Article:
Biswas, D, Maharatna, K, Panic, G et al. (5 more authors) (2017) Low-Complexity Framework for Movement Classification Using Body-Worn Sensors. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 25 (4). pp. 1537-1548. ISSN 1063-8210

https://doi.org/10.1109/tvlsi.2016.2641046

Low-complexity Framework for Movement Classification using Body-worn Sensors

Dwaipayan Biswas, Koushik Maharatna, Member, IEEE, Goran Panic, Evangelos B. Mazomenos, Josy Achner, Jasmin Klemke, Michael Jöbges and Steffen Ortmann

Abstract— We present a low-complexity framework for classifying elementary arm-movements (reach-retrieve, lift-cup-to-mouth, rotate-arm) using wrist-worn, inertial sensors. We propose that this methodology could be used as a clinical tool to assess rehabilitation progress in neurodegenerative pathologies tracking occurrence of specific movements performed by patients with their paretic arm. Movements performed in a controlled training-phase are processed to form unique clusters in a multi-dimensional feature-space. Subsequent movements performed in an uncontrolled testing-phase are associated to the proximal cluster using a minimum distance classifier (MDC). The framework involves performing the compute-intensive clustering on the training-dataset offline (Matlab) whereas the computation of selected features on the testing-dataset and the minimum distance (Euclidean) from pre-computed cluster centroids are done in hardware with an aim of low-power execution on sensor nodes.

The architecture for feature-extraction and MDC are realized using Coordinate Rotation Digital Computer based design which classifies a movement in (9n+31) clock cycles, n being number of data samples. The design synthesized in STMicroelectronics 130nm technology consumed 5.3 nW @50 HZ, besides being functionally verified upto 20 MHz, making it applicable for real-time high-speed operations. Our experimental results show that the system can recognize all three arm-movements with average accuracies of 86% and 72% for four healthy subjects using accelerometer and gyroscope data respectively, whereas for stroke survivors the average accuracies were 67% and 60%. The framework was further demonstrated as a FPGA-based real-time system, interfacing with a streaming sensor unit.

Index Terms— Clustering, classification, activity recognition, CORDIC, FPGA, low-complexity

I. Introduction

ACTIVITY recognition (AR) in nomadic settings has gained prominence in the research community for assessing human mobility through remote monitoring systems. Remote monitoring for long durations has been aided by the advancements ubiquitous and mobile computing facilities primarily using radio-frequency identification (RFID) [1], low-cost inertial sensors [2], and fusion of inertial sensor and vision-based approaches [3]. RFID and vision-based methods are primarily restricted to a defined region catering for indoor activities, requiring an un-hindered surveillance [4]. Moreover, for systems requiring real-time information, using high complexity image processing algorithms can lead to slower analysis [3]. Hence, body-worn inertial sensors have gained prominence over other approaches [5][6], particularly with the shift in research focus towards monitoring human activities performed in daily life which is a more natural indicator of the subject's involvement as compared to monitoring only during a prescribed exercise/training phase.

The fundamental requirement for a long-term continuous monitoring scenario using resource constrained WSN nodes, is a low-power operation to prolong the battery life. Typical remote monitoring systems employ computationally intensive data processing steps like feature extraction from the sensor data and pattern recognition (e.g. classification) which are carried out on off-line computational facilities. This involves continuous data transmission incurring significant amount of energy expenditure at the radio front-end of the sensors. Hence, for applications involving continuous remote monitoring (e.g. motion/fall detection for the elderly population in daily life), a low-power strategy is of paramount importance which can be achieved by performing low-complexity data processing in resource constrained environment of the sensor node itself [7].

In this work, we focus on the application area of arm movement recognition aimed at stroke rehabilitation. In neurodegenerative pathologies (e.g. stroke or cerebral palsy), detecting and classifying particular arm movements (e.g. clinically prescribed exercises) performed in daily life, can over time provide a measure of rehabilitation progress. A systematic exploration to recognize three fundamental movements of the upper limb associated with daily living activities using wrist-worn inertial sensors has already been reported in [8] employing a clustering and minimum distance classification based approach. Sensor data collected from each subject in a constrained training phase (e.g. in the laboratory) are clustered to form three unique clusters representing each movement in a multi-dimensional feature space. A minimum distance classifier (MDC) computes the proximity of the test data collected in an unconstrained scenario (e.g. out-of-laboratory), to each of the clusters. Classification of the movements performed in the subsequent testing phase involving the essential steps of – (1) computing selected time-domain

This work was supported by the European Union under the Seventh Framework Programme, grant agreement #288692, StrokeBack.

D. Biswas and K. Maharatna are with Electronics and Computer Science, University of Southampton, Southampton, UK (e-mail: {db9g10, km3}@ecs.soton.ac.uk). E.B Mazomenos is with the Centre for Medical Image Computing, UCL, London, U.K. (email: e.mazomenos@ucl.ac.uk). J. Achner, J. Klemke and M. Jöbges are with Brandenburg Klinik, Berlin, Germany (email: {josy.achner, ergo_n1, joebges}@brandenburgklinik.de). G. Panic and S. Ortmann are with IHP, Frankfurt (Oder), Germany, (email: {panic, ortmann}@ihp-microelectronics.com).

features from the sensor data and (2) the distance to the pre-computed cluster centroids can be mapped to a low-complexity architecture to achieve real-time detection of arm movements thereby providing an energy efficient solution towards long-term operation of wearable sensors [7].

Hence in this paper, we propose the design and implementation of a CoOrdinate Rotation Digital Computer (CORDIC) based low-complexity MDC for real-time arm movement recognition. The fundamental mathematical processes of the MDC have been formulated using the different transcendental functions realizable using CORDIC and an optimized implementation strategy has been adapted, analyzing the shared computational stages. The algorithm proposed in [8] has been implemented in an offline-online resource sharing mechanism, where the time and memory intensive process of feature extraction, selection and cluster formation using 10 runs of 10-fold cross-validation (CV), on the training data were done in an offline mode (in Matlab). The computation of the selected features (required for cluster formation) on the testing data and computation of the minimum distance (Euclidean) from the pre-computed cluster centroids was done in hardware, targeting real-time implementation.

The design was synthesized using STMicroelectronics 130 nm technology with a supply voltage of 1.08V and occupied 242K NAND2 equivalent cell area and consumed 5.3 nW @ 50 Hz, resulting in a low-complexity framework, applicable for real-time operations within a WSN node. The application area we consider is that of human activity recognition where a sampling frequency of up to 50 Hz is deemed sufficient for capturing kinematic information [9][10]. The design was further verified up to higher frequencies (viz. 20 MHz) and a total chip area of the layout was calculated as 2.21 mm^2. Our experimental results to classify movements of four healthy subjects and stroke survivors involving an archetypal activity of daily living (ADL), 'making-cup-of-tea', show that the system can recognize all three arm movements with average accuracies of 86% and 72% for healthy subjects using accelerometer and gyroscope data respectively, whereas for stroke survivors the average accuracies were 67% and 60%. The framework was further demonstrated as a real-time working system, interfacing a streaming inertial sensor unit, host PC and DE4 FPGA board to facilitate serial port controls, recognizing a performed arm movement in approximately 0.6 ms @780 KHz. The main contributions can be enlisted as the development of:

- CORDIC based low-complexity MDC architecture for online AR;
- system demonstrator for real-time AR;
- generic offline-online framework in conjunction with clustering, applicable in wide range of AR applications.

The rest of the paper is structured as follows: an overview of the application setup is described in Section II and the theoretical formulation of the MDC in terms of CORDIC rotation along with the architecture for the proposed framework is described in Section III. Section IV describes the implementation and performance evaluation of the system. Finally, related literature and discussion are presented in Sections V and VI respectively.

II. Application Setup

With an aim of continuous monitoring of activities performed in daily life by patients, the specific movements (or exercises) that need to be tracked as defined by clinicians need to be performed multiple times, following an exercise regime or a gaming session, in a controlled environment (clinic or home) [11][12]. The sensor data collected during this phase can be analysed through CV to determine the best cluster forming features and obtain the centroids of each cluster corresponding to each movement. This helps to perform a clinical profiling of the individual patient with respect to their movement quality. Movements performed in the uncontrolled nomadic environment (involving daily activities) can be associated to the proximal cluster centroid using the MDC to detect the occurrence of those particular movements. The merits of using the clustering based methodology over a plethora of other machine learning algorithms [10] for fine-grained arm movements, have been presented in detail in [8] whereas issues such as sensor selection/placement, data fusion have been addressed in [13] for arm movement recognition.

Given the application framework, this methodology can be implemented for online detection of arm movements in a resource-constrained environment of body-worn sensor nodes. The offline processing of the training data, involving the key steps of cluster formation and feature selection need only be done when requested by the clinician, depending on the rehabilitation progress of the patient over time. Furthermore, the test data can be classified in real-time by computing the required features and the distance to the pre-computed cluster centroids in near real-time, providing an energy efficient solution towards long-term operation of wearable sensors. The offline-online processing framework is illustrated in Fig. 1. Although here we have targeted arm movement as a case study, this framework can be suitably used for critical event monitoring such as fall-detection or in sports medicine.

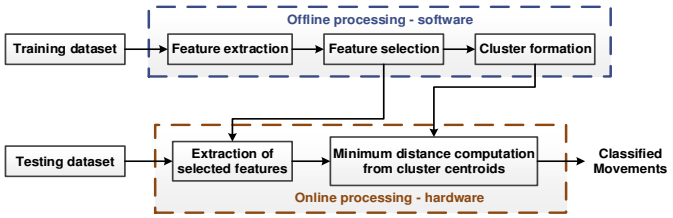

Fig. 1. Processing framework – offline/online processing of the training/testing dataset respectively.

Experiments were conducted in two phases - training/laboratory phase and a testing/out-of-laboratory phase on - four healthy subjects at the University of Southampton and four stroke survivors at the Brandenburg Klinik. The healthy subjects were both right arm dominant while the stroke population had either left or right arm impaired. For this investigation, three arm movements, elementary in nature were considered: (1) Action A – reach and retrieve object, (2) Action B – lift cup to mouth and (3) Action C – perform pouring/(un)locking action.

In the training phase, essential for the target cluster formation, each healthy participant performed – 240, 120 and 120 trials each of Action A, Action B and Action C respectively. The stroke survivors performed – 80 trials of A and 40 trials each of B and C. The collection of this training set helps to inherently capture the personalized (i.e. person-centric) movement patterns of the individuals through unique clusters augmenting accurate recognition [14]. The more number of trials pertaining to Action A w.r.t to B and C is representative of the generalised nature of the reach and retrieve movement performed frequently in daily lives.

The testing phase employs an archetypal activity-list (cf. Table I) emulating the process of 'making-cup-of-tea', commonly performed in daily life incurring repeated occurrences of the three investigated arm movements. The list comprises 20 individual activities having 10 occurrences of Action A and 5 each of Actions B and C. The healthy subjects performed the activity-list four times with a 10-minute rest period between trials whereas the stroke survivors performed two trials since they tend to tire quicker. The experiment was performed in an unconstrained manner ensuring wider range of variability in the data.

TABLE I
USE CASE ACTIVITY LIST – 'MAKING-CUP-OF-TEA'

	Activity	Action
1.	Fetch cup from desk	A
2.	Place cup on kitchen surface	A
3.	Fetch kettle	A
4.	Pour out extra water from kettle	C
5.	Put kettle onto charging point	A
6.	Reach out for power switch on the wall	A
7.	Drink glass of water while waiting for kettle to boil	B
8.	Reach out to switch off kettle	A
9.	Pour hot water from kettle in to cup	C
10.	Fetch milk from shelf	A
11.	Pour milk into cup	C
12.	Put bottle of milk back on shelf	A
13.	Fetch cup from kitchen surface	A
14.	Have a sip and taste the drink	B
15.	Have another sip while walking towards desk	B
16.	Unlock the drawer	C
17.	Retrieve biscuits from the drawer	A
18.	Eat a biscuit	B
19.	Lock the drawer	C
20.	Have a drink	B

For this investigation, we use tri-axial accelerometers (range ± 1.5 g) and tri-axial gyroscopes (range ± 500 °/s), housed in a Shimmer wireless 9DoF kinematic sensor module [15]. The impaired arm for the stroke survivors and the dominant arm for the healthy subjects, proximal to the wrist, were chosen for the sensor placement with the dorsal side of the forearm in contact with the XY plane and the Z-axis pointing away from it. Magnetometers were not considered for this investigation due to the presence of ferromagnetic materials in the home environment [16]. Data was collected @50 Hz, transmitted along with a timestamp to a host computer using Bluetooth.

III. ALGORITHM TO ARCHITECTURE MAPPING

The accuracy of any movement recognition technique is dependent on several factors such as: nature/number of activities, sensor type/number/placement, data mining and the classification methodology adopted [10]. Furthermore, there is a need for personalised evaluation especially for tracking activities that are susceptible to individual and temporal variation. In this paper, although the focus is primarily on an optimized architecture design for the testing phase, a brief overview of the algorithm and associated data processing especially in the training phase is quintessential since it determines the generation of the cluster centroids used by the MDC. The k-means clustering algorithm mentioned in [8], uses 10 time-domain features, extracted from each of the three axes of the accelerometer or the gyroscope sensors. The features are: 1) standard deviation, 2) root mean square, 3) information entropy, 4) jerk metric, 5) peak number, 6) maximum peak amplitude, 7) absolute difference, 8) index of dispersion, 9) kurtosis, 10) skewness.

The fundamental concept of clustering is to form groups of similar objects as a means of distinguishing them from each other and it is well-perceived that cluster analysis is primarily used for unsupervised learning where the class labels for the training data are unknown. However, k-means clustering can also be used for supervised learning as in our proposed methodology [8] where we are aware of the labels for the training data pertaining to the three movements, helping to have a definite estimate on the underlying cluster structure (three clusters), facilitating faster convergence during cluster formation. We use the regularized Mahalonobis distance considering the covariance of the data, where a parameter λ (0 or 1) is used to control the choice of distance measure (squared Mahalonobis or Euclidean). The clustering is performed on the feature vectors computed from the training data (accelerometer and gyroscope). It is performed in conjunction with a sequential forward selection (sfs) algorithm, selecting a combination of 2 to 30 ranked features (10 features computed on each tri-axial axes data) in each step and 10 runs of 10-fold CV (9 segments of training and 1 of testing – only considering training data) are carried out with each feature combination. Cluster centroids are selected based on an experimentally determined threshold (25%) of the expected number of data points for each of the three clusters formed (healthy subjects: Action A - 240 ± 60, Action B/C - 120 ± 30; patients: Action A - 80 ± 20, Action B/C - 40 ± 10). Therefore, offline processing provides a detailed list of feature combinations that resulted in a successful cluster formation and the highest corresponding accuracies (averaged over 10 runs) for each subject and each sensor type.

An important aspect is the choice of features since human AR studies typically incur the extraction of time and/or frequency domain features, as well as heuristic features from data which exhibit discriminative patterns for each movement. Commonly used frequency domain features as a result of signal transformation – Fourier, wavelet are well equipped in capturing dynamic movements like walking, running, etc. (high frequency components) while the orientation/postural information can be obtained from the low frequency

components. In this investigation we use time domain features since: 1) we consider fine-grain upper limb movement as compared to detecting gross/dynamic activities and static postures like standing, running, sitting, cycling, etc. and 2) time domain feature extraction incurs low complexity when mapped onto equivalent architecture.

A detailed study on sensor selection/placement and data fusion for the targeted arm movements have been reported in [13]. Particularly, it has been shown that higher recognition sensitivities are achieved using: 1) data from the wrist sensor module as compared to the elbow since the former is more responsive and produces significant discriminatory patterns for the arm movements being investigated and 2) similar time domain features extracted on individual sensor axes data as compared to considering he modulus of the tri-axial data and fusion of specific accelerometer-gyroscope signal combinations. Three unique sensor combinations for the wrist module (multiplying accelerometer-gyroscope signals) were created based on a priori consideration of the expected trajectory of the subject's arm with respect to the sensor position and orientation of the sensor axes for the investigated movements. The use of all the individual sensor signals, rather than a processed signal (i.e. moduli or fused), provides the classifier a wider pool of features to select and hence the recognition rate for the movements is reflected in the higher sensitivity achieved [13].

According to the application framework (cf. section II), the online processing stage aimed at real-time arm movement detection comprises of the key steps – 1) feature extraction from the test dataset and 2) associating the test data to the pre-computed cluster centroids using a MDC. In this section we present the architecture and implementation of the MDC in conjunction with the cluster centroids for detecting the three investigated arm movements. A detailed architecture and implementation of several of these features have been presented in [17], except for the jerk metric which is an important feature, quantifying the tremor inherent in the movement especially among the stroke population. Given the low-complexity when using CORDIC for formulating the features as demonstrated in [17] compared to other implementations, in this paper we use it to formulate – 1) the jerk metric and 2) the MDC for classifying the test data in the respective feature space. We present a brief overview of CORDIC fundamentals, used for the algorithmic-to-architecture formulation.

CORDIC is an iterative algorithm which uses 2D vector rotation for computing different transcendental functions employing the iterative equations:

$$\begin{aligned} x_{j+1} &= x_j - \mu\sigma_j . 2^{-j} \\ y_{j+1} &= y_j + \sigma_j . 2^{-j} . x_j \\ z_{j+1} &= z_j - \sigma_j . \alpha_j \end{aligned} \tag{1}$$

where, $[x_j, y_j]^T$, z_j and $\sigma_j \in \{1, -1\}$ are the intermediate result vector, the residual angle and the direction of vector rotation at the j-th iteration stage respectively; $\mu \in \{1, 0\}$ being the coordinate of rotation – circular and linear respectively.

TABLE II
GENERALISED CORDIC ALGORITHM IN TWO CO-ORDINATE SYSTEMS

μ	ROTATION MODE ($z_0 \rightarrow 0$)	VECTORING ($y_0 \rightarrow 0$)
1	$x_n = K(x_0 \cos z_0 - y_0 \sin z_0)$ $y_n = K(y_0 \cos z_0 + y_0 \sin z_0)$ $z_n = 0$	$x_n = K\sqrt{x_0^2 + y_0^2}$ $y_n = 0$ $z_n = z_0 + \tan^{-1}(y_0/x_0)$
0	$x_n = x_0$ $y_n = y_0 + x_0 z_0$ $z_n = 0$	$x_n = x_0$ $y_n = 0$ $z_n = z_0 + (y_0/x_0)$

In each coordinate system, CORDIC in general, can be operated in two modes - Vectoring and Rotation [18]. For an input vector $[x_0 \quad y_0]^T$, in the Vectoring mode ($y_0 \rightarrow 0$), the magnitude of the vector, angle between the initial vector and the principal coordinate axis is computed whereas in the rotation mode ($z_0 \rightarrow 0$), for a given angle of rotation the final vector is computed. These can be used for computing a series of transcendental functions as shown in Table II [18]. The transcendental functions generated by the vectoring CORDIC operation can be used for feature computation and the MDC. We use Vec_c, and Vec_l as operators representing vectoring CORDIC operation in circular and linear coordinate system respectively. The input dataset is represented by d_{si}, where $i \in \{0, 1, 2 \ldots n\text{-}1\}$ and d_i is the output of vectoring CORDIC operation on $d_{s(i-1)}$ data sample. The features and the MDC have been formulated in terms of CORDIC operation, in line with this convention.

A. Feature - Jerk Metric (jm)

The jerk metric characterizes the average rate of change of acceleration in a movement. It is calculated as the rms value of the derivative of the acceleration (jerk) normalized by the maximum value of the integral (velocity) [19] as shown in (2).

$$\text{jerk metric} = \frac{\text{rms}\left[\dfrac{d(d_{si})}{dt}\right]}{\max\left[\int (d_{si})\, dt\right]} \tag{2}$$

It is important to note here that although the calculation of jerk is physically related to the acceleration data but the same computing logic is also applied to the rotation data from the gyroscope, since the computed metric serves its purpose as a discriminating feature for characterising the movements. Since the data samples are equally spaced due to the constant sampling frequency, the first derivative is computed as the difference of the consecutive data samples using a subtractor. The integral of the data is computed using trapezoidal integration which involves the addition of the consecutive data samples and a divide by 2 (implemented as right shift). From (2), it can be deduced that the rms of the first derivative of the data samples ($\dot{d}_{si}$) can be computed using the operator Vec_c, which is shown in (3). The samples $\dot{d}_{si}$ are used as the y input to the CORDIC and the x-component of the output is fed back

to the x-component of the CORDIC input.

$$\text{rms} = \frac{1}{\sqrt{n}}\left(\prod_{i=0}^{n-1} \text{Vec}_c[d_i \quad \dot{d}_{si}]^T\right)_x \tag{3}$$

Therefore, with new data samples d_{si} arriving at each clock cycle, the x-component of the CORDIC output is computed as:

$$d_i = K\sqrt{d_{s0}^2 + d_{s1}^2 + \ldots + d_{s(i-1)}^2} \tag{4}$$

The x-component of the output generated after every complete CORDIC operation is scaled with a scale factor K. This is an essential step as feeding this result without scaling into the x-component of the CORDIC input results in an accumulation of the scale factor corresponding to each d_{si}, thereby affecting the formulation in (4). Hence, the scale factor compensation is invoked after every complete CORDIC operation (comprising N stages) with a set of input data, feeding the compensated output to the x-input of the CORDIC in the next iteration. Following n operations with the scale factor compensation the x output of the CORDIC yielding the final result is multiplied with $1/\sqrt{n}$ for obtaining the rms. The value $1/\sqrt{n}$ is pre-computed (n being a fixed number) and is multiplied with the final CORDIC output with the help of a reduced complexity multiplier-less shift-and-add technique or fixed-number multiplier.

The jerk metric is finally computed using the CORDIC operator Vec_l as shown in (5). Referring to Table I, $\max(\int d_{si})$ and $\text{rms}\left(\dot{d}_{si}\right)$ are set as the x_0 and y_0 inputs to the CORDIC, operating in vectoring mode in the linear coordinate system.

$$\text{jerk metric} = \left(\text{Vec}_l[\max\left(\int d_{si}\right) \quad \text{rms}\left(\dot{d}_{si}\right)]^T\right)_z \tag{5}$$

The implementation includes 1 subtractor and CORDIC (Vec_c) for computing $\text{rms}\left(\dot{d}_{si}\right)$ and 1 adder for computing $(\int d_{si})$ by trapezoidal integration. Finally, CORDIC (Vec_l) is re-used for computing the value of the feature. The jerk metric is dependent on the rms of the derivative and maximum of the integral taking (n + 1) cycles. Considering n as 256 data samples, representative of a movement for approximately 5 seconds (@50 Hz), facilitates a multiplier-less shift-and-add operation.

B. Minimum distance classifier (MDC)

The MDC methodology has been illustrated through a mathematical approach having three clusters (A, B, C formed using k-means on the training dataset for the three movements) and a test vector (T) to be associated in a 2-dimensional feature space (f_1 and f_2) in Fig. 2. The distance of T from each of the three centroids are denoted by d_A, d_B, d_C which are compared to estimate its proximity to the clusters.

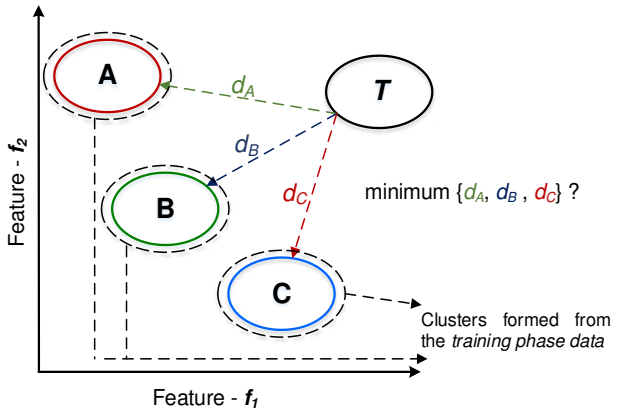

Fig. 2. Illustration of the minimum distance classification methodology.

According to Fig. 2, the two dimensional co-ordinates are: Cluster centroid A – (f_{A1}, f_{A2}) and Test vector T – (f_{T1}, f_{T2}). This feature space (f_1, f_2) can be extended to incorporate all 30 features. The Euclidean distance of the test feature vectors from the centroid can be computed as in (6), which can be further reframed (7), having functional similarity to rms computation and can be realized using CORDIC operator Vec_c (8), where the data samples d_{Asi}, are the computed differences between the feature vectors of the test dataset and the cluster centroids.

$$d_A = \sqrt{(f_{T0} \sim f_{A0})^2 + (f_{T1} \sim f_{A1})^2 + \cdots + (f_{T29} \sim f_{A29})^2} \tag{6}$$

$$d_A = \sqrt{\left(\sum_{i=0}^{29} d_{Asi}^2\right)} \tag{7}$$

$$d_A = \left(\prod_{i=0}^{n-1} \text{Vec}_c[d_{Ai} \quad d_{Asi}]^T\right)_x \tag{8}$$

Similar to rms computation (3), the samples d_{Asi} are fed to the y input of CORDIC while the final result (scaled with K) at the x output of CORDIC is obtained after n number of operations, where n is dependent on the number of features selected ($1 > n \leq 30$). Similarly, the distances d_B, d_C can be computed using Vec_c. The offline-online processing approach (cf. section II), has been illustrated in Fig. 3, representing the input-output signals which have been further described in Table III.

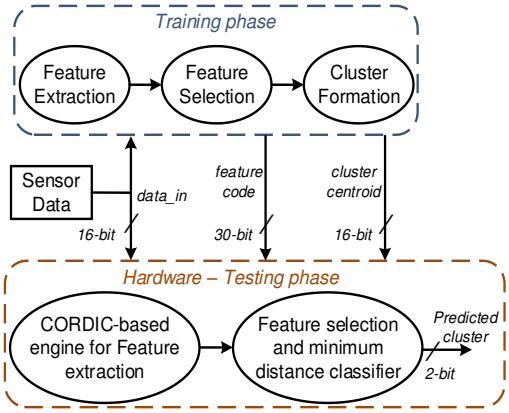

Fig. 3. Architecture for offline-online framework for MDC.

TABLE III
LIST OF INPUT-OUTPUT SIGNALS

Signals	Description
data_in	16-bit i/p for tri-axial sensor data corresponding to a movement performed in the testing phase (Acc_x, Acc_y, Acc_z or Gyro_x, Gyro_y, Gyro_z)
feature-code	30-bit i/p denoting the selected features out of total 30 features during cluster formation on the training dataset (having '1' for a selected feature else '0'); cf. Fig. 4.
cluster-centroid	16-bit i/p each for 3 cluster centroids formed from the features selected from the training phase data.
predicted-cluster	2-bit o/p for the predicted cluster computed as the minimum distance of the test dataset from the cluster centroids

The sequence of features (10 features) has been illustrated in Fig. 4, which are extracted from each tri-axial data segment (x, y and z) of each sensor type, thereby having total of 30 features [8]. The features selected (out of a total of 30) during the cluster formation are represented using a feature-code. An example 30-bit code: 000100000000000001001000000000, represents the features (3, 17, 20) viz. D_x (dispersion computed on x-axis data), jerk_y (jerk metric on y-axis) and rms_z (rms on z-axis) were selected during cluster formation.

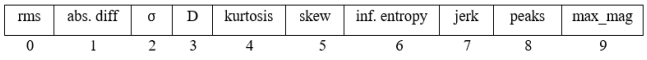

Fig. 4. Sequence of features extracted from each tri-axial data segment to form a 30-bit feature-code.

The architecture for the MDC, associating the test dataset to pre-computed cluster centroids is shown in Figs. 5 and 6. The feature-code helps to select the required features. The cluster centroid for that corresponding feature is selected through a sample counter (5-bit feature-counter) which counts through the 30-bit feature-code.

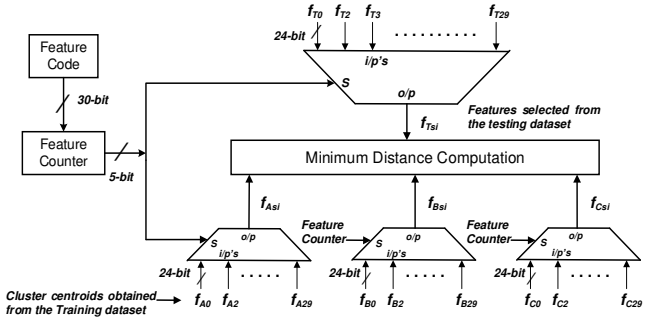

Fig. 5. Overview of the MDC architecture.

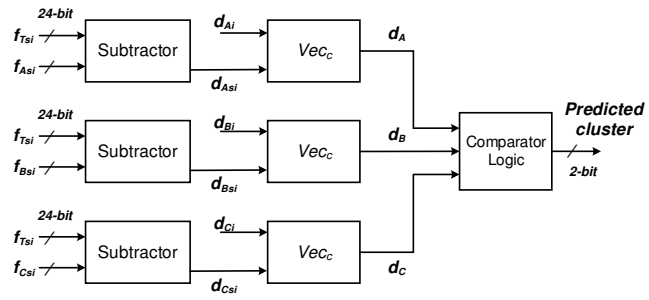

Fig. 6. Architecture for the minimum distance computation module.

The features selected from the testing set (f_{Tsi}) and the corresponding cluster centroids (f_{Asi}, f_{Bsi}, f_{Csi}) are passed onto the minimum distance computation module, shown in Fig. 6, using subtractors to compute the difference between the corresponding features and cluster centroids which are used to compute the distance (d_A, d_B and d_C) using operator Vec_c (7) to produce the respective distances of the test set from each centroid. A comparator is used to determine the proximal cluster, denoted as 2-bit output ('00' - A, '01' - B and '10' - C).

Here, we have used three CORDIC operations in parallel for distance computation from each centroid (cf. Fig. 5) which could be achieved by reusing one CORDIC module for a sequential computation but at the expense of an increased computation time. A high speed design has been preferred in view of real-time detection. Using multiple CORDIC modules has its effects on the chip area and power and hence a trade-off with the computation time is necessary for an optimal design. In worst-case scenario, if all 30 features are selected, the distance computation from each of the three centroids would involve 30 CORDIC operations. Re-using a single CORDIC, incurs additional processing time, along with the overheads of a control logic. The feature extraction engine consumes approximately 1 nW of power [17] given the low frequency operations (@50 Hz) and therefore computation time has been given priority in this design.

The computation of the features and the MDC incurs a recursive formulation which leads to a computing loop that cannot be achieved with a pipelined CORDIC architecture whereas using an iterative CORDIC implementation would have its effect on the throughput. Hence, a unit latency design coalescing all iterations in a single computing stage (one clock cycle) is adopted here. We present an estimate of the hardware complexity in terms of the total full adder (FA) count, which provides an objective reflection of the underlying architecture. The MDC requires 3 subtractors and 3 CORDIC modules (Vec_c). A b-bit Ripple carry adder/subtracter (RCA) requires b full adders (FA), therefore we can consider 3b FA for the subtractors. The hardware resource for one iteration of an N-stage CORDIC rotation (considering a generalized word-length b) can be computed as 2Nb FA. This can be reused for multiple iterations (e.g. rms computation). Although the MDC requires 3 CORDIC modules in parallel, 2 modules used for feature extraction (1 module used for std, rms, entropy, dispersion, kurtosis and skewness as reported in [17] and 1 module for jerk metric which is independent of the rest of the features) can be reused for MDC. Hence, in total we require (2Nb + 3b) FA for the MDC implementation. For MDC implementation having 16-stages (N) and 24-bit (b) datapath, we require **840 FA**. It is important to note here that for the complexity analysis we did not consider the comparator, the counter logic and the multiplexers.

The complexity of an alternate architecture (without CORDIC) for MDC implementation can be estimated considering – a squaring unit, non-restoring iterative cellular square rooter (SQRT) [20], an accumulator (replacing one CORDIC module for the root mean square operation) and 3

subtractors. Hence, in total requiring 3 squarers, 3 SQRTs, 3 accumulators and 3 subtractors. For the sake of convenience, 2 squaring units can be considered as 1 multiplier and an accumulator block can be considered as a FA (registers associated with the accumulator are not considered, accounting only for arithmetic operations).

A conventional array multiplier (CAM) requires b(b - 2) FA, b half adders (HA) and b^2 AND gates. Considering, 2 HA as 1 FA and 4 AND gates as 1 FA (due to area and transistor count), the total gate count of a CAM can be deduced as $(1.25b^2 – 1.5b)$ FA. Hence, for 3 squaring units (1.5 CAM), $A_{mult} = 1.5(1.25b^2 – 1.5b)$ FA, where (A_*) represents the total number of FA's in each circuit. A b-bit SQRT requires $0.125 \times b(b + 6)$ FA and similar number of XOR gates. Therefore, the total FA count for 3 SQRTs, (considering 2 XOR gates as 1 FA) is $A_{SQRT} = (0.1875b^2 + 1.125b)$ FA. Lastly, $A_{add/sub} = 6b$ FA (3 subtractors + 3 accumulators) are required. Therefore, the total gate count for the MDC computation using an alternate architecture in terms of FA count is $(A_{mult} + A_{SQRT} + A_{add/sub}) = (2.0625b^2 + 4.875b)$ FA. Hence, for a 24-bit datapath, we require **1305 FA** which is more than the CORDIC based implementation.

It is worthwhile to recollect here that the CORDIC based feature extraction [17] engine requires 4110 FA (for 16-stage CORDIC and 24-bit datapath) whereas the non-CORDIC feature extractor requires 6828 FA. Hence, even if the circuit elements from the non-CORDIC feature extractor are re-used for its equivalent MDC implementation, a unified CORDIC based feature extraction engine and its equivalent MDC implementation will incur low-complexity and result in an optimized design.

Another important factor is the effect of normalization. The clusters are formed in a multi-dimensional feature space where the cluster analysis takes place on the features extracted from the training data. These features are linearly normalized with respect to their minimum and maximum value. Therefore, the cluster centroids are represented by the normalised values (i.e. in the numeric range of 0 - 1) of the selected features. However, during the testing phase, the relevant features are extracted from the corresponding sensor data using the feature extraction engine and used by the MDC lie in different numeric ranges compared to the respective centroids. Therefore, prior to computing the Euclidean distance, the centroids are un-normalized and used as inputs to the RTL module.

IV. Implementation And Evaluation

A. Verification

The architecture for feature extraction and MDC was coded using Verilog as HDL with a target ASIC implementation. It is important to note here that although the input data is 16-bits wide, the datapath width in the CORDIC-based feature extraction engine and the MDC module is 24-bits. In order to achieve the desired 16-bit accuracy a 22-bit word-length should be selected [21], according to the formulation $(N + Log_2N + 2)$ and atleast 16 iterations. Therefore, to obtain a high accuracy a 24-bit CORDIC was used for this implementation. The design was functionally verified using data of 4 healthy subjects and 4 stroke survivors. For each healthy subject, there were 80 test vectors (4 trials of 'making-cup-of-tea', having 20 movements in each trial). Similarly, for each stroke survivor there were 40 test vectors (2 trials of 'making-cup-of-tea'). The results using the accelerometer and the gyroscope data are shown in Tables IV-V for healthy subjects and stroke survivors. The software evaluation results (Matlab) [8] are presented for comparison.

TABLE IV
Recognition Sensitivities For Arm Movements Of Healthy Subjects

Subject	Features	Sensitivities (%)			Overall accuracy (%)
		A	B	C	
RTL Evaluation					
Accelerometer					
Subject1	11	100	100	75	94
Subject2	2	85	55	85	78
Subject3	7	90	90	90	90
Subject4	23	85	90	70	83
Gyroscope					
Subject1	10	50	80	100	70
Subject2	27	70	80	70	73
Subject3	18	80	85	90	84
Subject4	20	40	90	75	61
Software Evaluation [8]					
Accelerometer					
Subject1	11	100	100	100	100
Subject2	2	80	5	80	61
Subject3	7	95	100	90	95
Subject4	23	95	100	85	94
Gyroscope					
Subject1	10	93	90	100	94
Subject2	27	100	80	60	85
Subject3	18	90	90	100	93
Subject4	20	30	95	85	60

TABLE V
Recognition Sensitivities For Arm Movements Of Stroke Survivors

Subject	Features	Sensitivities (%)			Overall accuracy (%)
		A	B	C	
RTL Evaluation					
Accelerometer					
Subject1	19	70	80	100	80
Subject2	19	85	20	100	73
Subject3	21	80	90	30	70
Subject4	8	20	80	50	43
Gyroscope					
Subject1	8	80	60	80	75
Subject2	10	60	90	50	65
Subject3	24	80	20	70	63
Subject4	30	50	50	0	38
Software Evaluation [8]					
Accelerometer					
Subject1	19	80	90	100	88
Subject2	19	90	20	100	75
Subject3	21	95	100	20	78
Subject4	8	10	80	60	40
Gyroscope					
Subject1	8	90	50	100	83
Subject2	10	60	100	60	70
Subject3	24	85	30	80	70
Subject4	30	60	40	0	40

Stroke survivors 1 and 4 represent two extreme conditions (late and early stage of recovery after-stroke) as evaluated by respective clinicians. Overall, the results of the RTL simulation are on the lower side when compared to the software evaluation. The average difference in accuracy between RTL and software simulation is 1.25% and 11% using accelerometer and gyroscope data respectively for healthy subjects. Similarly, for stroke survivors, the average difference in accuracy is 3.75% and 5.5% for the two sensor types respectively. The difference in the results (decrease in individual movement sensitivities and the overall accuracy) of the RTL implementation and software can be attributed to the following factors:

(**1**) accumulation of truncation error, a common phenomenon in fixed-point arithmetic operations and occurs due to the implemented logic. Moreover, the software implementation (Matlab) presents the results in a 64-bit operating system whereas the CORDIC-based RTL module has a datapath width of 24-bits. Since, in this implementation, to achieve 16-bit accuracy, 16 iterations are used and hence this recursive CORDIC operation results in error accumulation to a higher degree. Hence for the MDC, where a data point is being classified based on a distance value, this accumulated error could result in misclassification. On the other hand, healthy subject2, requiring the computation of minimum number of features, viz. 2, is an exception as the overall accuracy achieved is higher with RTL. The accumulated error in this case (for computing the two required features – standard deviation and root mean square computed on the y-axis data [8]) could have created a bias for the distance computation of the test data w.r.t the centroids, thereby affecting the classification results yielding a higher accuracy. This effect is also observed to a less extent for the following subject/action/sensor combinations: healthy – 2/C/gyroscope; 4/A/gyroscope and stroke - 1/B/gyroscope; 3/C/accelerometer; 4/A/accelerometer and 4/B/gyroscope. As further illustration, variation of recognition accuracies w.r.t features for healthy subject2 with accelerometer data (cf. Fig. 7), shows that using more features (beyond 2) does not result in successful cluster formations (blank spaces) or improved accuracy.

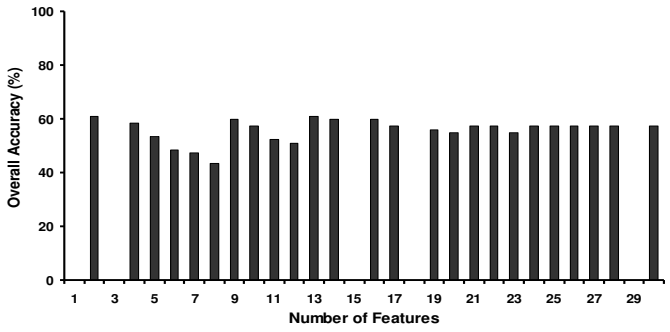

Fig. 7. Variation in accuracy with number of features for healthy Subject2 with accelerometer data during software evaluation [8].

(**2**) the difference of accuracy is further evident especially while computing a higher number of features. There are more number of test datasets for healthy subjects as compared to the stroke survivors and the high number of features computed (e.g. 30 for stroke survivor4 further contributes towards the mathematical error. It is evident from the feature computation engine [17] that the average error may become significant for the features particularly involving higher-order terms (e.g. kurtosis, skewness) even when the accuracy of the CORDIC itself is set high. Hence, to achieve higher accuracy, adjusting the datapath width for the MAC unit may be necessary depending on error tolerance of the application. A ranked list of the associated features for each subject chosen during cluster formation as a result of cross-validation is presented in [8]. The number of features selected for each subject represent the optimal number of top ranked features which resulted in successful cluster formation and highest cross-validation accuracies on the training dataset.

(**3**) in this implementation, a signal length of 256 data samples has been considered which can be represented on a dyadic scale and therefore any multiplication or division operation can be implemented through a shift. Hence, for testing with data already collected during the experimental protocol, an interpolation/extrapolation module in Matlab was implemented to pre-process the test data to restrict the sample size to 256 as opposed to the software implementation.

(**4**) lastly, in this design we have not filtered the raw sensor data (pre-processing step [8]), to keep the computations at a minimal level. Here, our focus was mainly on the implementation of the MDC and hence a filter block could be added to improve performance.

The achieved results, for both the healthy subjects (average accuracy of 86% and 72% with accelerometer and gyroscope respectively) and the stroke survivors (average accuracy of 67% and 60% with accelerometer and gyroscope respectively) can be considered favorable because the methodology was tested to detect activities performed in out-of-laboratory, semi-naturalistic scenario, having a significant degree of variability. The accuracy rates reported for the stroke survivors are acceptable, according to clinicians, since it provides a gross measure of impaired arm use. It is important to mention here that a misclassification of a performed movement may not have significant clinical impact because in this application (as opposed to other clinically critical remote monitoring applications, e.g. cardiovascular disease) the final decision on the rehabilitation measure and the corresponding prescription lies with the jurisdiction of the respective clinicians. This methodology could help to augment the clinical findings and provide a quantitative measure on the rehabilitation progress of patients over time outside the clinical environment. In view of the RTL simulation results, the conclusions drawn in [8] are still evident – 1) variability in data patterns due to poor repeatability and 2) considering more than one sensor type for specific cases can improve overall detection accuracy. This can be observed particularly for healthy subject2, where although the overall accuracy with accelerometer is 78% the sensitivity for Action B is low (55%), which is significantly improved when considering the gyroscope (80%). Similar trends are observed for healthy subject4 with Action A using gyroscope (40%), which can be

detected successfully when considering the accelerometer data (85%). Considering more than one sensor type could be beneficial for stroke survivors as can be seen for the following subject/action combinations – 2/C (gyroscope – 50%, accelerometer – 100%); 3/C (accelerometer – 30%, gyroscope – 70%). For subject 4, the overall accuracy with both sensors are not high, although it can be observed that Action A can be recognized by 60% (gyroscope), Action B by 80% and Action C by 50% (accelerometer). The low overall accuracy can be attributed to the fact that Subject 4 was at an early stage of rehabilitation and the impaired arm being tested was not the naturally dominant arm thereby resulting in poor repeatability.

The performance of the proposed clustering-MDC methodology was further compared against two well-known supervised learning algorithms – linear discriminant analysis (LDA) and support vector machines (SVM). LDA was chosen in view of its low-computational complexity and SVM known for producing high classification accuracy [22]. The average overall accuracy using LDA for 4 healthy subjects was 45% using accelerometer data and 53% using gyroscope data. Correspondingly for 4 stroke survivors, the average accuracy was 49% and 46% using accelerometer and gyroscope respectively. Similarly, using SVM, for the healthy subjects, the average accuracy was 54% and 68% using accelerometer and gyroscope respectively whereas for stroke survivors, the results were 55% and 50% using accelerometer and gyroscope data. Across all test cases, none of the subjects had all three movements classified with a sensitivity higher than 60% using either of the learning algorithms, thereby proving the effectiveness of our proposed methodology [8].

B. Synthesis and Layout

The design was synthesized using STMicroelectronics 130-nm technology library with a supply voltage of 1.08V and frequency of 50 Hz, where the synthesized design occupied an area of 242K (2-input NAND gate equivalent) and the dynamic power consumed was 5.3 nW. The design was also synthesized and functionally verified at a higher clock frequency of 20 MHz. The implementation of the feature extraction engine takes a maximum of 3n clock cycles [17] (where n is the number of input data samples), if it has to compute the all the 10 features. The MDC design takes (9n + 31) clock cycles in the worst case, considering it has to compute all the 30 features from the testing dataset and compute the Euclidean distance to the three cluster centroids. To estimate the total chip area a layout of the synthesized design was performed using the Cadence Encounter tool as shown in Fig. 8. The total area of the chip was estimated as 2.221 mm × 2.215 mm, having 25 signal pads and 8 power/ground pads. The 16-bit input/output port is used for: 1) i/p - three sensor data streams (AccX, AccY, AccZ or GyroX, GyroY, GyroZ) sequentially; 2) i/p - three centroids; 3) i/p - 30-bit feature-code split into - lower 16-bits followed by the higher 14-bits (padded with two zeroes) and 4) o/p - 2-bit (padded with 14 zeroes) signifying the predicted cluster label.

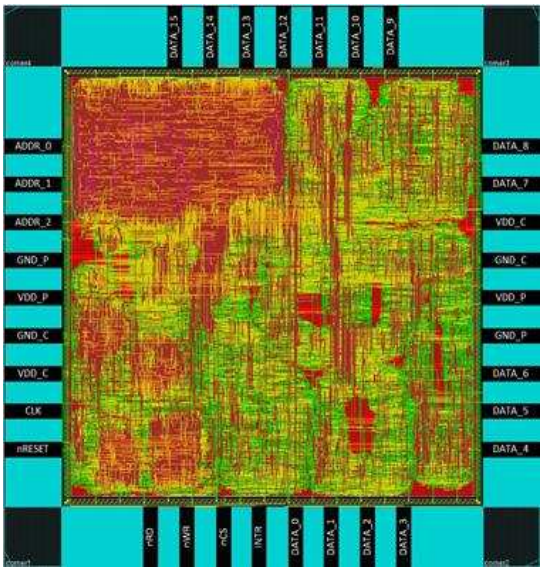

Fig. 8. Core chip layout with all pin assignments.

C. System Demonstrator using FPGA

The arm movement recognition framework (cf. section II) has been demonstrated as a prototype system using an Altera DE4 FPGA in conjunction with a wrist-worn inertial sensor. The hardware setup for real-time implementation is shown in Fig. 9 where the data from the sensors (tri-axial accelerometers) is transmitted to a host computer (i.e. PC) through Bluetooth. The raw data is converted to physical values and transmitted to the FPGA through RS232. The synthesized MDC HDL was integrated with RTL implementation of the RS232 receiver to complete the hardware functionality on the FPGA.

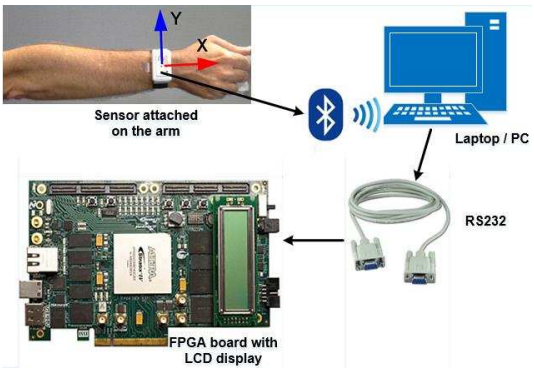

Fig. 9. FPGA-based demonstrator for real-time arm movement classification with movement data collected from the sensor attached to the arm.

The framework was validated with healthy subject2 performing one trial of 'making-cup-of-tea', where 18 out of the 20 movements were successfully detected. The three centroids are stored as binary data in the memory by using the megafunction in Quartus which allows the creation of a module that takes as input, memory initialization (MIF) files [23] and stores the data into the ROM of the FPGA. The feature code is sent using the synthesizable 'readmemb' function. The FPGA operates at a much higher frequency (780 KHz obtained through a clock-divider module) compared to the streaming sensor operating at 50 Hz. The sensor data was communicated to the host PC through Bluetooth using the application

ShimmerConnect [15]. The serial port control [24] was achieved through the .NET 4.5 framework and an application software (written in C#). The baud rate for transmitting the data from the PC to the FPGA was set to 4800 bits per second, where each set of data was of 64-bits (16-bits for each X, Y, Z axes and header code). The start of transmission was indicated by the header code which helps the receiver determine the correct axes value. A baud tick generator on the FPGA is used for interface synchronization which produces a pulse based on a counter logic. The classified arm movements are displayed on a 7-segment display in real-time (Action A – 1, B – 2 and C – 3). The synthesized design uses 40,753 logic units and 13,184 bits of memory (for storing the centroid and input tri-axial data). The prototype takes 515 clock cycles (≈ 0.6 ms) to produce the desired output since it computes two features for healthy subject2 (std takes 2n = 512 clock cycles) from the test data. For this demonstrator, we have used data from only one sensor type (i.e. accelerometers). However, this can be easily extended to incorporate the gyroscopes and the whole operation of 'feature extraction-MDC' can be independently performed on both sets of data to obtain the desired arm movement classification and these results can be analysed in line with the conclusions drawn in [8] and also in section IV-A where considering more than one sensor type has been advocated to ascertain impaired arm usage and rehabilitation progress.

V. Related Work

Real-time AR in body sensor networks is a challenging task and energy efficiency has received particular attention in recent years from the pervasive computing research community for ways to extend the battery life of sensors aimed at long-term monitoring. With the advent of context-aware processing, energy efficient processing on sensor nodes and mobile devices has taken precedence. A few recent papers [25-27] have discussed the need for reducing energy incurred on communication, with [27] showing the importance of on-node sensor processing over an off-node scheme saving up to 40% of energy trading off accuracy. Some of the recent online AR methods have looked into this aspect by processing on the sensors (e.g. low-power MSP430 microcontroller) or mobile phones (e.g. android) [27]. Another recent work [28] takes a hierarchical approach whereby they recognize hand gestures on the accelerometer sensor node using a Java based simulator but use this information to classify high-level activities on a mobile device by transmitting data through a wireless link. Apart from reducing communication (through on-node data processing and advocating light-weight algorithms), the focus has been on issues such as deactivation of power hungry sensors [29] (e.g. gyroscopes) and adaptive sampling rate [30]. Hence, to the best of our knowledge, this is the first work which has focused on an optimized, low-complexity algorithm-to-architecture mapping aimed towards a hardware/accelerator based design to be used within resource constrained senor nodes. Further energy saving design optimisations such as dynamic power management (for e.g. shutting down feature extraction engine during MDC) and clock gating techniques can be incorporated to enhance the proposed low-complexity implementation.

VI. Discussion

In this paper, we have presented the architecture and implementation of a low-complexity framework for arm movement classification in an out-of-laboratory environment using body-worn inertial sensors. A completely personalized approach has been presented and the results obtained have been encouraging and show that these particular arm movements can be reliably detected with stroke survivors exhibiting moderate levels of involuntary tremor in their movements. The framework was further demonstrated as a proof-of-concept real-time arm movement recognition system.

One of the key features in such a system is the need for adaptability which caters to the change in movement patterns over time pertaining to each patient thereby reflecting the improvement in their motor functionality as a result of the undergoing rehabilitation protocol. The demonstrated methodology can detect the change in movement patterns over a longitudinal scale by two means: 1) with decreasing movement recognition rates over time – due to the differing patterns of the daily life movements with respect to the pre-computed cluster centroids in the selected feature space and 2) clinical intervention – clinicians observe a considerable change in the movements performed by the patients in comparison to their previous assessment (the time of obtaining the training data for the clusters). In such circumstances, the patient's training data would be collected periodically and the cluster centroids and the associated features (new selected feature set) can be re-computed to reflect the changing movement patterns. The new cluster centroids and feature set will be subject-specific due to the inter-subject variability inherent within movement profiles, variation in the rehabilitation profile and the associated functional ability of each individual subject. This information could be further used by the MDC to recognize movements performed in daily life. Hence, we plan to carry out a longitudinal study in the near future to demonstrate the methodology for indicating rehabilitation progress.

In view of the designed architecture, there are a few fundamental factors which can be considered in future designs. First, the size of the register bank to store the incoming data samples from the sensors has been fixed at 256, representing 5 seconds of kinematic data (sensor streaming @50 Hz). This time duration is suitable for the healthy subjects for the completion of the elementary arm movements (actions) chosen for the experimental protocol. For patients, depending on the level of dexterity, the time taken to perform the movements might be more especially when they are in their initial stage of rehabilitation. The next available window size, in view of representing it in dyadic scale is 512 implying 10 seconds and would suit the requirements of patients needing more time to complete the actions. An alternate approach would be to reduce the sampling frequency in the range of 20~25 Hz which has also been considered to be suitable in human activity recognition [9][10]. Second, here we consider the Euclidean distance over the Mahalanobis distance [8] for the MDC as a proof-of-concept implementation since the later increases the complexity involved in computing the covariance matrix. Third, a fundamental exploration in terms of error accumulation and

propagation needs to be carried out and accordingly the datapath adjustment for ASIC implementation needs to be done in view of the target accuracy.

This design can be implemented as an ASIC chip and embedded on a sensor platform along with other processing components like A/D converter, filtering circuit, memory, power source, to be used for real-time AR. An ASIC would provide leverage in terms of area and power as compared to state-of-the-art microcontroller/mobile-platform based designs, aiding the development of a point-of-care monitoring system. This methodology could be extended for lower limb monitoring and used with patients suffering from other neurodegenerative disorders exhibiting movement profiles which are less fluidic in nature. Real-time detection of arm movements can be useful in a wide array of applications in the field of sports, human computer interaction or other treatments of arm dexterity. Therefore, the developed system can be used to track movements of required body segments in these respective fields outside a controlled environment.

REFERENCES

[1] F.E. Martınez-Pérez et al., "Activity inference for ambient intelligence through handling artifacts in a healthcare environment," Sensors, vol. 12, no. 1, pp. 1072–1099, 2012.

[2] B. Najafi et al., "Ambulatory system for human motion analysis using a kinematic sensor: monitoring of daily physical activity in the elderly," Biomedical Engineering, IEEE Transactions on, vol. 50, no. 6, pp. 711–723, 2003.

[3] C. Zhu and W. Sheng, "Motion-and location-based online human daily activity recognition," Pervasive and Mobile Computing, vol. 7, no. 2, pp. 256–269, 2011.

[4] L. Chen et al.," Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on, vol. 42, no. 6, pp. 790–808, 2012.

[5] J. Parkka et al., "Activity classification using realistic data from wearable sensors," Information Technology in Biomedicine, IEEE Transactions on, vol. 10, no. 1, pp. 119–128, 2006.

[6] K. Altun et al., "Comparative study on classifying human activities with miniature inertial and magnetic sensors," Pattern Recognition, vol. 43, no. 10, pp. 3605–3620, 2010.

[7] K. Maharatna et al., "Towards the development of next-generation remote healthcare system: Some practical considerations," in Circuits and Systems (ISCAS), 2012 IEEE International Symposium on, 2012, pp. 1–4.

[8] D. Biswas et al., "Recognizing upper limb movements with wrist worn inertial sensors using k-means clustering classification," Human Movement Science, vol. 40, pp. 59–76, 2015.

[9] M. Ermes et al., "Detection of daily activities and sports with wearable sensors in controlled and uncontrolled conditions," Information Technology in Biomedicine, IEEE Transactions on, vol. 12, no. 1, pp. 20–26, 2008.

[10] S. Chernbumroong et al., "Elderly activities recognition and classification for applications in assisted living," Expert Systems with Applications, vol. 40, no. 5, pp. 1662–1674, 2013.

[11] C. S. Lányi et al., "Developing serious game for victims of stroke," in 9th International Conference on Disability, Virtual Reality and Assoc. Technologies, 2012, pp. 503–506.

[12] S. Ma et al., "Overcoming the information overload problem in a multiform feedback-based virtual reality system for hand motion rehabilitation: healthy subject case study," Virtual Reality, vol. 16, no. 4, pp. 325–334, 2012.

[13] D. Biswas et al., "On the data analysis for classification of elementary upper limb movements," Biomedical Engineering Letters, vol. 4, no. 4, pp. 403-413, 2014.

[14] L. Bao and S. S. Intille, "Activity recognition from user-annotated acceleration data," in Pervasive computing, Springer, 2004, pp. 1–17.

[15] A. Burns et al., "SHIMMER™-A wireless sensor platform for noninvasive biomedical research," Sensors Journal, IEEE, vol. 10, no. 9, pp. 1527–1534, 2010.

[16] C. Kendell and E. D. Lemaire, "Effect of mobility devices on orientation sensors that contain magnetometers," J. Rehabil. Res. Dev, vol. 46, no. 7, pp. 957–962, 2009.

[17] D. Biswas and K. Maharatna, "A CORDIC-Based Low-Power Statistical Feature Computation Engine for WSN Applications," Circuits, Systems, and Signal Processing, vol. 34, no. 12, pp. 4011–4028, 2015.

[18] P. K. Meher et al., "50 years of CORDIC: Algorithms, architectures, and applications," Circuits and Systems I: Regular Papers, IEEE Transactions on, vol. 56, no. 9, pp. 1893–1907, 2009.

[19] S. Patel et al., "A novel approach to monitor rehabilitation outcomes in stroke survivors using wearable technology," Proceedings of the IEEE, vol. 98, no. 3, pp. 450–461, 2010.

[20] K.N. Vijeyakumar et al., "FPGA implementation of low power high speed square root circuits", in IEEE International Conference on Computational Intelligence & Computing Research (ICCIC), 18–20 Dec 2012, pp. 1–5.

[21] L. Vachhani et al., "Efficient CORDIC algorithms and architectures for low area and high throughput implementation," Circuits and Systems II: Express Briefs, IEEE Transactions on, vol. 56, no. 1, pp. 61–65, 2009.

[22] T. Chen, et al., "Design of a Low-Power On-Body ECG Classifier for Remote Cardiovascular Monitoring Systems" in IEEE Journal on Emerging and Selected Topics in Circuits and Systems, vol. 3, no. 1, pp. 75-85, March 2013.

[23] "Memory Initialization File." [Online]. Available: http://quartushelp.altera.com/15.0/mergedProjects/reference/glossary/def_mif.htm. [Accessed: 12–06-2015].

[24] "SerialPort Class." [Online]. Available: http://msdn.microsoft.com/en-us/library/system.io.ports.serialport%28v=vs.110%29.aspx. [Accessed: 06-2015].

[25] O.D. Lara and M.A. Labrador, "A Survey on Human Activity Recognition using Wearable Sensors" Surveys & Tutorials, IEEE Communications, vol. 15, no. 3, pp. 1192-1209, 2013.

[26] T.Rault et al., "A survey of energy-efficient context recognition systems using wearable sensors for healthcare applications", Pervasive and Mobile Computing, Jun. 2006, DOI: http://dx.doi.org/10.1016/j.pmcj.2016.08.003.

[27] N. Wang et al., "Energy and Accuracy Trade-Offs in Accelerometry-Based Activity Recognition", in 22nd International Conference on Computer Communications and Networks, Nassau, Bahamas, 1–6, 2013.

[28] L. Wang et al., "A hierarchical approach to real-time activity recognition in body sensor netwroks", Pervasive and Mobile Computing, vol. 8, no. 1, pp. 115-130, Feb. 2012.

[29] K. Lorincz et al.,"Mercury: a wearable sensor network platform for high-fidelity motion analysis", in 7th ACM Conference on Embedded Networked Sensor Systems, Berkeley, California, 183–196, 2009.

[30] X. Qi et al., "AdaSense: Adapting Sampling Rates for Activity Recognition in Body Sensor Networks", in 19th IEEE Real-Time and Embedded Technology and Applications Symposium, Philadelphia, PA, 163–172, 2013.

Novelty Statement – The work in this paper proposes a novel low-complexity architecture, designed for arm movement detection, based on the algorithm proposed in [8]. For the design, implementation and the FPGA-based prototype demonstrator, the CORDIC-based feature extraction engine proposed in [17] has been re-used. An FPGA-based demonstrator was published in our previous work (10.1109/ISCAS.2015.7168746), however it is based on a different algorithm, using a different board. Hence, as highlighted, the fundamental contribution – offline-online design framework and system demonstrator for movement detection using the MDC architecture is completely novel and unreported.

Dwaipayan Biswas obtained his MSc and PhD from the University of Southampton, UK in 2011 and 2015 respectively. He is presently a Research fellow working on biomedical signal processing and embedded systems at University of Southampton.

Koushik Maharatna received the Ph.D. degree from Jadavpur University, Calcutta, India, in 2002. From 2000 to 2003, he was a Research Scientist in IHP, Frankfurt (Oder), Germany. He is with ECS, University of Southampton, U.K. since 2006, currently as a Professor.

Goran Panic received his Ph.D. degree in 2014 from the Brandenburg University of Technology Cottbus-Senftenberg, Germany. He is working as a scientist in the Systems Department of IHP, Frankfurt (Oder), Germany.

Evangelos B. Mazomenos received the Ph.D. degree from the University of Southampton, U.K., in 2012. He is currently a Research Associate at the Centre for Medical Image Computing, University College London.

Josy Achner is an Occupational Therapist and head of occupational therapy of the neurologic department of the Brandenburg Klinik, Berlin, working since 2002.

Jasmin Klemke is an Occupational Therapist at the neurologic department of the Brandenburg Klinik, Berlin, working since 2006.

Michael Jöbges is a Consultant Neurologist and head of the neurologic department of the Brandenburg Klinik, Berlin. In 1996 he received a M.D. on a neurophysiological topic.

Dr. Steffen Ortmann received his diploma in computer science in 2007 and his PhD by scholarship in 2010 from the technical University of Cottbus, Germany. Since 2005 he is active in the sensor network research group of IHP.