Available online at www.sciencedirect.com

SCIENCE DIRECT®

J. Parallel Distrib. Comput. 64 (2004) 826–838

Journal of Parallel and Distributed Computing

http://www.elsevier.com/locate/jpdc

Vehicle classification in distributed sensor networks☆

Marco F. Duarte and Yu Hen Hu*

Department of Electrical and Computer Engineering, University of Wisconsin-Madison, 1415 Engineering Dr., Madison, WI 53706, USA

Received 2 July 2003; revised 8 December 2003

Abstract

The task of classifying the types of moving vehicles in a distributed, wireless sensor network is investigated. Specifically, based on an extensive real world experiment, we have compiled a data set that consists of 820 MByte raw time series data, 70 MByte of pre-processed, extracted spectral feature vectors, and baseline classification results using the maximum likelihood classifier. The purpose of this paper is to detail the data collection procedure, the feature extraction and pre-processing steps, and baseline classifier development. The database is available for download at http://www.ece.wisc.edu/˜sensit starting on July 2003.

1. Introduction

The emergence of small, low-power devices that integrate micro-sensing and actuation with on-board processing and wireless communication capabilities stimulates great interests in wireless distributed sensor networks (WDSN) [5,6,10], A WDSN is often deployed to perform tasks such as detection, classification, localization and tracking of one or more targets within the sensor field. The sensors are typically battery-powered and have limited wireless communication bandwidth. Therefore, efficient collaborative signal processing algorithms that consume less energy for computation and communication are needed for these applications [7].

Vehicle type classification is an important signal processing task that has found widespread military and civilian applications such as intelligent transportation systems. Typically, acoustic [13,12,2,9,4,14] or seismic [11] sensors are used for such a purpose. However, previous results have focused on the classification based on signals obtained at a single or few sensors and processed in a centralized manner. Hence these existing results is only partially useful for a WDSN application.

In this paper, we consider the implementation of such a task in a WDSN environment. Each sensor in the WDSN will be equipped with a microphone or a geophone. Upon detection of the presence of a vehicle in the vincinity of the sensor, the on-board processor will extract feature vectors based on the acoustic or seismic signal sensed by the sensors. In a wireless sensor network, the communication bandwidth is very limited. Hence, instead of sending the feature vector, a local pattern classifier at each sensor node will first make a local decision on what type of the vehicle is based on its own feature vector. Statistically, this is a multiple-hypotheses testing problem. The probability of correct classification can also be estimated. The local decision, together with the estimated probability of being a correct decision then can be encoded and transmitted efficiently via the wireless channel to a local fusion center ready for decision fusion. Hence, from a signal processing point of view, the WDSN vehicle classification problem comprises of two parts: local classification and global decision fusion.

The purpose of this paper is to describe the development of a WDSN vehicle classification data set, and the baseline performance when a set of existing pattern classification methods are applied. This data set is extracted based on the sensor data collected during a real world WDSN experiment carried out at Twenty-nine Palms, CA in November 2001. This data set includes: (a) the raw time series data observed at each individual sensors, (b) a set of acoustic feature vectors

☆This project is supported by DARPA under grant no. F 30602-00-2-0555.

*Corresponding author. Fax: +608-262-1267.

E-mail address: hu@engr.wisc.edu (Yu Hen Hu).

extracted from each sensor's microphone and (c) class label manually assigned to each feature vector by a human operator to ensure high accuracy of the class labels. Accompanying this data set is a suite of pattern classifier programs written in Matlab m-file format to perform local classification of the feature vectors provided in the data set. Also included are training and testing results of local classification and global decision fusion.

This data set and its accompanying programs are available for download at the web address: http://www.ece.wisc.edu/~sensit

By providing these to the research community, the results presented in this paper serve as a state-of-the-art baseline performance benchmark to be compared to future vehicle classification results obtained using this data set.

The rest of this paper is organized as follows: The characteristic of a WDSN will be discussed in Section 2. The twenty-nine Palms WDSN experiment will then be reviewed, and the raw acoustic data collection method will be summarized in Section 3. In Sections 4 and 5, we survey existing acoustic features used for the vehicle classification purpose. We then describe the spectrum based feature extraction procedure and feature selection procedure that yield a set of judiciously selected feature vectors. In Section 6, we briefly review several existing pattern classifiers, including the nearest neighbor classifier, maximum likelihood classifier with uni-variate Gaussian probability density function model, and support vector machine. Then the local classification results using these classifers will be reported. In Section 7, we report the decision fusion results based on majority voting as well as a weighted voting method.

2. Characteristics of a wireless distributed sensor network

In a wireless distributed sensor network, individual sensor nodes are deployed randomly over a given sensor field. Each sensor node will be equipped with an on-board processor, a wireless communication transceiver, various types of sensors, digital sampling devices, and battery. Often, sensor nodes within a geographical region will be grouped to form a local cluster so that a certain hierarchy of command and control over the entire sensor field can be established. Each local cluster will elect one or more sensor nodes as the cluster head where spatial decision fusion of sensors within a cluster will be performed.

Before vehicle type classification can be embarked, individual sensors will need to be activated, and then periodically perform target detection algorithm to detect the presence of a moving vehicle in the neighborhood of the sensor. Once a positive detection is made, the pattern classification algorithm will start running to classify the incoming acoustic signature into one of the pre-defined classes.

Up to now, all these tasks will be performed in the on-board microprocessor of each individual sensor node. Hence, the key issue here is to reduce complexity of computation and on-board storage requirement and therefore conserve on-board energy reserve. This energy constraint implies that not all classification algorithms will be suitable for the implementation on a WDSN sensor node. As such, performance and energy consumption trade-offs must be sought.

The local decisions can be encoded efficiently and transmitted from individual sensor node to the local cluster head for decision fusion. Since not all sensor nodes within a WDSN will detect the presence of a moving vehicle within the sensor field, not every sensor node will produce a local classification result. Furthermore, due to wireless communication error and possible network congestion, not all local decisions can be reported back to the cluster head in time for decision fusion. As such, the decision fusion must be performed with imperfect knowledge of local decisions.

The nodes distributed in a geographic region are usually partitioned according to space time cells as illustrated in Fig. 1. Each cell has a manager node which is responsible for coordinating the networking/routing protocols and CSP algorithms within that cell. Real-time sampled data is obtained from the sensors in different sensing modalities for different events involving moving target vehicles. Sensing modalities could be acoustic, seismic, Passive Infra-Red (PIR) to name a few.

Detection of an event involving a target at a node requires minimum a priori knowledge and can be performed using an energy-based Constant False Alarm Rate (CFAR) detection algorithm which dynamically adjusts the detection threshold. Temporal processing on the sampled data of a detected event at a node is carried on to obtain signatures or features that are used for classification. The type of features to be used (e.g. FFT-based, wavelet-based) and extracting those features relevant for classification is a highly challenging problem in itself and several methods commonly used in pattern recognition find their application here.

A wide variety of algorithms have been proposed in literature for the purpose of classification [3], each having its own advantages and disadvantages. The main objective in the distributed sensor network case is to develop low complexity algorithms that classify these extracted features so as to make efficient use of the limited power and bandwidth capabilities of the nodes. Techniques based on Maximum Likelihood (ML) estimation, Support Vector Machines (SVM), k-Nearest Neighbor (kNN) and Linear Vector Quantization have been developed. Different algorithms could be used in conjunction to provide algorithmic heterogeneity.

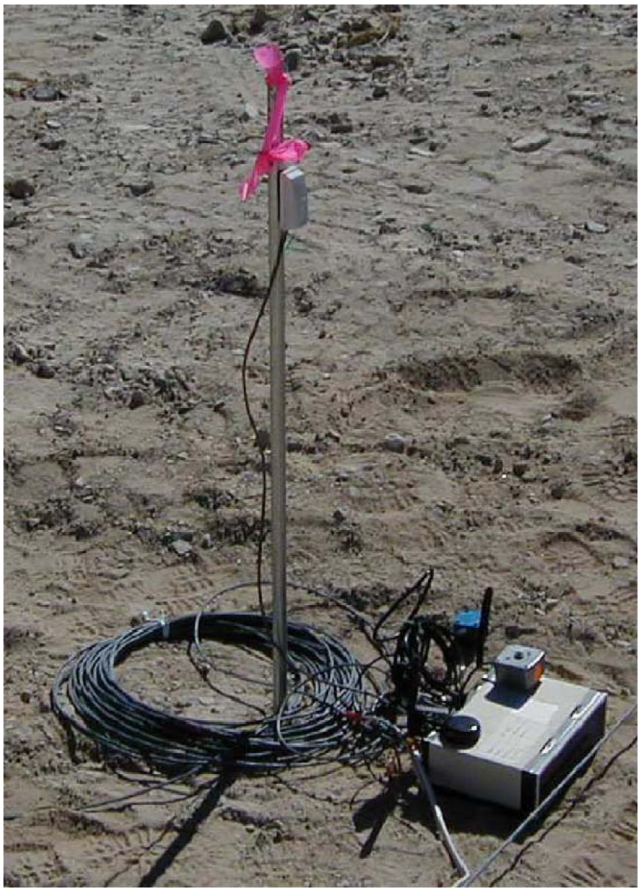

Fig. 1. A Sensoria WINS NG 2.0 node.

However, the insights offered by the ML technique and its low computational and storage requirements as compared to the other techniques makes it the favored algorithm for node-based classification.

3. Experiment description

The data set that are being discussed in this paper was collected at the third SensIT situational experiment (SITEX02), organized by DARPA/IXOs SensIT (Sensor Information Technology) program. In this experiment, seventy-five WINS NG 2.0 nodes [8] were deployed at the Marine Corps Air Ground Combat Center in Twenty-nine Palms, CA, USA. During a two-week period, various experiments have been conducted. A map of the entire field is depicted in Fig. 2 which consists of a east–west road and a south–north road and an intersection area. The data collected for this data set were recorded on a rectangular sub-region of size meters by meters during (dates). (Describe the runs, which type of vehicle moves from where to where, etc.)

Testing runs were performed by driving different kinds of vehicles across the testing field, where nodes were deployed following the arrangement shown in

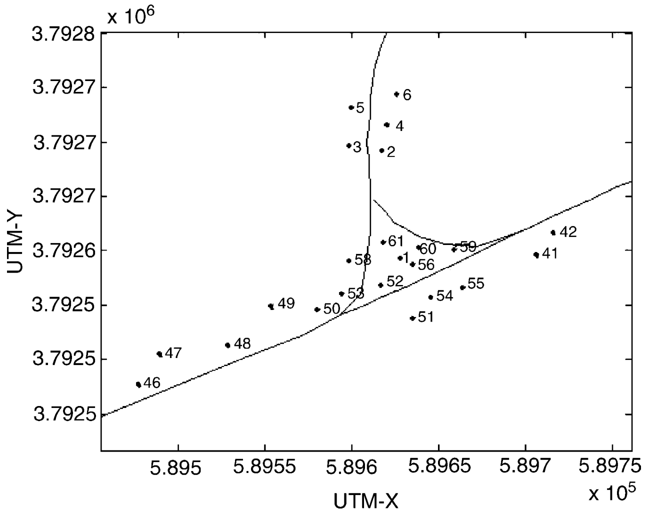

Fig. 2. Sensor field layout.

Fig. 2. The sensor field is an area of approximately $900 \times 300\ \text{m}^2$ at MCAGCC. The sensors, denoted by dots of different colors in Fig. 2 are placed along the side of the road. The separation of adjacent sensors ranges from 20–40 m.

The WINS NG 2.0 nodes, shown in Fig. 1, provide a system on which SensIT users can build and test their distributed sensor algorithms. Each sensor node is equipped with three types of sensing modalities: acoustic (microphone), seismic (geophone), and infrared (polarized IR sensor). The sampling rate for the acoustic and seismic signals are (how many) hertz, and for the PIR signal is (how many) hertz. The NG 2.0 nodes consists of a A/D converter and an on-board programmable digital signal processor that digitize the analog signal and place them into a circular buffer. For the purpose of recording raw sensor data for later analysis, a back-end ethernet network were laid that serves solely for the purpose of data collection.

Four target vehicle classes, namely Assault Amphibian Vehicle (AAV), Main Battle Tank (M1), High Mobility Multipurpose Wheeled Vehicle (HMMWV) and Dragon Wagon (DW) were used. Each node records the acoustic, seismic and infrared signal for the duration of the run. The objective is to detect the vehicles when they pass through each region. The type of the passing vehicle then will be identified, and the accurate location of that vehicle will be estimated using an *energy-based localization algorithm*.

4. Event extraction

The nodes used in the experiment record the data for different sensors, or modalities; data is recorded for acoustic, seismic and infrared modalities at a rate of 4960 Hz.

For the different vehicle types, the vehicle was driven around the three roads shown in the Fig. 2; each road received a different run number. The west to north road was numbered 1; the north to east road was numbered 2, and the east to west road was numbered 3. Subsequent runs were named incrementally. Thus, each run was named after the vehicle tested and the road covered; i.e. AAV3, AAV4, AAV5, etc.

After the series were recorded, it was needed to extract the actual event from the run series. Although the run might be several minutes in length, the event series will be much shorter, as it only spans the short period of time when the target is close to the node. During the Collaborative Signal Processing Tasks, the detection algorithm determines whether the vehicle is present or not in the region in order to perform classification on the time series. The CFAR detection algorithm outputs a decision every 0.75 s, based on the energy level of the acoustic signal, as shown in Fig. 3.

For the data set extraction, a k-Nearest Neighbor classifier was used to label each 0.75-s data segment from each separate node as a detection or non-detection. Two features are used for this classification: the distance between the vehicle and the node and the acoustic signal energy for that given time. The runs AAV3 and DW3 were used for training, and the events in these runs were identified manually, i.e. directly listening to the time series. From this classifier we obtain the event labelling for each one of the nodes for each run. We use clustering to reduce the number of events per run if possible (Fig. 4).

The result of this procedure is the extraction of time series of variable lengths that will contain the acoustic, seismic and PIR information of the time surrounding the closest point of approach of the vehicle to the node (Fig. 5).

5. Feature extraction

The event time series are used to extract multi-dimensional features for classification purposes. The infrared modality is not used at this stage, as the observation signal length is very short and not uniform across events. For this data set, the extracted features are based on the frequency spectrum of the acoustic and seismic signals of the event. The Fast Fourier Transform (FFT) of these signals is calculated for every 512 point sample (every 10.32 ms for the current sample rate), which yields 512 FFT points with resolution of 9.6875 Hz.

For the acoustic modality, we chose the first 100 points, containing frequency information of up to 968.75 Hz. The points are averaged by pairs, resulting in a 50-dimensional FFT-based feature with resolution of 19.375 Hz with information for frequencies up to 968.75 Hz. For the seismic modality, we chose the first

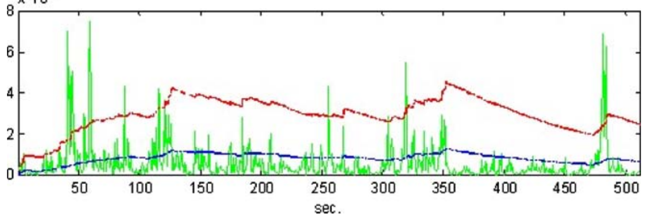

Fig. 3. Constant False Alarm Rate (CFAR) algorithm: times with high energy values are marked as detections.

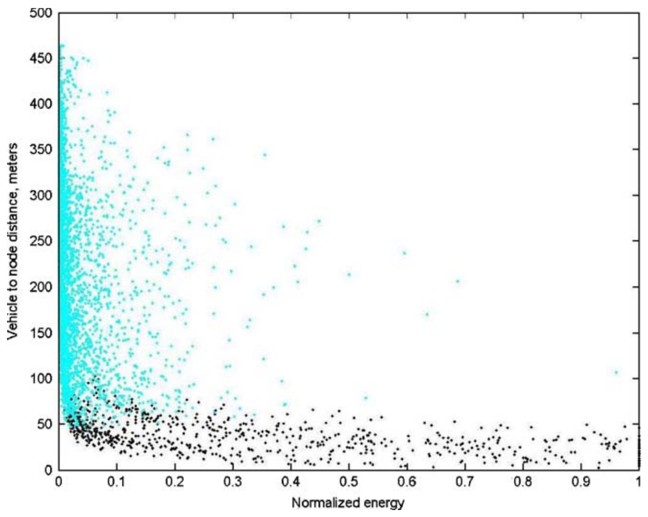

Fig. 4. Detection labelling for training runs. The two axes represent the two dimensions of the feature; dark marks represent detections and light marks represent non-detections.

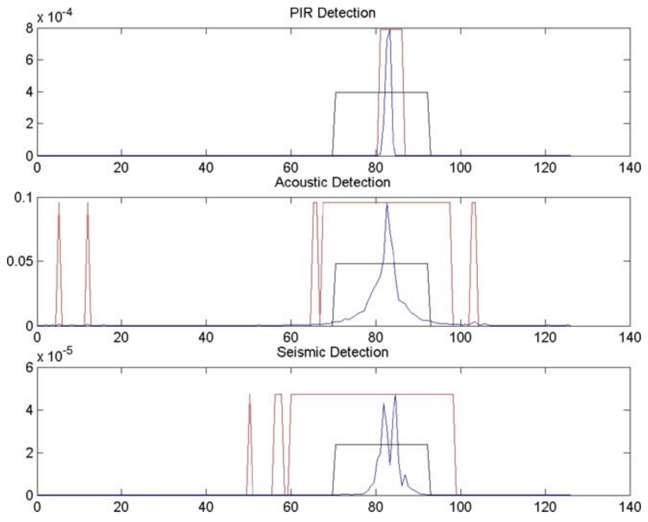

Fig. 5. Sample detection label and CFAR detection result; blue line represents energy, black line represents detection label derived from the kNN classifier, and the red line represents the CFAR detection result.

50 points, containing frequency information of up to 484.375 Hz. This results in a 50-dimensional FFT-based feature with resolution of 9.6875 Hz with information for frequencies up to 484.375 Hz. All features are normalized and means are removed.

6. Local classification

In this section, we will provide some baseline evaluation of the data set using three common classification algorithms. It is worth noting that in real-life situations, the largest error-inducing factor for the vehicle surveillance detection case is the presence of high-energy noise factors, such as wind and radio chatter. In order to avoid false classification of these false detections into a valid vehicle class, we have implemented a noise class with features extracted from the timeseries that show the occurrence of one of these high-energy noise events. Thus, for the experiments, we have created a three-class classification scenario; we test it using the k-Nearest Neighbor, Maximum Likelihood, and Support Vector Machine algorithms.

6.1. k-nearest neighbor classifier

k-NN, is one of the simplest, yet very accurate, classification methods. It is based on the assumption that examples that are close in the instance space belong to the same class. Therefore, an unseen instance should be classified as the majority class of its k $(1 \leqslant k)$ nearest neighbors in the training data set. Although the k-NN algorithm is quite accurate, the time required to classify an instance is high, since the distance (or similarity) of that instance to all the instances in the training set have to be computed. Therefore, the classification time in k-NN algorithm is proportional to the number of features and the number of training instances.

6.2. ML classifier

The samples (features) in each of the C classes are assumed to have been drawn independently according the probability law $p(\mathbf{x} \mid \omega_i)$, $i = 1, 2, \dots C$. We further assume that $p(\mathbf{x} \mid \omega_i)$ has a know parametric form, i.e. it is multivariate normal with the density

$$p(\mathbf{x} \mid \omega_i) = \frac{1}{(2\pi)^{d/2}|\boldsymbol{\Sigma}|^{1/2}} \exp\left[-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^H \boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right] \quad (1)$$

and is therefore determined uniquely by the value of a parameter vector $\boldsymbol{\theta}_i$ which consists of the components $\boldsymbol{\mu}_i$ and $\boldsymbol{\Sigma}_i$, the mean and covariance matrices, respectively.

$$p(\mathbf{x} \mid \omega_i) \approx N(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i) \quad (2)$$

Our problem of classification then reduces to using the information provided by the training samples to obtain good estimates for the unknown parameter vectors $\boldsymbol{\theta}_i$, $i = 1, 2, \dots, C$. For this we use a set of samples for a particular class i drawn independently from the probability density $p(\mathbf{x}|\boldsymbol{\theta}_i)$ to estimate the unknown parameter vector. Suppose this set contains n samples, $\mathbf{x}_1, \mathbf{x}_2, \dots \mathbf{x}_n$. Then the log-likelihood function can be represented as:

$$l(\theta) = \sum_{k=1}^{n} \ln p(\mathbf{x}_k \mid \theta). \quad (3)$$

The maximum likelihood estimate of $\boldsymbol{\theta}$ is, by definition, the value $\hat{\boldsymbol{\theta}}$ that maximizes $l(\boldsymbol{\theta})$. This maximum likelihood estimate $\hat{\boldsymbol{\theta}}$ can be obtained from the set of equations

$$\nabla_{\boldsymbol{\theta}} l = 0. \quad (4)$$

The ML estimates for $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ are thus given by:

$$\hat{\boldsymbol{\mu}} = \frac{1}{n}\sum_{k=1}^{n} \mathbf{x}_k, \quad (5)$$

$$\hat{\boldsymbol{\Sigma}} = \frac{1}{n}\sum_{k=1}^{n} (\mathbf{x}_k - \hat{\boldsymbol{\mu}})(\mathbf{x}_k - \hat{\boldsymbol{\mu}})^H. \quad (6)$$

Using a set of discriminant functions $g_i(\mathbf{x})$, $i = 1, 2, \dots, C$, the classifier is said to assign a feature vector $\mathbf{x}$ to class ω_i if $g_i(\mathbf{x}) > g_j(\mathbf{x})$ for all $j \neq i$.

For minimum error rate classification we take the maximum discriminant function to correspond to the maximum posterior probability

$$g_i(\mathbf{x}) = p(\omega_i \mid \mathbf{x}) = \frac{p(\mathbf{x} \mid \omega_i)p(\omega_i)}{\sum_{j=1}^{C} p(\mathbf{x} \mid \omega_j)p(\omega_j)} \quad (7)$$

which can be simplified to

$$g_i(\mathbf{x}) = \ln p(\mathbf{x} \mid \omega_i) + \ln p(\omega_i). \quad (8)$$

This expression can be readily evaluated since we have assumed the densities $p(\mathbf{x} \mid \omega_i)$ are multivariate normal:

$$g_i(\mathbf{x}) = -\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_i)^H \boldsymbol{\Sigma}_i^{-1}(\mathbf{x}-\boldsymbol{\mu}_i) - \frac{d}{2}\ln 2\pi - \frac{1}{2}\ln|\boldsymbol{\Sigma}_i| + \ln P(\omega_i) \quad (9)$$

6.3. Support vector machine classifier

The SVM classifier used here is a C support vector classification (C-SVC), implemented in LIBSVM. In short, C-SVC solves the following primal problem:

$$\min_{w,b,\varepsilon}\left(\frac{1}{2}w^T w + C\sum_{i=1}^{l}\varepsilon_i\right) \quad (10)$$

under constraints

$$y_i(w^T\phi(x_i) + b) \geqslant 1 - \varepsilon_i \quad (11)$$

and $\varepsilon \geqslant 0$ for $i = 1, 2, \dots, l$, x_i and y_i are the training data (feature vector) and the associated class label, respectively. The dual problem is:

$$\min_{w,b,\varepsilon}\left(\frac{1}{2}\alpha^T Q\alpha - e^T\alpha\right) \tag{12}$$

under constraints $0 \leqslant \alpha_i \leqslant C$ and $y^T\alpha = 0$, where e is a all-one vector, $C > 0$ is the upper bound, Q is an l by l positive semi-definite matrix, $Q_{ij} := y_i y_j K(x_i, x_j)$, and $K(x_i, x_j) := \phi(x_i)^T\phi(x_j)$ is the kernel. The function ϕ maps the training data x_i into a higher-dimensional space. And the decision rule for categorizing a test feature x is (assume two classes with labels 1 and −1):

$$sign\left(\sum_{i=1}^{l} y_i\alpha_i K(x_i, x) + b\right). \tag{13}$$

The C-SVC we used for classifying Sitex02 data has the C value equal to 1 ($C = 1$). The kernel used is a polynomial kernel with the following format:

$$K(x_i, x_j) = (1 + x_i^T x_j)^2. \tag{14}$$

Rough size estimates for the training sets are as shown in Table 1. SV stands for number of support vectors used.

6.4. Results metrics

The results are given in the form of a *confusion matrix*, which classifies the vectors/events by their actual classification (rows, x_i), and the experimental classification result (columns, y_i). The results from the partition tests are added up to get the result for each feature and each classifier.

The *detection probability* for each class is the ratio from the number of samples/events correctly classified for that class to the total number of samples/events in that class: $P(y_i \mid x_i)$.

The *false alarm probability* for each class is the ratio from the number of samples/events of all other classes classified as that class to the total number of samples/events of other classes: $P(y_i \mid x_i)$.

The *classification rate* is the ratio from the number of samples/events correctly classified for all classes to the total number of samples/events: $P(y_i \wedge x_i)$.

Example:

$$\text{Confusion matrix} \quad \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix}.$$

$$\text{Detection probability for class 1} \quad \frac{a}{a+b+c}.$$

False alarm probability for class 2

$$\frac{b+h}{a+b+c+g+h+i}.$$

Classification rate

$$\frac{a+e+i}{a+b+c+d+e+f+g+h+i}. \tag{15}$$

To validate the results of a classifier given a certain data set, the set is randomly split into two parts: one is used as the training set and the other is used as a validation set, in order to estimate the generalization error. A simple generalization of this method is the *m-way or m-fold cross-validation*. In this case, the data set is randomly divided into m disjoint sets of roughly equal size n/m, where n is the total number of feature vectors available in the data set. The classifier is trained m times, each time with a different set held out as a validation set. The estimated performance is the mean of the m errors. In this case, we use $m = 3$ and name the three different validation cases Q1, Q2 and Q3.

6.5. Results

Tables 2 and 3 show the confusion matrices for the different classification algorithms tested with the current data set for acoustic and seismic modality, respectively (Fig. 6). The table offers a wealth of information regarding the feasibility of differentiation among the proposed classes, as well as the effect of unwanted noise in the classification process. Tables 4 and 5 present the detection, false alarm and classification rates for the same cases.

7. Region fusion

Apart from the localization and tracking of the target, it is also necessary to classify the type of vehicle within

Table 1
Training set sizes for SVM classifier

Partition	Q1		Q2		Q3		Average	
Classification Modality	File Size (kB)	Set Size (SVs)	File Size (kB)	Set Size (SVs)	File Size (kB)	Set Size (SVs)	File Size (kB)	Set Size (SVs)
Acoustic	21952.8877	38518	21796.37305	38246	21781.75293	38218	21843.67122	38327.33333
Seismic	25424.08789	44531	25544.11621	44742	25443.52	44565	25470.58	44612.67

Table 2
Confusion matrices for different classifiers using 3-way cross-validation on acoustic modality

Testing Partition	Q1			Q2			Q3			Total		
k-nearest neighbor	5165	657	1791	5176	627	1811	5073	614	1927	15414	1898	5529
	531	5344	2932	542	5410	2856	502	5412	2894	1575	16166	8682
	1569	2029	11037	1599	2018	11019	1672	1975	10989	4840	6022	33045
Maximum likelihood	5667	829	1117	5597	805	1212	5554	875	1185	16818	2509	3514
	1282	5730	1795	1253	5804	1751	1263	5743	1802	3798	17277	5348
	1660	2859	10116	1667	2991	9978	1732	2853	10051	5059	8703	30145
Support vector machine	5134	829	1650	5032	824	1758	5032	836	1746	15198	2489	5154
	574	5296	2937	589	5290	2929	552	5291	2965	1715	15877	8831
	621	2729	11285	645	2833	11158	719	2703	11214	1985	8265	33657

Table 3
Confusion matrices for different classifiers using 3-way cross-validation on seismic modality

Testing Partition	Q1			Q2			Q3			Total		
k-nearest neighbor	4195	2020	1398	4235	2027	1352	4206	2085	1323	12636	6132	4073
	3033	4409	1365	2957	4401	1450	3005	4388	1415	8995	13198	4230
	2509	3294	8832	2456	3254	8926	2500	3326	8810	7465	9874	26568
Maximum likelihood	5090	1769	754	5050	1819	745	5173	1679	762	15313	5267	2261
	2983	3541	2283	2918	3535	2355	2955	3506	2347	8856	10582	6985
	2693	1057	10885	2622	1100	10914	2638	1167	10831	7953	3324	32630
Support vector machine	4388	2614	611	4372	2635	607	4489	2490	635	13249	7739	1853
	1949	4989	1869	1843	5100	1865	1901	4908	1999	5693	14997	5733
	2318	1964	10353	2237	1956	10443	2267	1976	10393	6822	5896	31189

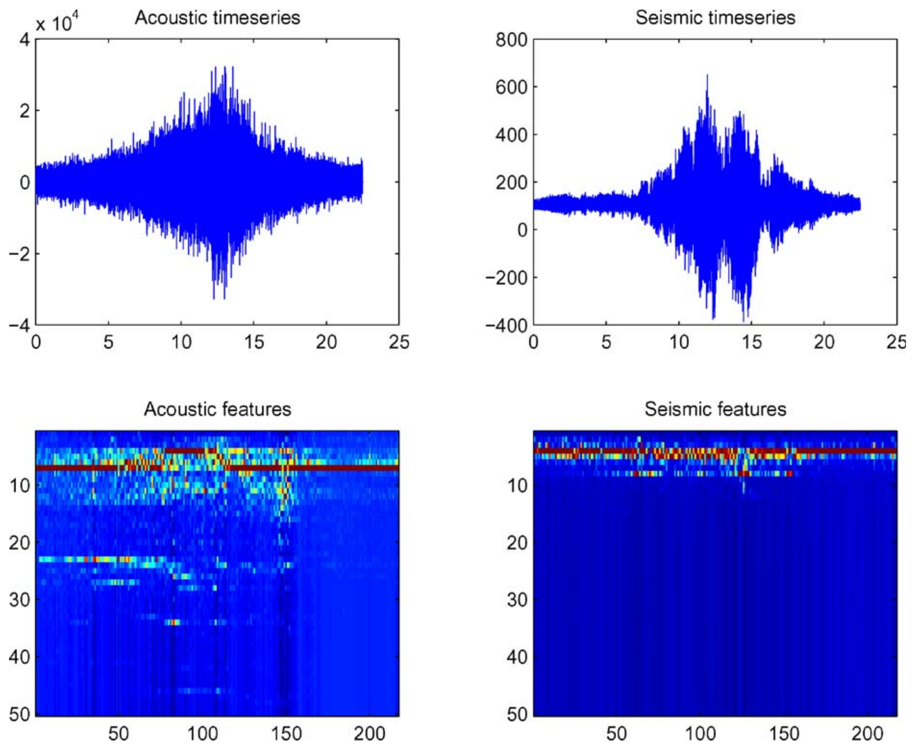

Fig. 6. Sample time series and classification features for acoustic and seismic modalities.

the region based on target classification results reported from member sensor nodes. Note that in our current system architecture, the target localization may be performed prior to region-wide target classification. Hence, if the target position is relatively accurate, it is possible to use the estimated target location and known

Table 4
Classification, detection and false alarm rates for different classifiers using 3-way cross-validation on acoustic modality

Measurement class	Detection rate			False alarm rate			Classification
	AAV(%)	DW(%)	Noise(%)	AAV(%)	DW(%)	Noise(%)	rate
k-nearest neighbor	67.48	61.18	75.26	29.39	32.88	30.07	69.36
Maximum likelihood	73.63	65.39	68.66	34.50	39.36	22.72	68.95
Support vector machine	66.54	60.09	76.66	19.58	40.38	29.35	69.48

Table 5
Classification, detection and false alarm rates for different classifiers using 3-way cross-validation on seismic modality

Measurement class	Detection rate			False alarm rate			Classification
	AAV(%)	DW(%)	Noise(%)	AAV(%)	DW(%)	Noise(%)	rate
k-nearest neighbor	55.32	49.95	60.51	56.57	54.81	23.81	56.24
Maximum likelihood	67.04	40.05	74.32	52.33	44.81	22.08	62.81
Support vector machine	58.01	56.76	71.03	48.58	47.62	19.56	63.79

sensor coordinates to calculate the target–sensor distance. Then, one may estimate the empirically derived probability of correct classification at a particular sensor node based on the distance information as described in Section 3.2.

7.1. Data fusion

Statistically speaking, data fusion [1] is the process of estimating the joint posterior probability (likelihood function in the uninformed prior case) based on estimates of the marginal posterior probability. Let $x(i)$ denote the feature vector observed at the ith sensor node within the region, C_k denotes the kth type of vehicle, the goal is to identify a function $f(\cdot)$ such that

$$\begin{aligned} &P(x \in C_k \mid x(1), \ldots, x(N)) \\ &\quad \triangleq P(x \in C_k \mid \underset{\sim}{x}). \\ &\quad \approx f(g(P(x \in C_k \mid x(i))), 1 \leqslant i \leqslant N). \end{aligned} \tag{16}$$

In our current work, we let the maximum function $g(z_k) = 1$ if $z_k > z_j$, $k \neq j$, and $g(z_k) = 0$ otherwise. Hence, our approach is known as *decision fusion*. Conventionally, there are two basic forms of the fusion function f.

(1) *Multiplicative form*: If we assume that $x(i)$ and $x(j)$ are statistically independent feature vectors, then

$$P(x \in C_k \mid \underset{\sim}{x}) = \prod_{i=1}^{N} P(x \in C_k \mid x(i)). \tag{17}$$

This approach is not realistic in the sensor network application and cannot be easily adapted to a decision fusion framework.

(2) *Additive form*: The fusion function is represented as a weighted sum of the marginal posterior probability or local decisions:

$$\hat{P}(x \in C_k) = \sum_{i=1}^{N} w_i g_i(P(x \in C_k \mid x(i))). \tag{18}$$

A baseline approach of region-based decision fusion would be simply choose $w_i = 1$ for $1 \leqslant i \leqslant N$. This would be called the *simple voting* fusion method.

7.2. Maximum a posterior decision fusion

With distance-based decision fusion, we make each of the weighting factors w_i in Eq. (18) a function of distance and signal to noise ratio, that is $w_i = h(d_i, s_i)$ where d_i is the distance between the ith sensor and the target and s_i is the signal-to-noise ratio defined as

$$\mathrm{SNR}_{\mathrm{dB}} = 10 \times \log_{10}\left(\frac{E_\mathrm{s} - E_\mathrm{n}}{E_\mathrm{n}}\right). \tag{19}$$

where E_s is the signal energy and E_n is the noise mean energy, both determined by the CFAR detection algorithm. We can use then the characterization gathered from the experiment referred in Section 2 to formulate a Maximum A Posterior (MAP) Probability Gating network, using the Bayesian estimation

$$\hat{P}(x \in C_k \mid \underset{\sim}{x}) = P(x \in C_k \mid \underset{\sim}{x}, d_i, s_i) \cdot P(d_i, s_i). \tag{20}$$

The prior probability $P(d_i, s_i)$ is the probability that the target is at the distance range d_i, and the acoustic signal SNR_{dB} is at the s_i range, and can be estimated empirically from the experiments. The conditional probability $P(\underset{\sim}{x} \mid d_i, s_i)$ is also available from the empirically gathered data. With these, we may simply assign the following weights in Eq. (18):

$$w_i = P(\underset{\sim}{x} \mid d_i, s_i) \cdot P(d_i, s_i). \tag{21}$$

In other words, if a particular sensor's classification result is deemed as less likely to be correct, it will be excluded from the classification fusion.

We now have another possible choice of w_i. That is,

$$w_i = \begin{cases} 1 & d_i < d_j,\ j \neq i, \\ 0 & \text{otherwise.} \end{cases} \tag{22}$$

This choice of weights represents a nearest neighbor approach, where the result of the closest node to the target is assumed to be the region result.

We can use other choices that are functions only of distance. In this work, we use a simple threshold function:

$$w_i = \begin{cases} 1 & d_i \leqslant d_{\max}, \\ 0 & \text{otherwise.} \end{cases} \tag{23}$$

We compare these three different methods of choosing w_i to the baseline method of setting $w_i = 1$ for all i, and test them using seven different experiments in the Sitex02 data set, using one out of n training and testing. Our metrics are the classification rate and the rejection rate.

The classification rate is the ratio between the number of correctly classified samples and the total numbered of samples classified as vehicles. The rejection rate is the rate between the number of samples rejected by the classifier and the total number of samples ran through the classification algorithm. Consequentially, the acceptance rate is the complement of the rejection rate.

There are two rejection scenarios with our current classifier scheme; one is at the node level, where one of the classes characterized during training collects typical samples of events with high energy that do not correspond to vehicles. These events are incorrectly detected and include such noises as wind, radio chatter and speech. The other is at the region level, where the region fusion algorithm does not specify satisfactorily a region classification result, i.e. no nodes were closer than $d_{\max}$ to the vehicle for the distance-based region fusion algorithm.

It is desired to obtain high classification rates while preserving low rejection rates. The results are listed in Tables 6 and 7. To analyze the impact of localization errors in the different methods, errors were injected to the ground truth coordinates following a zero-mean Gaussian distribution with several standard deviations. The results are shown in Tables 8–13.

7.3. Results

For Tables 6–13, the cells that give the highest classification rate are highlighted, including tied cases. It is seen that nearest neighbor method yields out the best results consistently when the error is low or nonexistent—in 9 out of 14 cases. The distance-based and MAP-based methods give comparable results in

Table 6
Classification rate fusion results using 4 methods

Fusion method	MAP Bayesian 77.19%	$d_{\max}$ = 50 m80.82%	Nearest neighbor **83.55%**	Majority voting 75.58%
AAV3(%)	33.87	50.79	**73.33**	27.12
AAV6(%)	**100.00**	**100.00**	**100.00**	**100.00**
AAV9(%)	89.80	90.63	84.31	**91.84**
DW3(%)	80.00	83.78	**85.71**	82.50
DW6(%)	**100.00**	**100.00**	**100.00**	**100.00**
DW9(%)	66.67	75.00	**75.86**	63.33
DW12(%)	**70.00**	65.52	65.63	64.29

Table 7
Rejection rate fusion results using 4 methods

Fusion method	MAP Bayesian 9.53%	$d_{\max}$ = 50 m21.56%	Nearest neighbor **7.40%**	Majority voting 10.40%
AAV3(%)	3.13	**1.56**	6.25	7.81
AAV6(%)	4.29	27.14	**2.86**	7.14
AAV9(%)	3.92	37.25	**0.00**	3.92
DW3(%)	4.76	11.90	**0.00**	4.76
DW6(%)	6.06	9.09	**0.00**	**0.00**
DW9(%)	**14.29**	31.43	17.14	**14.29**
DW12(%)	30.23	32.56	**25.58**	34.86

Table 8
Classification rate fusion results using 4 methods, and error injection with $\sigma = 12.5$ m

Fusion method	MAP Bayesian 77.14%	$d_{\max}$ = 50 m80.51%	Nearest neighbor **81.89%**	Majority voting 75.58%
AAV3(%)	32.79	56.45	**67.21**	27.12
AAV6(%)	**100.00**	**100.00**	**100.00**	**100.00**
AAV9(%)	**93.88**	90.63	84.31	91.84
DW3(%)	80.00	81.08	**83.33**	82.50
DW6(%)	**100.00**	**100.00**	**100.00**	**100.00**
DW9(%)	66.67	**78.26**	75.86	63.33
DW12(%)	**66.67**	57.14	62.50	64.29

Table 9
Rejection rate fusion results using 4 methods, and error injection with $\sigma = 12.5$ m

Fusion method	MAP Bayesian 9.75%	$d_{\max} = 50$ m 22.32%	Nearest neighbor **7.40%**	Majority voting 10.40%
AAV3(%)	4.69	**3.13**	6.25	7.81
AAV6(%)	4.29	25.71	**2.86**	7.14
AAV9(%)	3.92	37.25	**0.00**	3.92
DW3(%)	4.76	11.90	**0.00**	4.76
DW6(%)	6.06	9.09	**0.00**	**0.00**
DW9(%)	**14.29**	34.29	17.14	**14.29**
DW12(%)	30.23	34.88	**25.58**	34.86

Table 10
Classification rate fusion results using 4 methods, and error injection with $\sigma = 25$ m

Fusion method	MAP Bayesian 77.74%	$d_{\max} = 50$ m 79.42%	Nearest neighbor 79.29%	Majority voting **75.56%**
AAV3(%)	37.70	54.39	**55.36**	27.12
AAV6(%)	**100.00**	**100.00**	**100.00**	**100.00**
AAV9(%)	89.80	**100.00**	88.24	91.84
DW3(%)	80.00	**82.86**	80.95	82.50
DW6(%)	**100.00**	**100.00**	**100.00**	**100.00**
DW9(%)	66.67	72.00	**72.41**	63.33
DW12(%)	**70.00**	46.67	58.06	64.29

Table 11
Rejection rate fusion results using 4 methods, and error injection with $\sigma = 25$ m

Fusion method	MAP Bayesian 9.75%	$d_{\max} = 50$ m 24.78%	Nearest neighbor **8.63%**	Majority voting 10.40%
AAV3(%)	**4.69**	10.94	12.50	7.81
AAV6(%)	4.29	30.00	**2.86**	7.14
AAV9(%)	3.92	50.98	**0.00**	3.92
DW3(%)	4.76	16.67	**0.00**	4.76
DW6(%)	6.06	6.06	**0.00**	**0.00**
DW9(%)	**14.29**	28.57	17.14	**14.29**
DW12(%)	30.23	30.23	**27.91**	34.88

Table 12
Classification rate fusion results using 4 methods, and error injection with $\sigma = 50$ m

Fusion method	MAP Bayesian 77.74%	$d_{\max} = 50$ m **80.48%**	Nearest neighbor 76.72%	Majority voting 75.58%
AAV3(%)	37.70	**51.28**	39.29	27.12
AAV6(%)	**100.00**	**100.00**	**100.00**	**100.00**
AAV9(%)	89.80	**95.00**	86.27	91.84
DW3(%)	80.00	**84.62**	78.57	82.50
DW6(%)	**100.00**	95.24	96.97	**100.00**
DW9(%)	66.67	**72.22**	71.43	63.33
DW12(%)	**70.00**	65.00	64.52	64.29

Table 13
Rejection rate fusion results using 4 methods, and error injection with $\sigma = 50$ m

Fusion method	MAP Bayesian 9.95%	$d_{\max} = 50$ m 46.01%	Nearest neighbor **9.24%**	Majority voting 10.40%
AAV3(%)	**4.69**	39.06	12.50	7.81
AAV6(%)	5.71	45.71	**4.29**	7.14
AAV9(%)	3.92	60.78	**0.00**	3.92
DW3(%)	4.76	38.10	**0.00**	4.76
DW6(%)	6.06	36.36	**0.00**	**0.00**
DW9(%)	**14.29**	48.57	20.00	**14.29**
DW12(%)	30.23	53.49	**27.91**	34.88

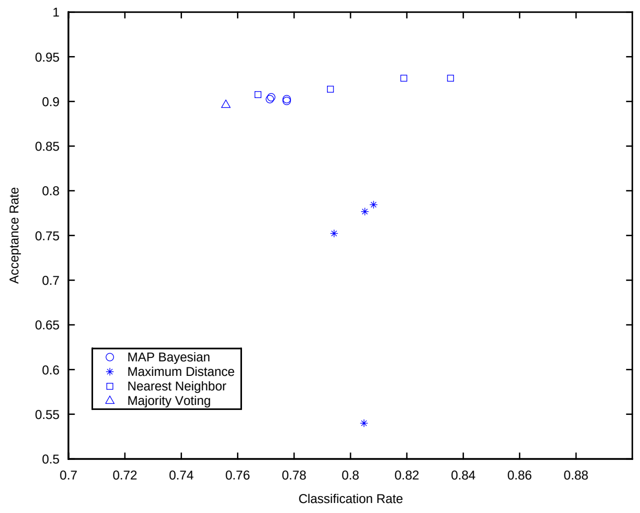

Fig. 7. Average classification and acceptance rate results for different classification region fusion methods.

cases where the error is larger (each method has the highest rate in 4–6 cases out of 14). However, the rejection rates are unacceptable for the distance-based method, even with nonexistent error, with an average of 35%.

Fig. 7 shows the average performance of the different methods for all the error injection scenarios. The results of the error impact experiments show that the MAP-based classification fusion is not heavily affected by the error injection; the change for the classification rate is less than 0.1% in average for an error injection up to $\sigma = 50$ m and the rejection rate increases 0.1% in average. The effects on the other methods are more pronounced, with a change of 3% in average in classification rate for the Nearest Neighbor method and an increase of 24% in the rejection rate of the distance-based method.

These experiments show higher classification rates for the MAP and Nearest Neighbor approaches compared to the baseline majority voting approach, while maintaining comparable acceptance rates. Further research is needed on additional considerations to avoid transmission of node classifications that have low probability of being correct; it is expected that both the Nearest Neighbor method and an adapted minimum-threshold MAP-based method will easily allow for these additions.

8. Conclusions

In this paper we have introduced a data set extracted from a real-life vehicle tracking sensor network, and have explained in detail the processing and algorithms used for data conditioning and classification. It is seen in the results that although the classification rates for the available modalities are only acceptable, methods used in multisensor networks such as data fusion and decision fusion will enhance the performance of these tasks. Future research in this direction is active, and it is hoped that the data set made available here will be helpful for implementation and development.

Appendix

The data set described here is available at our website, http://www.ece.wisc.edu/~sensit/, under Research Results. Three files are available: timeseries.zip, energies.zip and events.zip.

A.1. *timeseries.zip*

This file contains all the run timeseries in their original binary recording format. No processing has been done to these files, but tools are included in the next two archives. The files are organized by runs, which one folder per run (`AAV3`, `AAV4`, `AAV5`, `AAV6`, `AAV7`, `AAV8`, `AAV9`, `AAV10`, `AAV11`, `DW1`, `DW2`, `DW3`, `DW4`, `DW5`, `DW6`, `DW7`, `DW8`, `DW9`, `DW10`, `DW11`, `DW12`).The files are named using the naming convention `sensitnn-m-xxx.txt`, where `xxx` is the run name, `nn` is the node number and `m` is the modality number (1 for acoustic, 2 for seismic and 3 for PIR).

A.2. *energies.zip*

This file contains the energy values for the different runs available. The files are organized by runs, nodes and modalities. The main directory contains the following files:

- `sitex02.exe`: Executable file to convert original binary data files to ASCII formatted files.
- `energies.m`: Matlab script to generate energy information from ASCII data files.
- `nodexy.txt`: Location information for the nodes, given in UTM coordinates.

For each run you will find a directory named after the run (`AAV3`, `AAV4`, `AAV5`, `AAV6`, `AAV7`, `AAV8`, `AAV9`, `AAV10`, `AAV11`, `DW1`, `DW2`, `DW3`, `DW4`, `DW5`, `DW6`, `DW7`, `DW8`, `DW9`, `DW10`, `DW11`, `DW12`). This directory will contain a `xxx_gt.txt` file (`xxx` being the run or directory name), which contains the ground truth information for the run, or the location information in UTM coordinates recorded every 0.75 seconds. The directory will also contain several subdirectories: one for each node (`n1`, `n2`, `n3`, `n4`, `n5`, `n6`, `n41`, `n42`, `n46`, `n47`, `n48`, `n49`, `n50`, `n51`, `n52`, `n53`, `n54`, `n55`, `n56`, `n58`, `n59`, `n60`, `n61`) and one for each modality (`acoustic_1`, `seismic_2` and `pir_3`). The node subdirectories will contain the energy files for all three modalities for that node, the detection label for the node and the timestamp file; the modality subdirectories will contain the energy files for all nodes for that modality and the timestamp file. The timestamp file is named `timestamp.txt`; the energy files are named using the convention `xxxcpann_m.txt` and the detection label files are named using the convention `xxxlabelnn.txt`, where `xxx` is the run name, `nn` is the node number and `m` is the modality number (1 for acoustic, 2 for seismic and 3 for PIR).

(1) *Extraction procedure*: To convert the binary data files into ASCII data files, you will need the `sitex02.exe` file; use the command

```
sitex02 source.dat destination.txt
```

where `source.dat` is the filename of the binary file and `destination.txt` is the filename of the output ASCII file.

To extract the energy information, run the `energies.m` script in Matlab using the command

```
energies(runname,nodes)
```

where `runname` is the run name in character vector format, and `nodes` is the vector of node numbers. This script requires the ASCII data files to be placed in a subfolder named `output`, using the naming convention `sensitnn-m-xxx.txt`, where `xxx` is the run name, `nn` is the node number and `m` is the modality number (1 for acoustic, 2 for seismic and 3 for PIR). The energy file will be saved in the `output` subfolder, using the convention `xxxcpann_m.txt`, where `xxx` is the run name, `nn` is the node number and `m` is the modality number (1 for acoustic, 2 for seismic and 3 for PIR). The script will return 0 when it runs successfully and −1 on error.

A.3. *events.zip*

This file contains the event time series and features for the different runs available.The files are organized by vehicles, runs, nodes and modalities. The main directory contains the following files:

- `acousticfeatures.m`: Matlab script to generate training and testing files from event timeseries.
- `afm_mlpatterngen.m`: Matlab script to extract feature information from acoustic event timeseries.
- `extractevents.m`: Matlab script to extract event timeseries using the complete run timeseries and the ground truth/label information.
- `extractfeatures.m`: Matlab script to extract feature information from all acoustic and seismic event timeseries for a given run and set of nodes.
- `sfm_mlpatterngen.m`: Matlab script to extract feature information from seismic event timeseries.
- `ml_train1.m`: Matlab script implementation of the Maximum Likelihood Training Module (see Section 6).
- `ml_test1.m`: Matlab script implementation of the Maximum Likelihood Testing Module (see Section 6).

- `knn.m`: Matlab script implementation of the k-Nearest Neighbor Classifier Module (see Section 6).

There are folders for the different file organizations: `run` is sorted by run, and `vehicle` is sorted by vehicle type. In `run`, for each run you will find a directory named after the run (`AAV3`, `AAV4`, `AAV5`, `AAV6`, `AAV7`, `AAV8`, `AAV9`, `AAV10`, `AAV11`, `DW2`, `DW3`, `DW4`, `DW5`, `DW6`, `DW7`, `DW8`, `DW9`, `DW10`, `DW11`, `DW12`). This directory will contain several subdirectories: one for each node that has at least one event (the possible nodes are `n1`, `n2`, `n3`, `n4`, `n5`, `n6`, `n41`, `n42`, `n46`, `n47`, `n48`, `n49`, `n50`, `n51`, `n52`, `n53`, `n54`, `n55`, `n56`, `n58`, `n59`, `n60`, `n61`) and one for each modality (`acoustic_1` and `seismic_2`). The node subdirectories will contain the timeseries and feature files for both modalities for all events in the node; the modality subdirectories contain two separate subdirectories, `timeseries`, which contains the timeseries data and `features`, which contains the feature files for all events for that run. In `vehicles`, there is a directory for each vehicle type (`AAV, DW`), which contain a subdirectory for each modality (`acoustic_1` and `seismic_2`). In turn, each one of these contains two separate subdirectories, `timeseries` which contains the timeseries data and `features` which contains the feature files for all events for that run. In all cases, the timeseries files and the feature files are named using the conventions `xxxeventnn_k_m.txt` and `xxxevfeatnn_k_m.txt` respectively, where `xxx` is the run name, `nn` is the node number, `k` is the event number and `m` is the modality number (1 for acoustic and 2 for seismic).

(1) *Extraction procedure*: The scripts require the input files (run timeseries and run labels, the latter ones included in `energies.zip` for this case) to be placed in a subfolder named `output`, using the naming convention `sensitnn-m-xxx.txt`, for the run timeseries files and `xxxlabelnn_m.txt` for the run labels, where `xxx` is the run name, `nn` is the node number and `m` is the modality number (1 for acoustic and 2 for seismic). All output files are saved in the same `output` folder.

To extract the event timeseries, run the `extractevents.m` script in Matlab using the command

```
extractevents(runname,nodes)
```

where `runname` is the run name in character vector format, and `nodes` is the vector of node numbers. The event timeseries files will be saved using the convention `xxxeventnn_k_m.txt`, where `xxx` is the run name, `nn` is the node number, `k` is the event number and `m` is the modality number (1 for acoustic and 2 for seismic).

To extract the feature files from the event timeseries, run the `extractfeatures.m` script in Matlab using the command

```
extractfeatures(runname,nodes,type)
```

where `runname` is the run name in character vector format, `nodes` is the vector of node numbers, and `type` is a character defining the vehicle type for the given run 'a' for AAV, 'd' for DW and 'h' for HMMWV. The energy file will be saved using the convention `xxxevfeatnn_k_m.txt`, where `xxx` is the run name, `nn` is the node number, `k` is the event number and `m` is the modality number (1 for acoustic and 2 for seismic).

All scripts will return 0 when it runs successfully and −1 on error.

(2) *Notes*: No DW1 event files were extracted because of the mismatch in the initial timestamp between modalities.

A.4. Script customization

All scripts included with the data series can be customized to suit different feature extraction parameters, vehicle selections and name conventions. Basic Matlab proficiency is required to understand and customize the processing scripts.

References

[1] R.R. Brooks, S.S. Iyengar, Multi-Sensor Fusion: Fundamentals and Applications With Software, Prentice-Hall PTR, Upper Saddle River, NJ, 1998.

[2] H.C. Choe, R.E. Karlsen, G.R. Gerhart, T. Meitzler, Wavelet-based ground vehicle recognition using acoustic signals, Proceedings of the SPIE, 1996.

[3] R. Duda, P. Hart, D. Stork, Pattern Classification, Wiley, New York, 2001.

[4] K.B. Eom, Analysis of acoustic signatures from moving vehicles using time-varying autoregressive models, Multidimensional Systems Signal Process. 10 (1999) 357–378.

[5] D. Estrin, D. Culler, K. Pister, G. Sukhatme, Connecting the physical world with pervasive networks, IEEE Pervasive Comput. 1 (1) (2002) 59–69.

[6] D. Estrin, L. Girod, G. Pottie, M. Srivastava, Instrumenting the world with wireless sensor network, Proceedings of ICASSP'2001, Salt Lake City, UT, 2001, pp. 2675–2678.

[7] D. Li, K.D. Wong, Y.H. Hu, A.M. Sayeed, Detection, classification and tracking of targets, IEEE Signal Process. Mag. 19 (2002) 17–29.

[8] W. Merrill, K. Sohrabi, L. Girod, J. Elson, F. Newberg, W. Kaiser, Open standard development platforms for distributed sensor networks, Proc. SPIE—Unattended Ground Sensor Technologies and Applications IV 4743 (2002) 327–337.

[9] A.Y. Nooralahiyan, M. Dougherty, D. McKeown, H.R. Kirkby, A field trial of acoustic signature analysis for vehicle classification, Transport. Res. Part C 5C (1997) 165–177.

[10] C. Savarese, J.M. Rabaey, J. Reutel, Localization in distributed Ad-hoc wireless sensor networks, Proceedings of ICASSP'2001, Salt Lake City, UT, 2001, pp. 2037–2040.

[11] J.F. Scholl, L.P. Clare, J.R. Agre, Seismic attenuation characterization using tracked vehicles, Proceedings of the Meeting of the MSS Specialty Group on Battlefield, Acoustic and Seismic Sensing, 1999.

[12] R.T. Sokolov, J.C. Rogers, Removing harmonic signal nonstationarity by dynamic resampling, Proceedings of the IEEE International Symposium on Industrial Electronics, 1995.

[13] D.W. Thomas, B.R. Wilkins, The analysis of vehicle sounds for recognition, Pattern Recognition 4 (1972) 379–389.

[14] H. Wu, M. Siegel, P. Khosla, Vehicle sound signature recognition by frequency vector principal component analysis, IEEE Trans. Instrum. Meas. 48 (1999) 1005–1009.

Further reading

A.Z. Averbuch, V.A. Zheludev, I. Kozlov, Wavelet based algorithm for acoustic detection of moving ground and airborne targets, Proceedings of the SPIE, 2000.

D. Li, Y.H. Hu, Energy based collaborative source localization using acoustic micro-sensor array, J. Appl. Signal Process. (to appear).

D. Middleton, Selection of advanced technologies for detection of trucks, Proceedings of the SPIE, 1998.

N. Srour, Back propagation of acoustic signature for robust target identification, Proceedings of the SPIE, 2001.

G. Succi, T.K. Pedersen, R. Gampert, G. Prado, Acoustic target tracking and target identification-recent results, Proceedings of the SPIE, 1999.

T.L. Tung, Y. Kung, Classification of vehicles using nonlinear dynamics and array processing, Proceedings of the SPIE, 1999.