An Optimized Embedded Target Detection System Using Acoustic and Seismic Sensors

Kyunghun Lee*, Benjamin S. Riggan†, and Shuvra S. Bhattacharyya*‡
*Department of Electrical and Computer Engineering, University of Maryland, College Park, MD, USA
†U.S. Army Research Laboratory, Adelphi, MD, USA
‡Department of Pervasive Computing, Tampere University of Technology, Tampere, Finland
{leekh3, ssb}@umd.edu {benjamin.s.riggan.civ}@mail.mil

Abstract—**Detection of targets using low power embedded devices has important applications in border security and surveillance. In this paper, we build on recent algorithmic advances in sensor fusion, and present the design and implementation of a novel, multi-mode embedded signal processing system for detection of people and vehicles using acoustic and seismic sensors. Here, by "multi-mode", we mean that the system has available a complementary set of configurations that are optimized for different trade-offs. The multi-mode capability delivered by the proposed system is useful to supporting long lifetime (long term, energy-efficient "standby" operation), while also supporting optimized accuracy during critical time periods (e.g., when a potential threat is detected). In our target detection system, we apply a strategically-configured suite of single- and dual-modality signal processing techniques together with dataflow-based design optimization for energy-efficient, real-time implementation. Through experiments using a Raspberry Pi platform, we demonstrate the capability of our target detection system to provide efficient operational trade-offs among detection accuracy, energy efficiency, and processing speed.**

I. INTRODUCTION

Sensor networks for detection of targets such as people and vehicles are of great relevance in defense and security applications. In such networks, use of non-image sensors, such as acoustic and seismic sensors, are of interest in part because of their power efficiency compared to image sensors. Various studies have been focused on development and enhancement of acoustic and seismic signal processing algorithms for high target detection accuracy.

For large-scale deployment of such networks, it is critical to provide methods for their cost- and energy-efficient realization, while providing high detection accuracy and low false alarm rate. In support of these objectives, a significant body of research has focused on the development of novel algorithms for fusion, target detection, and classification from acoustic and seismic signals (e.g., see [1], [2], [3]). In this paper, we develop design optimization methods that are complementary to this body of prior algorithm-oriented work. In particular, we focus on system design and implementation issues that are important for delivering the accuracy offered by relevant fusion/detection algorithms along with energy-efficient and resource-constrained execution capability on low cost sensor node platforms.

To balance objectives of low average energy consumption (streamlined standby operation) and optimized accuracy during times of critical operation (e.g., when potential threats are actively being monitored), we develop a novel, multi-mode system design that provides alternative configurations to support optimized trade-offs for these standby and critical operation scenarios. Transitions between these modes can then be triggered based on specific application requirements — for example, transitions may be manually-driven by personnel operating a monitoring station or they may be triggered automatically using some kind of finite state machine logic.

In this work, we apply a dataflow-based methodology for model-based implementation and design optimization of the proposed multi-mode target detection system. Dataflow methods are widely used in many areas of signal processing system design (e.g., see [4]). In addition to supporting design optimization, our application of dataflow methods helps to promote reliability and efficiency of the developed implementation, as well support the retargetability of the system to other types of sensor node platforms. We provide extensive experimental results to motivate the use of alternative modes in our dataflow-based target detection system design, and to quantify the useful range of operational trade-offs provided by the different modes.

II. RELATED WORK

Various algorithms have been proposed that are relevant to *person-and-vehicle detection* (*PVD*) using energy-efficient sensing modalities, including acoustic and seismic modalities. For example, Dibazar et al. develop neural networks that operate on seismic signals from footsteps and vehicles [3]. Damarla and Kaplan develop a decision-level fusion architecture for tracking groups of people using acoustic and seismic signal processing [2]. Ben Salem et al. present an adaptive target detection system that employs mobile devices as sensor node platforms, and applies different acoustic signal processing techniques for different signal-to-noise ratio conditions, and energy consumption constraints [5]. Our work in this paper differs from these prior works in that we simultaneously handle (1) multiple sensing modalities (acoustic and seismic); (2) both decision and feature level fusion for improved accuracy; and (3) design optimization for energy- and resource-constrained embedded implementation.

In this work, we build on our recent algorithmic investigation of PVD using acoustic and seismic signals [6]. In this investigation, we introduced an adaptation to PVD of sensor fusion based on Dempster-Shafer Theory (DST) [7], [8], and we also introduced a PVD algorithm, called Accumulation of Local Feature-level Fusion Scores (ALFFS). ALFFS extracts cepstral features, and applies an accumulative, feature-level fusion approach. Our work in this paper employs the algorithms developed in [6] as a starting point, and addresses system design and implementation challenges that are critical to practical deployment of the algorithms. The system design and implementation-oriented focus of this work is significantly different from that of [6], which is focused on algorithmic aspects.

III. System Design

The input to our PVD system consists of multi-modal sensor streams that arrive from a pair of sensors — one acoustic sensor and one seismic sensor. Here, we use the term *multi-modal* to represent multiple sensing modalities, while *multi-mode* refers to the incorporation of alternative operational modes with complementary trade-offs. The PVD system operates on fixed-length frames of signal samples. The number of samples per frame is determined by two system parameters — the frame duration t_f, and sample rate N_r.

The output of our PVD system is a sequence $(y_1, y_2, \ldots)$ of target classification results, where each $y_i \in \{P, V, N\}$ provides the derived detection result for the ith frame. Here, P, V, and N correspond, respectively, to detection of a person, a vehicle, or noise (the absence of any person or vehicle), respectively. This is referred to as a *multi-class* classification system since the system must discriminate across more than two classes.

Both the DST-based and ALFFS approaches employed in our PVD system employ support vector machine (SVM) subsystems as core building blocks for the classification process. SVMs are widely used in machine learning applications due to their robustness and classification performance (e.g., see [9]). In each mode of our PVD system, we apply multiple, *Binary SVM Classifiers* (*BSCs*) along with voting logic to perform the targeted multiclass classification task. In particular, we employ BSCs that are configured to perform P vs. N, P vs. V, and V vs. N classification. These BSCs are embedded in different ways into different PVD architectures that are associated with the alternative modes. These different embeddings provide an efficient range of operational trade-offs, which we will demonstrate quantitatively in Section IV.

Our PVD system involves four modes of operation, which we refer to as the *acoustic* mode, *seismic* mode, *DST fusion* mode, and *ALFFS* mode. These modes are denoted as μ_{ac}, μ_{sei}, μ_{dst}, and μ_{alf}, respectively. The modes provide progressively higher levels of accuracy, while the latter two modes — which involve dual-modality processing — consume higher levels of energy and require longer run-time. The first mode μ_{ac} provides lower accuracy compared to all of the other modes, while providing no significant benefit, as evaluated on our target platform, in terms of the run-time or energy efficiency. Thus, μ_{ac} is disabled in the final implementation. However, the mode is useful to have available for experimentation purposes, and for enhanced configurability. For example, the accuracy of μ_{ac} may improve significantly if the design is (1) retargeted to a platform that employs a higher quality acoustic sensor or (2) adapted to an application in which acoustic signals provide better discrimination potential compared to seismic signals (e.g., speech detection or recognition).

In Section III-A through Section III-C, we present dataflow graph specifications for the different modes in our PVD system. The presentation here is focused on highlighting relevant aspects of the embedded software architecture. For details of the underlying algorithms, we refer the reader to [6].

Each dataflow graph, including all of its encapsulated actors (dataflow graph vertices) and connections (graph edges), is implemented using the LIghtweight Dataflow Environment (LIDE) [10]. LIDE is a software tool that facilitates design and implementation of embedded signal processing systems. More specifically, our dataflow graph implementations employ LIDE-C, which is the integration of LIDE with the C programming language. We have used LIDE-C for design and implementation of the entire PVD system, including the dataflow graphs for the different modes.

Actors in LIDE-C, as in other dataflow tools, execute in terms of discrete units of execution, which we refer to as *firings*. As the enclosing signal processing application operates on successive samples or frames of data, each dataflow actor in general executes iteratively through a sequence of successive firings. The operation of an actor is often explained in terms of the computation it performs in a single firing.

A. Single-Modality Operation

Figure 1 illustrates the dataflow graph employed in our design and implementation of the two single-modality modes, μ_{ac} and μ_{sei}. These two modes use exactly the same actors and edges. The key difference in the dataflow graph configurations between the two modes is that different sets of parameters are employed by the SVMs within the SVM Bank actor.

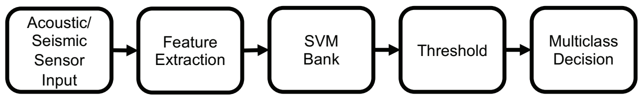

Fig. 1. Dataflow graph for single-modality PVD.

The Acoustic/Seismic Sensor Input ("Sensor Input") actor injects digitized samples that are produced by an analog/digital (A/D) converter that is interfaced to the relevant sensor device — that is, interfaced to the acoustic sensor in μ_{ac} and the seismic sensor in μ_{sei}. In our implementation of this sensor input actor, we employ C-based Linux libraries that are developed for GNU-Linux platforms, including x86/64 Linux and ARM-based Linux platforms such as the Odroid and Raspberry PI platforms [11].

The Feature Extraction actor in Figure 1 applies cepstral analysis to extract features from the digitized samples arriving from the sensor source. For FFT computation, we employ a LIDE-C wrapper around an optimized module from the FFTW library [12]. On each firing, the Feature Extraction actor consumes N_s samples, corresponding to a single frame of sensor data, and produces N_c cepstral coefficients, where N_c is a parameter of the actor, and N_s can be derived as the product of the system frame duration t_f and sample rate N_r. The generated cepstral coefficients are subsequently employed (in the downstream portion of the dataflow graph) as the features of the input frame.

The cepstral features extracted from the Feature Extraction actor are sent as input to the SVM Bank actor, as illustrated in Figure 1. The SVM Bank actor, like the

Feature Extraction actor, is an important actor in all of the different modes of our PVD system. The SVM Bank actor applies three different BSCs, which we denote by β_{pv}, β_{pn}, β_{vn}. These BSCs are trained, at design time (offline), to discriminate respectively between P vs. V, P vs. N, and V vs. N. Recall that P, V, and N, respectively represent the decision classes "person", "vehicle", and "noise". In a given firing of the SVM Bank actor, each of the three encapsulated BSCs produces a real-valued score ϕ. The sign (negative or positive) of ϕ indicates the predicted decision class (between the two candidate classes), and the absolute value of ϕ provides an indicator of the strength or "confidence" of the prediction. On each firing, the SVM Bank actor produces as output three real-valued, scalar outputs, which correspond to the scores generated by the three encapsulated BSCs.

The Threshold actor in the dataflow graph consumes a block of real values $x_1, x_2, \ldots, x_M$, and simply applies a threshold τ to each one to produce a binary output. In all of our applications of the Threshold actor in the PVD system, the block size M is equal to 3, and each input block corresponds to scalar scores associated with P vs. V, P vs. N, and V vs. N discrimination. The threshold τ, however, is not identical in all PVD modes. In the case of Figure 1, we apply $\tau = 0$ for both modalities. The output of the Threshold actor is in general a block $z_1, z_2, \ldots, z_M$ of binary values, where for each i, $z_i = 0$ if $x_i < \tau$, and $z_i = 1$ if $x_i \geq \tau$. In the case of μ_{ac} and μ_{sei}, these binary values correspond to prediction results for each of the BSCs employed within the SVM Bank.

The Multiclass Decision actor in Figure 1 is used to combine blocks of binary prediction results into a corresponding stream of multiclass decision results. Again, we use block size $M = 3$. The input block consists of a triplet of binary decisions, (z_1, z_2, z_3), corresponding to β_{pv}, β_{pn}, and β_{vn}, respectively. For example, $z_1 = 0$ if the BSC β_{pv} has generated a prediction of P for the most recent signal frame, and $z_1 = 1$ if β_{pv} has predicted V. The Multiclass Decision actor applies a simple voting rule to generate a single classification result from within the set $\{P, V, N\}$. In case of a tie (all three input predictions are different), the actor produces N as the classification result.

B. DST-based Fusion Mode

Figure 2 shows the dataflow graph for the dual-modality mode μ_{dst}. In this graph, the Acoustic Sensor Input and Seismic Sensor Input actors can be viewed as multiple concurrent instantiations of the single sensor input actor in Figure 1. These actors inject digitized data acquired from both sensing modalities for processing and fusion in the downstream portion of the dataflow graph.

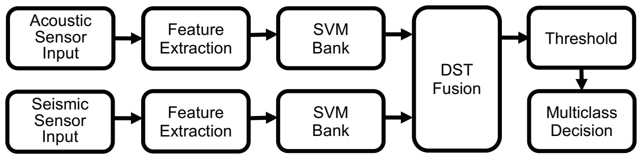

Fig. 2. Dataflow graph for the dual-modality mode μ_{dst}.

The Feature Extraction actors in the μ_{dst} dataflow graph represent multiple instantiations — one for each sensing modality — of the actor with the same name in Figure 1. The cepstral coefficients extracted by each of these two Feature Extraction actors are processed by separate SVM Bank actors. The SVM Bank actors in Figure 2 are identical to the corresponding actor in Figure 1; however, they are configured differently at design time. Each of the BSCs encapsulated within the SVM Bank in the upper (acoustic) branch is trained for the associated binary classification task based on acoustic data, and similarly, the training for the lower SVM Bank actor is based on seismic data. In other words, both trained versions of the SVM Bank actor in Figure 1 are instantiated concurrently in Figure 2.

The DST Fusion actor in Figure 2 is the only "new" actor in this dual-modality dataflow graph compared to Figure 1. This actor applies a dual-modality fusion algorithm based on Dempster-Shafer Theory, as mentioned in Section II. For complete details on this algorithm, we refer the reader to [6]. In a given firing, the DST Fusion actor takes as input two frames of cepstral coefficients $a(1), a(2), \ldots, a(N_c)$ and $s(1), s(2), \ldots, s(N_c)$, which are extracted as features from the corresponding acoustic and seismic input signal frames. From the results of its underlying fusion algorithm, the actor then produces (similar to the SVM Bank actor in Figure 1) three real-valued, scalar outputs, which represent binary classification scores for discrimination between P vs. V, P vs. N, and V vs. N, respectively. In the case of the DST Fusion actor, each output score σ is a non-negative real number with $\sigma < 1$ corresponding to one decision class and $\sigma > 1$ corresponding to the other.

The Threshold actor in Figure 2 applies block size $M = 3$ and threshold $\tau = 1$ to produce, on each firing, a triple of binary prediction results. This triple is then processed by the Multiclass Decision actor to produce a single PVD classification result from the set $\{P, V, N\}$.

C. ALFFS Mode

Figure 3 shows the dataflow graph for the second dual-modality mode, which is μ_{alf}. As with the μ_{dst} subsystem, input samples are injected into this graph using the Acoustic Sensor Input and Seismic Sensor Input actors. Overlapping windows of samples from each input frame are then processed using the two instances of the Feature Extraction actor shown in Figure 3.

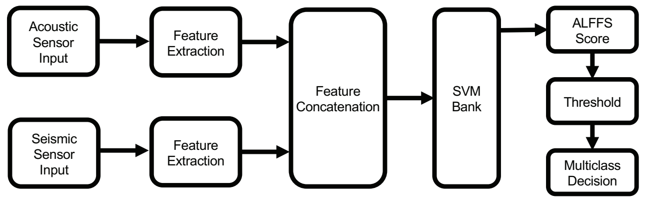

Fig. 3. Dataflow graph for dual-modality PVD using ALFFS.

In μ_{alf}, each Feature Extraction actor is configured to process overlapping windows of input data through appropriate setting of two actor parameters, called the *threshold parameter* and *consumption parameter*. These parameters, denoted respectively by *thr* and *cns*, control the flow of data from the first-in, first-out (FIFO) buffer that corresponds to the input edge e_{in} of the actor. The threshold parameter specifies the number of samples that must be present on e_{in} before the actor can be fired, and the cns parameter specifies how many tokens are consumed (removed) from e_{in} on each firing. This results in a processing pattern whereby successive firings of each Feature Extraction actor process overlapping windows of data, where each window contains thr samples, and adjacent windows contain $(thr - cns)$ samples of overlapping data.

This use of distinct threshold and consumption parameters is closely related to a similar distinction that is part of the computation graph model [13]. The parameters can be implemented efficiently through a straightforward adaptation of LIDE where the *enable function* (used to determine whether a LIDE actor can be fired) is controlled by the threshold parameter and the *invoke function* (which executes an actor firing) consumes samples based on the consumption parameter. This type of threshold- and consumption-parameterized feature extraction provides efficient sliding window operation.

In the Feature Extraction configurations used in Figure 1 and Figure 2, $thr = cns$, and the window size in the enclosing dataflow graphs is effectively equal to the input frame size (i.e., multi-window processing is not employed). The μ_{alf} mode is the only PVD system mode that requires $thr \neq cns$, due to the windowed behavior of the underlying ALFFS algorithm [6].

In particular, in ALFFS, feature extraction is performed on N_w overlapping windows of a given input frame. The N_c cepstral coefficients extracted from each window are processed by the SVM Bank actor, as shown in Figure 3. The coefficients extracted from each pair of corresponding windows associated with the two sensing modalities are concatenated by the Feature Concatenation actor before arriving as input to the SVM Bank actor. SVM Bank then fires N_w times, and processes a $(2 \times N_c)$-element feature vector on each firing. This results in a total of $3 \times N_w$ values that arrive at the input of the ALFFS Score actor. These values represent the collection of binary prediction triples (P vs. V, P vs. N, and V vs. N) for all of the windows. Corresponding elements of the triples are added in the ALFFS actor, and the resulting sums are analyzed, as illustrated in Figure 3, to derive the final multiclass classification result for the multi-modal input frame. For further details on the algorithm that underlies the ALFFS mode, we refer the reader to [6].

IV. Experiments

In this section, we present an experimental evaluation of the multi-mode PVD system design presented in Section III. We compare the run-time and energy consumption performance of the different system modes using the same input data. For this purpose, pre-collected input frames (frames of acoustic and seismic signals) are stored within flash memory on the targeted embedded platform. The "sensor input" actors in Figure 1, Figure 2 and Figure 3 are configured in these experiments to read the pre-collected data from flash memory and inject it into the associated dataflow graphs for processing. The sensor input actors in practice obtain the data directly from the sensors. However, the purpose in these experiments is to demonstrate how the proposed methods provide optimized trade-offs for improving processing capabilities at the network edge.

The pre-collected data used in these experiments is obtained from datasets that were collected from acoustic and seismic sensors on Spesutie Island at the Aberdeen Proving Grounds in Maryland, USA. Further details about these datasets can be found in [14]. We employed a dataset that consists of 1000 data frames, where each frame contains 6 seconds of acoustic and seismic data. 500 of these frames were used in our experiments for training, and the other 500 frames were used for testing.

The target platform that we used in our experiments is the Raspberry Pi 3 Model B, which is equipped with 1GB RAM, a 4x ARM Cortex A53 CPU, and a Broadcom VideoCore IV GPU. The operating system used was Raspbian 4.4. The device we used for measuring power consumption is the Tektronix Keithley Series 2280 Precision Measurement DC Power Supply.

The energy consumption of the sensors is not included in the values reported in this section. This is because separate energy sources may be used for the sensors, and our intent is to focus in the paper on trade-offs between *processing efficiency* (energy and speed) versus *accuracy* for alternative signal processing techniques. However, the design methodology applied in this paper can be readily adapted to develop a multi-mode system whose modes are selected in a manner that takes into account the energy efficiency of the sensors. Although developing such adaptations may be useful, it is beyond the scope of this work but may be addressed in the future with large scale sensor networks.

Various parameter values employed in our experiments are summarized in Table I.

TABLE I. Parameter values used in our experiments.

Description	symbol	value	units
Frame duration	t_f	6	seconds
Sample rate	N_r	4096	Hz
Number of cepstral coefficients	N_c	50	
Number of windows (ALFFS)	N_w	50	
Window overlap ratio (ALFFS)	w_r	0.4	

Table II shows the measured accuracy of the four different modes in our PVD system. The accuracy is measured as (z_c/F), where z_c is the number of correct classifications, and F is the number of frames of input data (i.e., the total number of classification events). From Table II, we see significant variation in accuracy among the modes, with a significant gap from each lower accuracy mode to the next higher accuracy mode. As expected, the dual-modality modes have higher accuracy compared to the single-modality ones.

TABLE II. Accuracy comparison (%).

	Acoustic	Seismic	DST Fusion	ALFFS
Accuracy (%)	67.94	76.55	81.56	98.40

Table III summarizes measurements of power consumption P (Watts), run-time R (seconds per data frame), and energy consumption $E = R \times P$ (Joules per data frame). Here, a "data frame" corresponds to a single frame of acoustic or seismic input data for the single-modality modes. For the dual-modality modes, a data frame encapsulates an acoustic input frame together with its corresponding seismic input frame. The reported power values are derived by measuring the power consumption in the associated modes, and subtracting from these measurements the baseline power consumption (i.e., the power consumed when the processing platform is idle). This gives an estimate of the power consumption that is attributable to the processing requirements of each mode.

TABLE III. POWER CONSUMPTION, RUN-TIME, AND ENERGY EFFICIENCY COMPARISON.

	Current (A)	Power (W)	Run-time (sec)	Energy per frame (J)
Acoustic	0.14359	0.71795	0.142834	0.102548
Seismic	0.14130	0.70650	0.147289	0.104060
DST Fusion	0.15652	0.78261	0.170359	0.133323
ALFFS	0.15065	0.75325	0.250434	0.188639

While the ALFFS mode provides superior accuracy, this enhanced discrimination capability comes at the expense of higher power consumption, and longer execution time. These costs in turn combine to increase energy consumption, leading to faster battery drain. Conversely, the single-modality modes are more energy efficient, but are not as accurate as the modes that employ fusion techniques. This loss in accuracy means that there will be a higher rate of missed events or false event detections. The run-time, accuracy, and energy consumption for DST Fusion provide an intermediate trade-off between the single-modality modes and ALFFS.

These results demonstrate that each of the four modes provides a distinct, Pareto-optimal (non-dominated) design point among the four design points represented by the PVD system modes. Given a multidimensional design evaluation space involving $N > 2$ metrics, a design point p_1 is said to *dominate* another point p_2 if p_1 is better than or equal to p_2 in terms of all of the N relevant metrics, and p_1 is better than p_2 in terms of at least one metric. For example, from the data shown in Table II and Table III, the acoustic mode (design point) is not dominated by any of the other modes (e.g., because it is better than all other modes in terms of energy consumption per processed input frame). On the other hand, if accuracy and power consumption were the only metrics considered in the design evaluation space, then the acoustic mode would be dominated by the seismic mode. Intuitively, a dominated mode is redundant or "expendable" in the context of the associated design evaluation space.

Despite its status as a non-dominated mode, it can be argued that for this application, the acoustic mode incurs an excessive loss in accuracy in exchange for relatively small improvements in run-time and energy consumption compared to the seismic mode. However, the remaining three modes — the seismic, DST fusion, and ALFFS modes — represent diverse operational alternatives that offer significantly different trade-offs among the three design evaluation metrics E, R, P. For example, during critical states of operation, such as when a potential threat is detected, ALFFS may be used, while the seismic mode may be used for standby operation, and the DST fusion mode may be used during times of anticipated transition between standby and critical states.

V. CONCLUSION

In this paper, we have introduced an optimized, multi-mode embedded target detection system that employs acoustic and seismic sensors for detection of people and vehicles. The system provides two complementary modes of operation that include single-modality processing using seismic and acoustic sensors, respectively. The system also provides two dual-modality modes that incorporate sensor fusion, using methods based on Dempster Shafer Theory (DST) and a recently-introduced algorithm called Accumulation of Local Feature-level Fusion Scores (ALFFS). Experimental results demonstrate that the seismic, DST Fusion, and ALFFS modes provide flexibility in dynamically reconfiguring system execution across a range of useful operational trade-offs. Additionally, the acoustic mode is included within the system to enhance adaptability to other target detection applications and sensing devices that are more amenable to acoustic signal processing. Useful directions for future work include investigating trade-offs involving alternative windowing configurations in ALFFS, and developing low power hardware accelerators to further improve system trade-offs.

REFERENCES

[1] T. Damarla, A. Mehmood, and J. Sabatier, "Detection of people and animals using non-imaging sensors," in *Proceedings of the International Conference on Information Fusion*, 2011, pp. 1–8.

[2] T. Damarla and L. M. Kaplan, "A fusion architecture for tracking a group of people using a distributed sensor network," in *Proceedings of the International Conference on Information Fusion*, 2013, pp. 1776–1783.

[3] A. A. Dibazar, H. O. Park, and T. W. Berger, "The application of dynamic synapse neural networks on footstep and vehicle recognition," in *Proceedings of the International Joint Conference on Neural Networks*, 2007, pp. 1842–1846.

[4] S. S. Bhattacharyya, E. Deprettere, R. Leupers, and J. Takala, Eds., *Handbook of Signal Processing Systems*, 2nd ed. Springer, 2013.

[5] H. Ben Salem, T. Damarla, K. Sudusinghe, W. Stechele, and S. S. Bhattacharyya, "Adaptive tracking of people and vehicles using mobile platforms," *EURASIP Journal on Advances in Signal Processing*, vol. 2016, no. 65, pp. 1–12, 2016.

[6] K. Lee, B. S. Riggan, and S. S. Bhattacharyya, "An accumulative fusion architecture for discriminating people and vehicles using acoustic and seismic signals," in *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, New Orleans, Louisiana, March 2017.

[7] A. P. Dempster, "Upper and lower probabilities induced by a multivalued mapping," *The Annals of Mathematical Statistics*, vol. 38, no. 2, pp. 325–339, 1967.

[8] H. Wu, M. Siegel, R. Stiefelhagen, and J. Yang, "Sensor fusion using Dempster-Shafer theory [for context-aware HCI]," in *Proceedings of the IEEE Instrumentation and Measurement Technology Conference*, 2002, pp. 7–12.

[9] M. A. Hearst, S. T. Dumais, E. Osman, J. Platt, and B. Scholkopf, "Support vector machines," *IEEE Intelligent Systems and their Applications*, vol. 13, no. 4, pp. 18–28, 1998.

[10] C. Shen, W. Plishker, H. Wu, and S. S. Bhattacharyya, "A lightweight dataflow approach for design and implementation of SDR systems," in *Proceedings of the Wireless Innovation Conference and Product Exposition*, Washington DC, USA, November 2010, pp. 640–645.

[11] "Measurement computing (MCC) Linux drivers," https://github.com/wjasper/Linux_Drivers/, visited on March 3, 2017.

[12] M. Frigo and S. G. Johnson, "The design and implementation of FFTW3," *Proceedings of the IEEE*, vol. 93, no. 2, pp. 216–231, 2005.

[13] R. M. Karp and R. E. Miller, "Properties of a model for parallel computations: Determinacy, termination, queuing," *SIAM Journal of Applied Math*, vol. 14, no. 6, November 1966.

[14] S. M. Nabritt, T. Damarla, and G. Chatters, "Personnel and vehicle data collection at aberdeen proving ground (APG) and its distribution for research," US Army Research Laboratory, Tech. Rep. ARL-MR-0909, October 2015.