPGA design was compiled using gcc version 4.6.3 with -O3 compiler flag. Execution speed was tested on a 1.6 GHz Intel Core i5 Sandy Bridge processor (CPU) with 4 GB of memory, 3 MB cache and using the Mac OSX Mavericks Operating System. CPU execution time was recorded by appropriately instrumenting the program using the high resolution Linux timer. Both FPGA and CPU results do not include input/output overheads and the effect of other peripheral devices has been taken into account.

B. Power and Energy Consumption

The Vivado Power Report estimates the $M = 1$ design running at 250 MHz to have a power consumption of 0.3 W, 99% of this being static power. In comparison, the same

Algorithm 1 Online SAD.

```
procedure ANOMALY(x, $\Delta t$)                                          ▷ called for each new sample $x_i$
    for $j$ in $0 : (M-1)$ do                                             ▷ Over all frequencies
        $X_i \leftarrow X_{i-1}\gamma e^{j\omega_j \Delta t} + (1-\gamma)x_i$    ▷ Calculate updated recursive DFT for frequency $\omega_j = 2\pi j/M$
        $P_i \leftarrow |X_i|^2$                                          ▷ Calculate power spectra
        $Q_i \leftarrow$ shiftr($P_i, (L - \log_2 b - 1)$)                ▷ Quantize $X_i$ by shifting right
        $d_R \leftarrow$ index for substring $Q_{i-W_R+1} \dots Q_{i-W_R+b}$    ▷ This substring drops out of reference window
        $d_D \leftarrow$ index for substring $Q_{i-W_D+1} \dots Q_{i-W_R+b}$    ▷ This substring drops out of detector window
        decrement $BM_R[d_R]$                                             ▷ Account for substring $d_R$ dropping out
        decrement $BM_D[d_D]$                                             ▷ Account for substring $d_D$ dropping out
        $i_D \leftarrow$ index for substring $Q_{i-b+1} \dots Q_i$        ▷ This is the new substring index
        increment $BM_D[i_D]$                                             ▷ Account for new substring in $BM_D$
        increment $BM_R[i_D]$                                             ▷ Account for new substring in $BM_R$
        $S_i \leftarrow$ssd($S_{i-1}, BM_D, BM_R, d_D, d_R, i_R$)         ▷ Incrementally compute score according to (9)
    end for
end procedure
```

TABLE II
COMPARISON OF FPGA SAD IMPLEMENTATION COMPARED WITH A C IMPLEMENTATION, DISREGARDING INPUT/OUTPUT. TIMES REPORTED ARE TO PROCESS A SINGLE DATA INPUT.

M	Througput	Latency	CPU Time	Throughput Speedup	Latency Reduction
1	40 ns	68 ns	34 ns	0.9×	0.5×
4	40 ns	68 ns	273 ns	7×	4×
8	40 ns	68 ns	544 ns	14×	8×
16	40 ns	68 ns	1085 ns	27×	16×
256	40 ns	68 ns	17969 ns	449×	264×

TABLE III
RESOURCE UTILISATION FOR $M = 1$. NUMBERS IN PARENTHESES INDICATE THE AVAILABLE RESOURCES ON THE CHOSEN FPGA.

LUTs	Flip Flops	BlockRAM	DSP
220 (433200)	541 (866400)	3 (2940)	14 (3600)

processor used for the speed tests draws 1.50 W, making the FPGA approximately 5× more power efficient. The CPU requires 34 ns which is similar to that of the FPGA, making the energy efficiency approximately 4.25× better. For larger values of M, energy efficiency is significantly higher.

C. Verification of SAD

The correctness of the implementation was verified via simulation using Vivado. The utility of the SAD algorithm for detecting anomalies is demonstrated in this section by testing on three different types of time series data [14]: the Marotta space shuttle valve, tracking in a 2D video and an electrocardiogram (ECG).

Each dataset was sourced from the online repository maintained by Keogh et al. [14]. Unless specified otherwise, the parameters for SAD were determined by a grid search over a range of values and the accuracy was determined by comparing detected anomalies with anomalies known a-priori. If two configurations produced very similar anomaly scores, the configuration which reduced the computational requirements was chosen.

As an example of automated machine prognostics, we used the Marotta space shuttle value time series. Fig. 4 shows a selected subset of the time series along with the aggregated anomaly score using SAD with parameters $M = 64, \gamma = 0.95, W_R = 5000, W_D = 1000$ and $b = 4$. It can be seen in the figure that there are two regions of anomalous behaviour. Between sample 500-1000, the score is increased due to the presence of noise and the first large spike around sample 400. From sample 4100-5000, an unexpected notch is present. The dotted line represents the user-defined threshold l set to 0.5. Fig. 5 shows the corresponding power spectra and the anomaly spectrum (these are best viewed in colour). It can be seen that the anomaly spectra forms distinct patterns in each of the two anomalous regions.

The SAD parameters for the 2D video hand tracking time series were $M = 16, \gamma = 0.99, W_R = 600, W_D = 100$ and $b = 8$. Fig. 6 shows a selected subset of the time series along with the aggregated anomaly score using SAD. The individual channels are shown in Fig. 7. The anomaly around 2000 clearly causes an increased anomaly score in the low and high frequency channels, and potential for using scores over different frequency channels is apparent.

As a final example, wearable electronics have been increasing in popularity, personal health monitoring technology seems likely to be available to consumers in the coming years. As such, low power, automated health monitor hardware will become increasingly important. In this test, real ECG data was used to simulate this application. A selected portion of the ECG time series is shown in Fig. 8 along with the anomaly score. SAD parameters were $M = 16, \gamma = 0.9, W_R = 300, W_D = 100$ and $b = 8$. Around sample 1100, a distinct peak appears in the anomaly score corresponding to a premature ventricular contraction with a difference of 2× between the anomaly discord and the rest of the signal. This allows for a large range of possible threshold values suitable

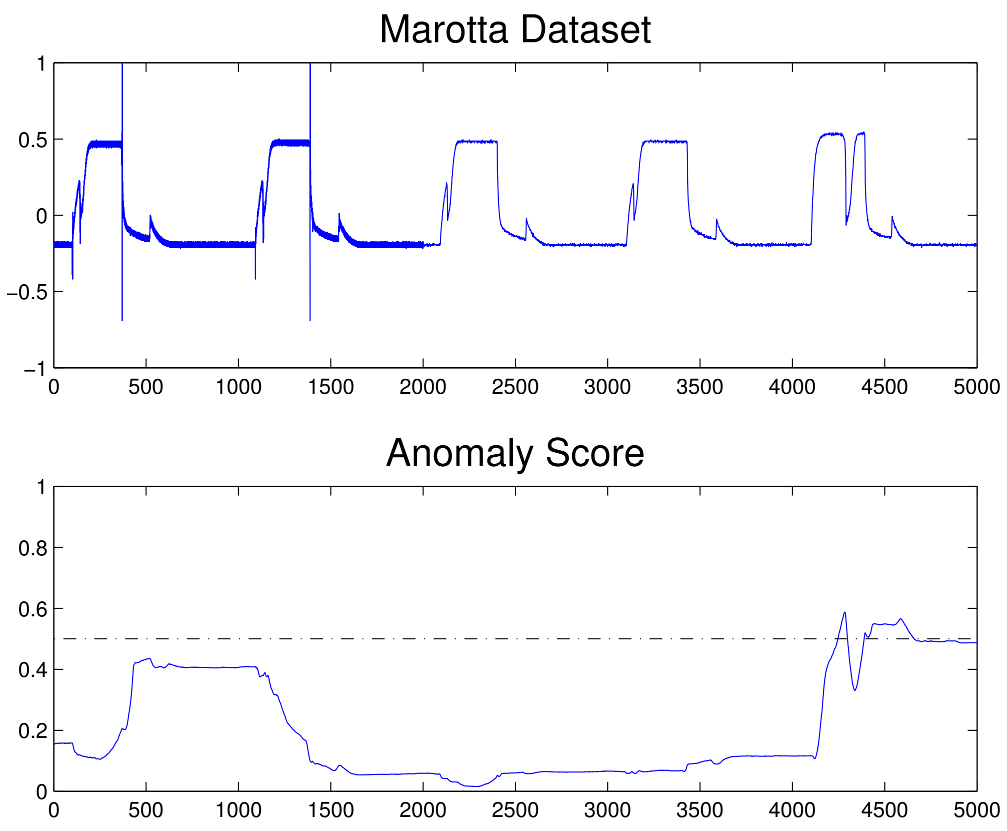


Fig. 4. Detection of an anomaly in the Marotta space shuttle valve time series. The raw data is shown as well as an aggregated anomaly score.

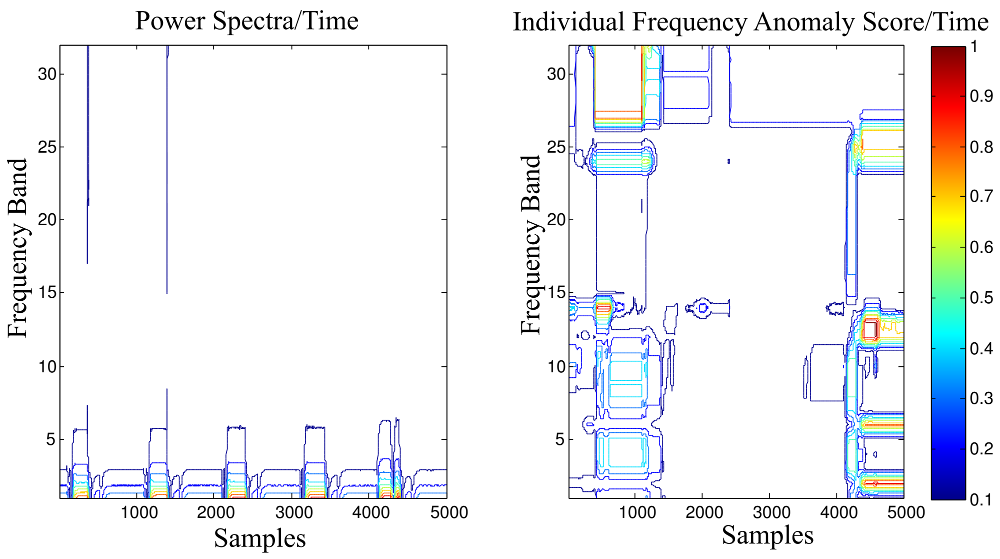


Fig. 5. Power and anomaly spectrum for the Marotta space shuttle valve time series.

for identifying the anomaly.

VI. Conclusion

In this paper, new online algorithms for Spectral Anomaly Detection (SAD) were proposed which lead to efficient hardware implementations. We further showed that our implementations offer one to two orders of magnitude improvement in speed, latency, power and energy over a single threaded implementation compiled from the same C source code.

Our work demonstrates the feasibility of addressing severely

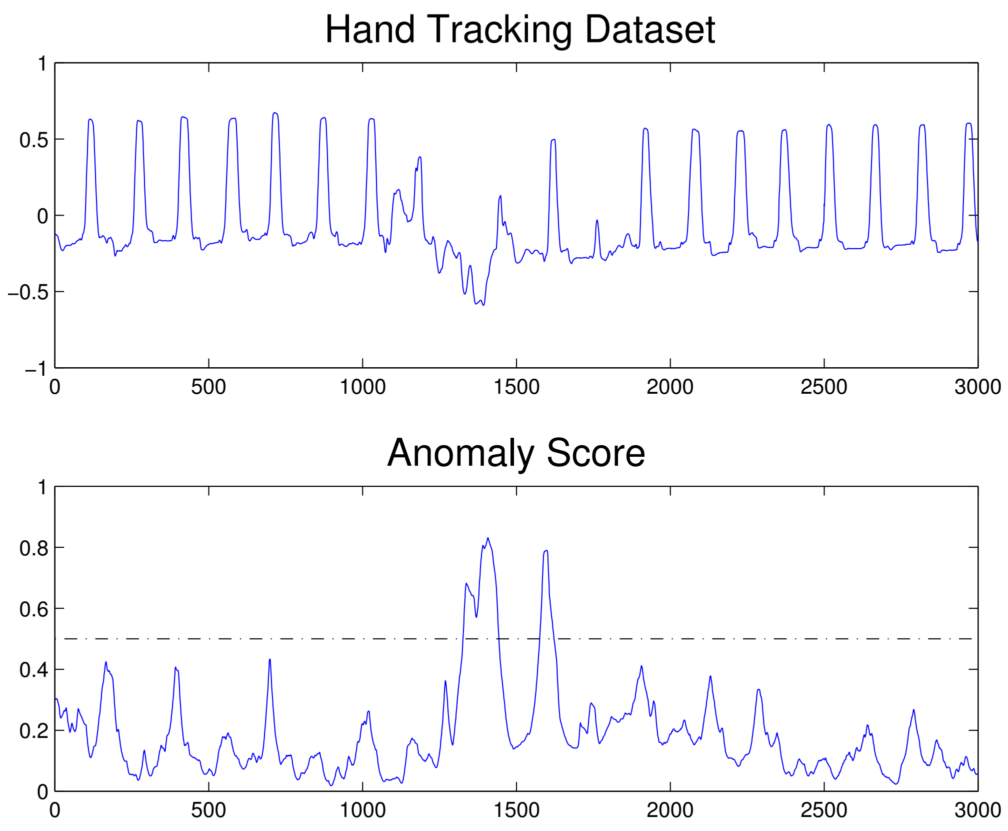


Fig. 6. Hand tracking in a 2D video. The raw data is shown as well as an aggregated anomaly score.

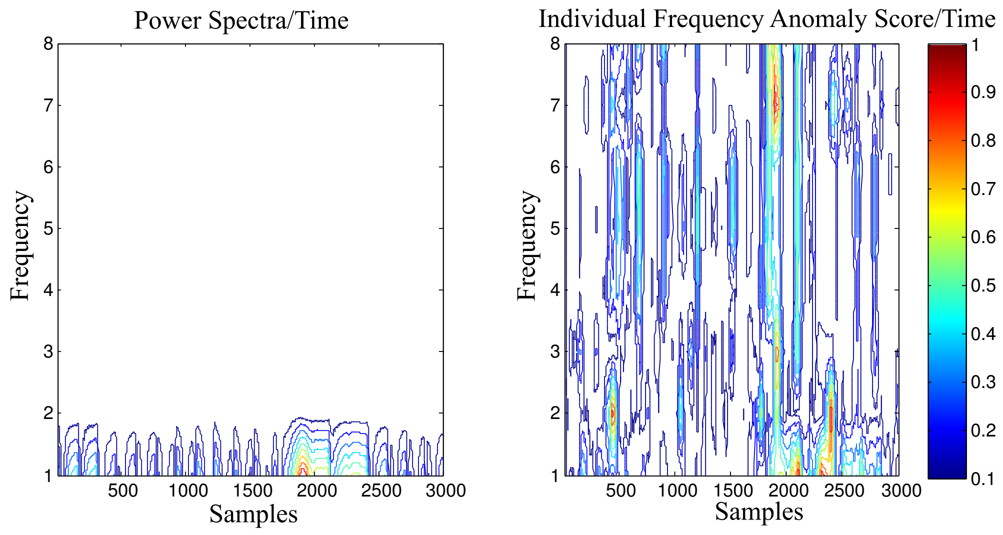


Fig. 7. Power and anomaly spectrum for the hand tracking time series.

constrained real-time SAD applications using FPGA technology. We believe that there are abundant opportunities for the application of this work in prognostics and health management.

In future work, we will: study alternative ways to consolidate the anomaly scores over different frequencies; test on irregular time series; test against multicore implementations; optimise FPGA hardware utilisation for better performance and devise better techniques for automatically determining parameters for our SAD implementation.

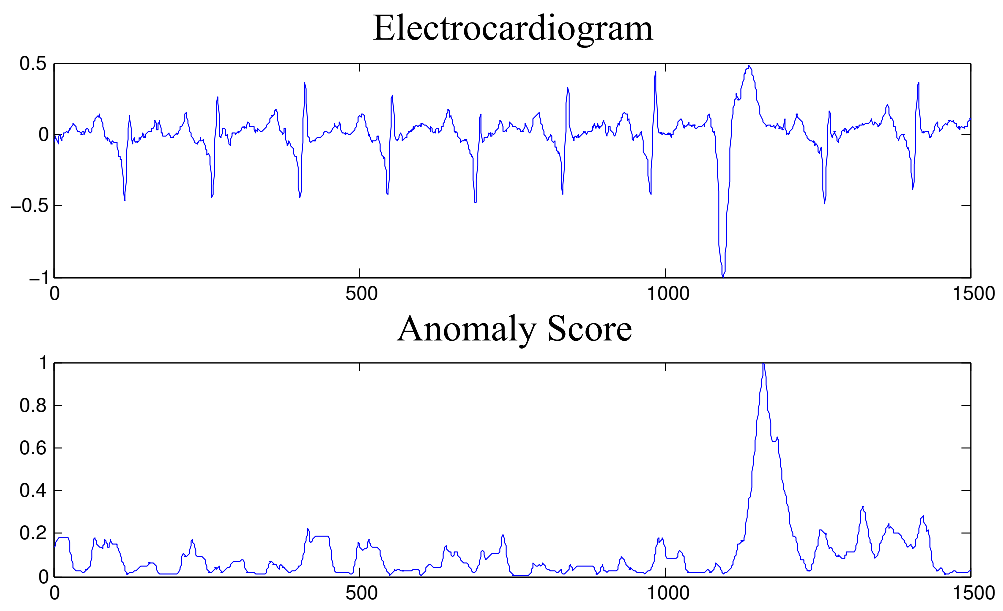


Fig. 8. Detection of an anomaly in ECG data. The raw data is shown as well as an aggregated anomaly score.

Acknowledgement

This research was partially supported by Australian Research Councils Linkage Projects funding scheme (project numbers LP110200413 and LP130101034).

References

[1] P. Hayton, S. Utete, D. King, S. King, P. Anuzis, and L. Tarassenko, "Static and dynamic novelty detection methods for jet engine health monitoring." *Philos Trans A Math Phys Eng Sci*, vol. 365, no. 1851, pp. 493–514, 2007. [Online]. Available: http://www.biomedsearch.com/nih/Static-dynamic-novelty-detection-methods/17255049.html

[2] J. D. Scargle, "Studies in astronomical time series analysis. ii-statistical aspects of spectral analysis of unevenly spaced data," *The Astrophysical Journal*, vol. 263, pp. 835–853, 1982.

[3] S. A. Pasha and P. H. Leong, "Cluster analysis of high-dimensional high-frequency financial time series," in *IEEE Symposium on Computational Intelligence for Financial Engineering & Economics - (CIFEr)*, 2013, pp. 68–75.

[4] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Comput. Surv.*, vol. 41, no. 3, pp. 15:1–15:58, Jul. 2009. [Online]. Available: http://doi.acm.org/10.1145/1541880.1541882

[5] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in C (2Nd Ed.): The Art of Scientific Computing*. New York, NY, USA: Cambridge University Press, 1992.

[6] A. Patcha and J.-M. Park, "An overview of anomaly detection techniques: Existing solutions and latest technological trends," *Comput. Netw.*, vol. 51, no. 12, pp. 3448–3470, Aug. 2007. [Online]. Available: http://dx.doi.org/10.1016/j.comnet.2007.02.001

[7] M. Agyemang, K. Barker, and R. Alhajj, "A comprehensive survey of numeric and symbolic outlier mining techniques," *Intell. Data Anal.*, vol. 10, no. 6, pp. 521–538, Dec. 2006. [Online]. Available: http://dl.acm.org/citation.cfm?id=1609942.1609946

[8] V. Hodge and J. Austin, "A survey of outlier detection methodologies," *Artif. Intell. Rev.*, vol. 22, no. 2, pp. 85–126, Oct. 2004. [Online]. Available: http://dx.doi.org/10.1023/B:AIRE.0000045502.10941.a9

[9] A. Das, D. Nguyen, J. Zambreno, G. Memik, and A. Choudhary, "An FPGA-based network intrusion detection architecture," *Information Forensics and Security, IEEE Transactions on*, vol. 3, no. 1, pp. 118–132, March 2008.

[10] K. M. Carter and W. W. Streilein, "Probabilistic reasoning for streaming anomaly detection," in *Statistical Signal Processing Workshop (SSP), 2012 IEEE*. IEEE, 2012, pp. 377–380.

[11] N. Kumar, V. Lolla, and E. Keogh, "Time-series Bitmaps: a Practical Visualization Tool for Working with Large Time Series Databases," in *SIAM International Conference on Data Mining*, 2005, pp. 531–535. [Online]. Available: http://epubs.siam.org/doi/abs/10.1137/1.9781611972757.55

[12] J. Lin, E. Keogh, S. Lonardi, and B. Chiu, "A Symbolic Representation of Time Series, with Implications for Streaming Algorithms," in *Proceedings of the 8th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, 2003, pp. 2–11.

[13] D. J. Moss, Z. Zhang, N. J. Fraser, and P. H. Leong, "Spectral anomaly detector: Online repository," 2014. [Online]. Available: https://github.com/djmmoss/SAD

[14] E. Keogh, X. Xi, L. Wei, and C. A. Ratanamahatana, "The ucr time series classification/clustering homepage," 2011. [Online]. Available: http://www. cs. ucr. edu/˜ eamonn/time_series_data

Errata for

An FPGA-based Spectral Anomaly Detection System

in proceedings of
International Conference on Field Programmable Technology,
ICFPT 2014, Shanghai

Duncan J.M. Moss, Zhe Zhang, Nicholas J. Fraser and Philip H.W. Leong

In Section V part B, the power consumption was incorrectly reported. The sentences

In comparison, the same processor used for the speed tests draws $1.50\ W$*, making the FPGA approximately* $5\times$ *more power efficient. The CPU requires* $34\ ns$ *which is similar to that of the FPGA, making the energy efficiency approximately* $4.25\times$ *better.*

should be changed to

In comparison, the same processor used for the speed tests draws $11.25\ W$*, making the FPGA approximately* $37.5\times$ *more power efficient. The CPU requires* $34\ ns$ *which is similar to that of the FPGA, making the energy efficiency approximately* $33.75\times$ *better.*