Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/pr

Wavelet-based feature extraction using probabilistic finite state automata for pattern classification[☆]

Xin Jin, Shalabh Gupta, Kushal Mukherjee, Asok Ray [*]

Department of Mechanical Engineering, The Pennsylvania State University, University Park, PA 16802, USA

ARTICLE INFO

Article history:
Received 13 July 2010
Received in revised form
2 November 2010
Accepted 4 December 2010
Available online 16 December 2010

Keywords:
Time series analysis
Symbolic dynamics
Feature extraction
Pattern classification
Probabilistic finite state automata

ABSTRACT

Real-time data-driven pattern classification requires extraction of relevant features from the observed time series as low-dimensional and yet information-rich representations of the underlying dynamics. These low-dimensional features facilitate *in situ* decision-making in diverse applications, such as computer vision, structural health monitoring, and robotics. Wavelet transforms of time series have been widely used for feature extraction owing to their time–frequency localization properties. In this regard, this paper presents a symbolic dynamics-based method to model surface images, generated by wavelet coefficients in the *scale-shift* space. These symbolic dynamics-based models (e.g., probabilistic finite state automata (PFSA)) capture the relevant information, embedded in the sensor data, from the associated Perron-Frobenius operators (i.e., the state-transition probability matrices). The proposed method of pattern classification has been experimentally validated on laboratory apparatuses for two different applications: (i) early detection of evolving damage in polycrystalline alloy structures, and (ii) classification of mobile robots and their motion profiles.

1. Introduction

Tools for real-time data-driven pattern classification facilitate performance monitoring of complex dynamical systems if the physics-based models are either inadequate or unavailable [1]. In this regard, a critical issue is real-time analysis of time series for information compression into low-dimensional feature vectors that capture the relevant information of the underlying dynamics [2–4]. Time series analysis is a challenging task if the data set is voluminous (e.g., collected at a fast sampling rate), high-dimensional, and noise-contaminated. In general, the success of data-driven pattern classification tools depends on the quality of feature extraction from the observed time series. To this end, several feature extraction tools, such as principal component analysis (PCA) [3], independent component analysis (ICA) [5], kernel PCA [6], dynamic time warping [7], derivative time series segment approximation [8], artificial neural networks (ANN) [9], hidden Markov models (HMM) [10], and wavelet transforms [11–13] have been reported in technical literature. Wavelet packet decomposition (WPD) [12] and fast wavelet transform (FWT) [13] have been used for extracting rich problem-specific information from sensor signals. Feature extraction is followed by pattern classification (e.g., using support vector machines (SVM)) [4,14].

Recently, the concepts of Symbolic Dynamics [15] have been used for information extraction from time series in the form of symbol sequences [1,16]. Keller and Lauffer [17] used tools of symbolic dynamics for analysis of time series data to visualize qualitative changes of electroencephalography (EEG) signals related to epileptic activity. Along this line, a real-time data-driven pattern identification tool, called *Symbolic Dynamic Filtering* (SDF) [18,19], has been built upon the concepts of Symbolic Dynamics, Information Theory, and Statistical Mechanics [1]. In the SDF method, time series data are converted to symbol sequences by appropriate partitioning [19]. Subsequently, a probabilistic finite-state automata (PFSA) [18] are constructed from these symbol sequences that capture the underlying system's behavior by means of information compression into the corresponding state-transition probability matrices. SDF-based pattern identification algorithms have been shown by experimental validation in the laboratory environment to yield superior performance over several existing pattern recognition tools (e.g., PCA, ANN, Bayesian Filtering, Particle Filter, Unscented Kalman Filtering, and Kernel Regression Analysis) in terms of early detection of small changes in the statistical characteristics of the observed time series [20].

Partitioning of time series is a crucial step for symbolic representation of sensor signals. To this end, several partitioning techniques have been reported in literature, such as *symbolic false*

☆ This work has been supported by the U.S. Office of Naval Research under Grant No. N00014-09-1-0688, and the U.S. Army Research Laboratory and the U.S. Army Research Office under Grant No. W911NF-07-1-0376. Any opinions, findings and conclusions expressed in this publication are those of the authors and do not necessarily reflect the views of the sponsoring agencies.

* Corresponding author. Tel.: +1 814 865 6377.
E-mail addresses: xuj103@psu.edu (X. Jin), szg107@psu.edu (S. Gupta), kum162@psu.edu (K. Mukherjee), axr2@psu.edu (A. Ray).

Nomenclature

α	scale in wavelet transform
Δt	sampling interval
ℓ	length of window $\mathcal{W}$
$\mathbf{p}$	feature vector (state probability vector)
$\mathbf{x}_k$	time series data collected at instant k
$\mathcal{C}$	set of class labels
$\mathcal{H}$	coordinate set denoting the scale-shift data points
$\mathcal{M}$	mapping for conversion of time series to symbol sequence
$\mathcal{O}$	reduction of the state set $\mathcal{Q}$
$\mathcal{Q}$	set of all possible states in a window $\mathcal{W} \subset \mathcal{H}$
$\mathcal{R}$	interval spanning the range of wavelet coefficient amplitudes
$\mathcal{S}$	wavelet surface profile
$\mathcal{S}_\Sigma$	map defining symbolization of a wavelet surface profile
$\mathcal{W}$	two-dimensional window of size $\ell \times \ell$ to convert a symbol block to a state
μ	anomaly measure
Ω	compact region in the phase space of the continuously varying dynamical systems
Φ_i	mutually exclusive and exhaustive cells in Ω
Π	state transition matrix
$\psi_{s,\tau}$	wavelet basis function ψ with scale factor s and time shift τ
Σ	symbol alphabet
σ_k	symbol generated by $\mathcal{M}$ from time series $\mathbf{x}_k$
τ_i	slow-time epoch i
$\tilde{f}$	noise-corrupted version of original signal f
B	signal bandwidth
d	divergence measure
F_c	center frequency that has the maximum modulus in the Fourier transform of the wavelet
f_p	pseudo-frequency for scale generation
f_s	sampling frequency
$F_{s,\tau}$	wavelet transform of function $f(t) \in \mathbb{H}$, where $\mathbb{H}$ is a Hilbert space
k	level of noise contaminating the signal f
m	number of most probable symbols
$O(N)$	time complexity of an algorithm to complete a problem of size N
o_i	element of the reduced set $\mathcal{O}$
P	map defining the partitioning of interval $\mathcal{R}$
q	state of a symbol block formed by the window $\mathcal{W}$
w	additive white gaussian noise with zero mean and unit variance

nearest neighbor partitioning (SFNNP) [21], *wavelet-transformed space partitioning* (WTSP) [22], and *analytic signal space partitioning* (ASSP) [23]. In particular, the wavelet transform-based method is well-suited for time–frequency analysis of non-stationary signals, noise attenuation, and reduction of spurious disturbances from the raw time series data without any significant loss of pertinent information [24]. In essence, WTSP is suitable for analyzing the noisy signals, while SFNNP and ASSP may require additional preprocessing of the time series for denoising. However, the wavelet transform of time series introduces two new domain parameters (i.e., scale and shift), thereby generating an image of wavelet coefficients. Thus, the (one-dimensional) time series data is transformed into a (two-dimensional) image of wavelet coefficients. In this context, for SDF-based analysis of wavelet-transformed data, the prior work [18,22] suggested stacking of the wavelet coefficients from multiple scales to represent the two-dimensional scale-shift wavelet domain by a one-dimensional data sequence. However, this procedure of conversion of the two-dimensional domain to one-dimensional is non-unique and potentially lossy depending on the choice of the stacking procedure.

This paper extends the concept of SDF for feature extraction in the (two-dimensional) scale-shift domain of wavelet transform without any need for non-unique conversion to one-dimensional sequence. In addition, the proposed method is potentially applicable for analysis of regular images for feature extraction and pattern classification. From these perspectives, the major contributions of the paper are as follows:

1. Development of an SDF-based feature extraction method for analysis of two-dimensional data (e.g., regular images and wavelet transform of time series in the scale-shift wavelet domain).
2. Experimental validation (in the laboratory environment) of the feature extraction method for pattern classification in two different applications:
   (i) Early detection of damage in structural materials for timely prediction of forthcoming failures.
   (ii) Behavior recognition in mobile robots by identification of their type and motion profiles.

The paper is organized into seven sections including the present one. Section 2 briefly describes the concepts of symbolic dynamic filtering (SDF) and its application to wavelet-transformed data. Section 3 presents the procedure of feature extraction from the symbolized wavelet image by construction of a probabilistic finite state automaton (PFSA). Section 4 describes the pattern classification algorithms. Sections 5 and 6 present the experimental validations on two applications: (i) early detection of failures in polycrystalline alloys and (ii) classification of mobile robots and their motion profiles, respectively. The paper is concluded in Section 7 along with recommendations for future research.

2. Symbolic dynamics and encoding

This section presents the underlying concepts of symbolic dynamic filtering (SDF) for feature extraction from time series of sensor signals. Details of SDF have been reported in previous publications for analysis of (one-dimensional) time series [18,19]. A Statistical Mechanics concept of time series analysis using symbolic dynamics has been presented in [1]. This section briefly reviews the concepts of SDF for self-sufficiency of the paper and then presents the extension for analysis of (two-dimensional) wavelet images for feature extraction. The major steps of the SDF method for feature extraction are delineated as follows:

1. Encoding (possibly nonlinear) system dynamics from observed sensor data (e.g., time series and images) for generation of symbol sequences.
2. Information compression via construction of probabilistic finite state automata (PFSA) from the symbol sequences to generate feature vectors that are representatives of the underlying dynamical system's behavior.

2.1. Review of symbolic dynamics

In the symbolic dynamics literature [15], it is assumed that the observed sensor time series from a dynamical system are represented as a symbol sequence. Let Ω be a compact (i.e., closed and totally bounded) region in the phase space of the continuously varying dynamical system, within which the observed time series is confined [18,19]. The region Ω is partitioned into $|\Sigma|$ cells $\{\Phi_0,\ldots,\Phi_{|\Sigma|-1}\}$ that are mutually exclusive (i.e., $\Phi_j \cap \Phi_k = \emptyset$ $\forall j \neq k$) and exhaustive (i.e., $\bigcup_{j=0}^{|\Sigma|-1} \Phi_j = \Omega$), where Σ is the *symbol alphabet* that labels the partition cells. A trajectory of the dynamical system is described by the discrete time series data as: $\{\mathbf{x}_0,\mathbf{x}_1,\mathbf{x}_2,\ldots\}$, where each $\mathbf{x}_i \in \Omega$. The trajectory passes through or touches one of the cells of the partition; accordingly the corresponding symbol is assigned to each point $\mathbf{x}_i$ of the trajectory as defined by the mapping $\mathcal{M}: \Omega \rightarrow \Sigma$. Therefore, a sequence of symbols is generated from the trajectory starting from an initial state $\mathbf{x}_0 \in \Omega$, such that

$$\mathbf{x}_0 \rightarrowtail \sigma_0\sigma_1\sigma_2\ldots\sigma_k\ldots, \tag{1}$$

where $\sigma_k \triangleq \mathcal{M}(\mathbf{x}_k)$ is the symbol at instant k. (Note: The mapping in Eq. (1) is called *Symbolic Dynamics* if it attributes a legal (i.e., physically admissible) symbol sequence to the system dynamics starting from an initial state.) The next subsection describes how the time series are transformed into wavelet images in scale-shift domain for generation of symbolic dynamics.

2.2. *Transformation of time series to wavelet domain*

A crucial step in symbolic dynamic filtering [18,19] is partitioning of the data space for symbol sequence generation [16]. Various partitioning techniques have been suggested in literature for symbol generation, which include variance-based [25], entropy-based [26], and hierarchial clustering-based [27] methods. A survey of clustering techniques is provided in [2]. Another partitioning scheme, based on *symbolic false nearest neighbors* (SFNN), was reported by Kennel and Buhl [21]. These techniques rely on partitioning the phase space and may become cumbersome and extremely computation-intensive if the dimension of the phase space is large. Moreover, if the data set is noise-corrupted, then the symbolic false neighbors would rapidly grow in number and require a large symbol alphabet to capture the pertinent information. Therefore, symbolic sequences as representations of the system dynamics should be generated by alternative methods because phase-space partitioning might prove to be a difficult task. Technical literature has suggested appropriate transformation of the signal before employing the partitioning method for symbol generation [18]. One such technique is the *analytic-signal-space partitioning* (ASSP) [23] that is based on the analytic signal which provides the additional phase information in the sensor data. The wavelet-transformed space partitioning (WTSP) [22] is well-suited for time–frequency analysis of non-stationary signals, noise attenuation, and reduction of spurious disturbances from the raw time series data without any significant loss of pertinent information [24,19]. Since SFNNP and ASSP may require additional preprocessing of the time series for denoising, this paper has used WTSP for construction of symbolic representations of sensor data as explained below.

In wavelet-based partitioning, time series are first transformed into the wavelet domain, where wavelet coefficients are generated at different shifts and scales. The choice of the wavelet basis function and wavelet scales depends on the time–frequency characteristics of individual signals [19]. The wavelet transform of a function $f(t) \in \mathbb{H}$ is given by

$$F_{s,\tau} = \frac{1}{\sqrt{\alpha}} \int_{-\infty}^{\infty} f(t)\psi_{s,\tau}^{*}(t)\,dt, \tag{2}$$

where $s>0$ is the scale, τ is the time shift, $\mathbb{H}$ is a Hilbert space, $\psi_{s,\tau}(t) = \psi((t-\tau)/s)$ and $\psi \in \mathbf{L}_2(\mathbb{R})$ is such that $\int_{-\infty}^{\infty} \psi(t)\,dt = 0$ and $\|\psi\|_2 = 1$.

Wavelet preprocessing of sensor data for symbol sequence generation helps in noise mitigation. Let $\tilde{f}$ be a noise-corrupted version of the original signal f expressed as

$$\tilde{f} = f + kw, \tag{3}$$

where w is additive white gaussian noise with zero mean and unit variance and k is the noise level. The noise part in Eq. (3) would be reduced if the scales over which coefficients are obtained are properly chosen.

For every wavelet, there exists a certain frequency called the center frequency F_c that has the maximum modulus in the Fourier transform of the wavelet. The pseudo-frequency f_p of the wavelet at a particular scale α is given by the following formula [28]:

$$f_p = \frac{F_c}{\alpha \Delta t}, \tag{4}$$

where Δt is the sampling interval. Then the scales can be calculated as follows:

$$\alpha^i = \frac{F_c}{f_p^i \Delta t}, \tag{5}$$

where $i=1,2,\ldots$, and f_p^i are the frequencies that can be obtained by choosing the locally dominant frequencies in the Fourier transform. The maximum pseudo-frequency $f_p^{\max}$ should not exceed the Nyquist frequency [28]. Therefore, the sampling frequency f_s for acquisition of time series data should be selected at least twice the larger of the maximum pseudo-frequency $f_p^{\max}$ and the signal bandwidth B, i.e., $f_s \geq 2\max(f_p^{\max}, B)$.

Fig. 1 shows an illustrative example of transformation of the (one-dimensional) time series in Fig. 1(a) to a (two-dimensional) wavelet image in Fig. 1(b). The amplitudes of the wavelet coefficients over the scale-shift domain are plotted as a surface. Subsequently, symbolization of this wavelet surface leads to the formation of a symbolic image as shown in Fig. 1(c).

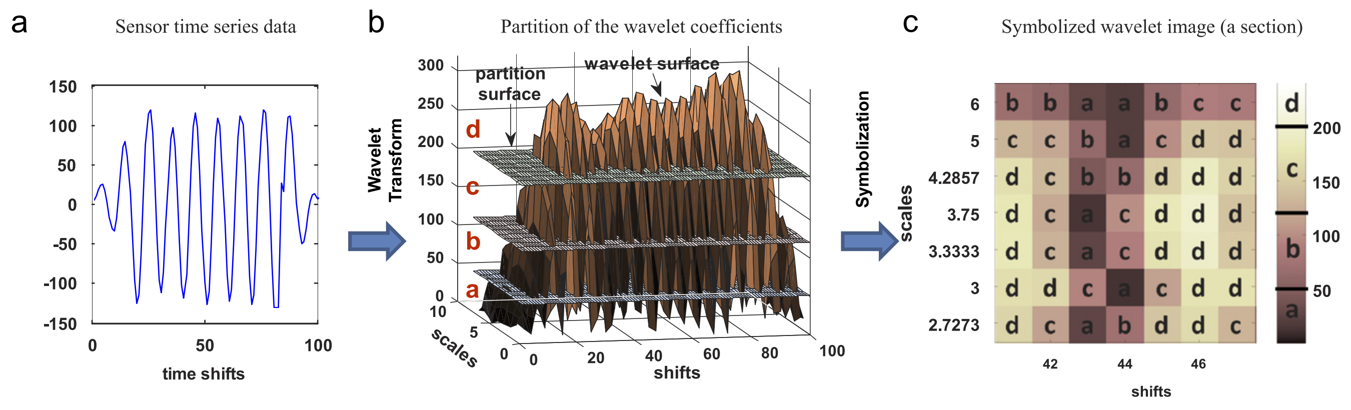

Fig. 1. Symbol image generation via wavelet transform of the sensor time series data and partition of the wavelet surface in ordinate direction.

2.3. Symbolization of wavelet surface profiles

This section presents partitioning of the wavelet surface profile in Fig. 1(b), which is generated by the coefficients over the two-dimensional scale-shift domain, for construction of the symbolic image in Fig. 1(c). The *x*–*y* coordinates of the wavelet surface profiles denote the shifts and the scales, respectively, and the *z*-coordinate (i.e., the surface height) denotes the pixel values of wavelet coefficients.

Definition 1 (*Wavelet Surface Profile*). Let $\mathcal{H} \triangleq \{(i,j) : i,j \in \mathbb{N}, 1 \le i \le m, 1 \le j \le n\}$ be the set of coordinates consisting of $(m \times n)$ pixels denoting the scale-shift data points. Let $\mathcal{R}$ denote the interval that spans the range of wavelet coefficient amplitudes. Then, a wavelet surface profile is defined as

$$\mathcal{S} : \mathcal{H} \to \mathcal{R}. \tag{6}$$

Definition 2 (*Symbolization*). Given the symbol alphabet Σ, let the partitioning of the interval $\mathcal{R}$ be defined by a map $P : \mathcal{R} \to \Sigma$. Then, the symbolization of a wavelet surface profile is defined by a map $\mathcal{S}_\Sigma \equiv P \circ \mathcal{S}$ such that

$$\mathcal{S}_\Sigma : \mathcal{H} \to \Sigma \tag{7}$$

that labels each pixel of the image to a symbol in Σ.

The wavelet surface profiles are partitioned such that the ordinates between the maximum and minimum of the coefficients along the *z*-axis are divided into regions by different planes parallel to the *x*–*y* plane. For example, if the alphabet is chosen as $\Sigma = \{a,b,c,d\}$, i.e., $|\Sigma| = 4$, then three partitioning planes divide the ordinate (i.e., *z*-axis) of the surface profile into four mutually exclusive and exhaustive regions, as shown in Fig. 1(b). These disjoint regions form a partition, where each region is labeled with one symbol from the alphabet Σ. If the intensity of a pixel is located in a particular region, then it is coded with the symbol associated with that region. As such, a symbol from the alphabet Σ is assigned to each pixel corresponding to the region where its intensity falls. Thus, the two-dimensional array of symbols, called *symbol image*, is generated from the wavelet surface profile, as shown in Fig. 1(c).

The surface profiles are partitioned by using either the maximum entropy partitioning (MEP) or the uniform partitioning (UP) methods [22,19]. If the partitioning planes are separated by equal-sized intervals, then the partition is called the *uniform partitioning* (UP). Intuitively, it is more reasonable if the information-rich regions of a data set are partitioned finer and those with sparse information are partitioned coarser. To achieve this objective, the *maximum entropy partitioning* (MEP) method has been adopted in this paper such that the entropy of the generated symbols is maximized. The procedure for selection of the alphabet size $|\Sigma|$, followed by generation of a MEP, has been reported in [19]. In general, the choice of alphabet size depends on specific data set and experiments. The partitioning of wavelet surface profiles to generate symbolic representations enables robust feature extraction, and symbolization also significantly reduces the memory requirements [19].

For the purpose of pattern classification, the reference data set is partitioned with alphabet size $|\Sigma|$ and is subsequently kept constant. In other words, the structure of the partition is fixed at the reference condition and this partition serves as the reference frame for subsequent data analysis [18].

3. Feature extraction via construction of probabilistic finite-state automata

This section presents construction of a *probabilistic finite state automaton* (PFSA) for feature extraction based on the symbol image generated from a wavelet surface profile.

3.1. Conversion from symbol image to state image

For analysis of (one-dimensional) time series, a PFSA is constructed such that its states represent different combinations of blocks of symbols on the symbol sequence. The edges connecting these states represent the transition probabilities between these blocks [18,19]. Therefore, for analysis of (one dimensional) time series, the 'states' denote all possible symbol blocks (i.e., words) within a window of certain length. Let us now extend the notion of 'states' on a two-dimensional domain for analysis of wavelet surface profiles via construction of a '*state image*' from a '*symbol image*'.

Definition 3 (*State*). Let $\mathcal{W} \subset \mathcal{H}$ be a two-dimensional window of size $(\ell \times \ell)$ that is denoted as $|\mathcal{W}| = \ell^2$. Then, the state of a symbol block formed by the window $\mathcal{W}$ is defined as the configuration $q = S_\Sigma(\mathcal{W})$.

Let the set of all possible states (i.e., two-dimensional words or blocks of symbols) in a window $\mathcal{W} \subset \mathcal{H}$ be denoted as $\mathcal{Q} \triangleq \{q_1, q_2, \ldots, q_{|\mathcal{Q}|}\}$, where $|\mathcal{Q}|$ is the number of (finitely many) states. Then, $|\mathcal{Q}|$ is bounded above as $|\mathcal{Q}| \le |\Sigma|^{|\mathcal{W}|}$; the inequality is due to the fact that some of the states might have zero probability of occurrence. Let us denote $\mathcal{W}_{i,j} \subset \mathcal{H}$ to be the window where (i,j) represents the coordinates of the top-left corner pixel of the window. In this notation, $q_{i,j} = S_\Sigma(\mathcal{W}_{i,j})$ denotes the state at pixel $(i,j) \in \mathcal{H}$. Thus, every pixel $(i,j) \in \mathcal{H}$ corresponds to a particular state $q_{i,j} \in \mathcal{Q}$ on the image. Every pixel in the image $\mathcal{H}$ is mapped to a state, excluding the pixels that lie at the periphery depending on the window size. Fig. 2 shows an illustrative example of the transformation of a *symbol image* to the *state image* based on a sliding window $\mathcal{W}$ of size (2×2). This concept of state formation facilitates capturing of long range dynamics (i.e., word to word interactions) on a symbol image.

In general, a large number of states would require a high computational capability and hence might not be feasible for real-time applications. The number of states, $|\mathcal{Q}|$, increases with the window size $|\mathcal{W}|$ and the alphabet size $|\Sigma|$. For example, if $\ell = 2$ and $|\Sigma| = 4$, then the total number of states are $|\mathcal{Q}| \le |\Sigma|^{\ell^2} = 256$. Therefore, for computational efficiency, it is necessary to compress the state set $\mathcal{Q}$ to an effective reduced set $\mathcal{O} \triangleq \{o_1, o_2, \ldots, o_{|\mathcal{O}|}\}$ [19] that enables mapping of two or more different configurations in a window $\mathcal{W}$ to a single state. State compression must preserve sufficient information as needed for pattern classification, albeit possibly lossy coding of the wavelet surface profile.

In view of the above discussion, a probabilistic state compression method is employed, which chooses the m most probable symbols, from each state as a representation of that particular state. In this method, each state consisting of $\ell \times \ell$ symbols is compressed to a reduced state of length $m < \ell^2$ symbols by choosing the top m symbols that have the highest probability of occurrence arranged in

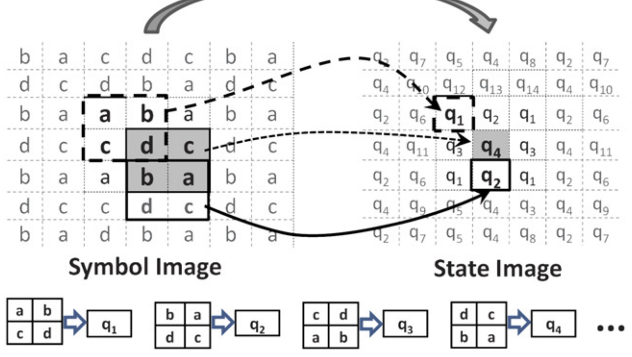

Fig. 2. Conversion of the symbol image to the state image.

descending order. If two symbols have the same probability of occurrence, then either symbol may be preferred with equal probability. This procedure reduces the state set $\mathcal{Q}$ to an effective set $\mathcal{O}$, where the total number of compressed states is given as: $|\mathcal{O}| = |\Sigma|^m$. For example, if $|\Sigma| = 4$, $|\mathcal{W}| = 4$ and $m=2$, then the state compression reduces the total number of states to $|\mathcal{O}| = |\Sigma|^m = 16$ instead of 256. This method of state compression is motivated from the renormalization methods in *Statistical Physics* that are useful in eliminating the irrelevant local information on lattice spin systems while still capturing the long range dynamics [1]. The choice of $|\Sigma|$, ℓ and m depends on specific applications and noise level as well as the available computational power, and is made by an appropriate tradeoff between robustness to noise and capability to detect small changes. For example, a large alphabet may be noise-sensitive while a small alphabet could miss the information of signal dynamics [19].

3.2. Construction of PFSA

A probabilistic finite state automaton (PFSA) is constructed such that the states of the PFSA are the elements of the compressed state set $\mathcal{O}$ and the edges are the transition probabilities between these states. Fig. 3 shows an example of a typical PFSA with four states. The transition probabilities between states are defined as

$$\wp(o_k|o_l) = \frac{N(o_l,o_k)}{\sum_{k'=1,2,\ldots,|\mathcal{O}|} N(o_l,o_{k'})} \quad \forall o_l,o_k \in \mathcal{O}, \tag{8}$$

where $N(o_l,o_k)$ is the total count of events when o_k occurs adjacent to o_l in the direction of motion. The calculation of these transition probabilities follows the principle of sliding block code [15]. A transition from the state o_l to the state o_k occurs if o_k lies adjacent to o_l in the positive direction of motion. Subsequently, the counter moves to the right and to the bottom (row-wise) to cover the entire state image, and the transition probabilities $\wp(o_k|o_l)$, $\forall o_l,o_k \in \mathcal{O}$ are computed using Eq. (9). Therefore, for every state on the state image, all state-to-state transitions are counted, as shown in Fig. 4. For example, the dotted box in the bottom-right corner contains three adjacent pairs, implying the transitions $o_1 \to o_2$, $o_1 \to o_3$, and $o_1 \to o_4$ and the corresponding counter of occurrences $N(o_1,o_2)$, $N(o_1,o_3)$ and $N(o_1,o_4)$, respectively, are increased by one. This procedure generates the stochastic state-transition probability matrix of the PFSA given as

$$\Pi = \begin{bmatrix} \wp(o_1|o_1) & \ldots & \wp(o_{|\mathcal{O}|}|o_1) \\ \vdots & \ddots & \vdots \\ \wp(o_1|o_{|\mathcal{O}|}) & \ldots & \wp(o_{|\mathcal{O}|}|o_{|\mathcal{O}|}) \end{bmatrix}, \tag{9}$$

where $\Pi \equiv [\pi_{jk}]$ with $\pi_{jk} = \wp(o_k|o_j)$. Note: $\pi_{jk} \geq 0$ $\forall j,k \in \{1,2,\ldots|\mathcal{O}|\}$ and $\sum_k \pi_{jk} = 1$ $\forall j \in \{1,2,\ldots|\mathcal{O}|\}$.

In order to extract a low-dimensional feature vector, the stationary state probability vector $\mathbf{p}$ is obtained as the left eigenvector corresponding to the (unique) unity eigenvalue of the (irreducible) stochastic transition matrix Π. The state probability vectors $\mathbf{p}$ serve as the '*feature vectors*' and are generated from different data sets from the corresponding state transition matrices. These feature vectors are also denoted as '*patterns*' in this paper.

3.3. Summary of SDF for feature extraction

The major steps of SDF for feature extraction are summarized below:

- Acquisition of time series data from appropriate sensor(s) and signal conditioning as necessary.
- Wavelet transform of the time series data with appropriate scales to generate the wavelet surface profile.
- Partitioning of the wavelet surface profile and generation of the corresponding symbol image.
- Conversion from symbol image to state image via probabilistic state compression strategy.
- Construction of PFSA and computation of the state transition matrices that in turn generate the state probability vectors as the feature vectors (i.e., patterns).

The advantages of SDF for feature extraction and subsequent pattern classification are summarized below:

- Robustness to measurement noise and spurious signals.
- Adaptability to low-resolution sensing due to the coarse graining in space partitions [18].
- Capability for detection of small deviations because of sensitivity to signal distortion.
- Real-time execution on commercially available inexpensive platforms.

4. Pattern classification using SDF-based features

Once the feature vectors are extracted in a low-dimensional space from the observed sensor time series, the next step is to classify these patterns into different categories based on the particular application. Technical literature abounds in diverse methods of pattern classification, such as divergence measure, k-nearest neighbor (k-NN) algorithm [29], support vector machine (SVM) [14], and artificial neural network (ANN) [30]. The main focus of this paper is to develop and validate the tools of Symbolic Dynamic Filtering (SDF) for feature extraction from wavelet surface profiles generated from sensor time series data. Therefore, the SDF method for feature extraction is used in conjunction with the standard pattern classification algorithms, as described in the experimental validation sections.

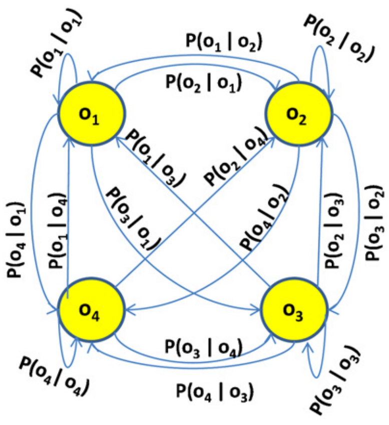

Fig. 3. An example of a 4-state PFSA.

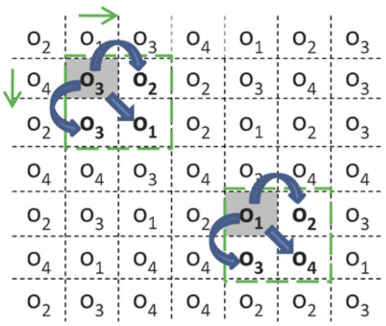

Fig. 4. An example of feature extraction from the state image.

Pattern classification using SDF-based features is posed as a two-stage problem, i.e., the training stage and the testing stage. The sensor time series data sets are divided into three groups: (i) partition data, (ii) training data, and (iii) testing data. The partition data set is used to generate partition planes that are used in the training and the testing stages. The training data set is used to generate the training patterns of different classes for the pattern classifier. Multiple sets of training data are obtained from independent experiments for each class in order to provide a good statistical spread of patterns. Subsequently, the class labels of the testing patterns are generated from testing data in the testing stage. The partition data sets may be part of the training data sets, whereas the training data sets and the testing data sets must be mutually exclusive.

Fig. 5 depicts the flow chart of the proposed algorithm that is constructed based on the theory of SDF. The partition data is wavelet-transformed with appropriate scales to convert the one-dimensional numeric time series data into the wavelet image. The corresponding wavelet surface is analyzed using the *maximum entropy principle* [22,19] to generate the partition planes that remain invariant for both the training and the testing stage. The scales used in the wavelet transform of the partitioning data also remain invariant during the wavelet transform of the training and the testing data. In the training stage, the wavelet surfaces are generated by transformation of the training data sets corresponding to different classes. These surfaces are symbolized using the partition planes to generate the symbol images. Subsequently, PFSAs are constructed based on the corresponding symbol images, and the training patterns (i.e., state probability vectors $\mathbf{p}$ or state transition matrices Π) are extracted from these PFSAs. Similar to the training stage, the PFSA and the associated pattern is generated for different data sets in the testing stage. These patterns are then classified into different classes using pattern classifier, such as SVM, k-NN and ANN.

Consider a classification problem of $|\mathcal{C}|$ classes, where $\mathcal{C}$ is the set of class labels. In the training stage, feature vectors $\mathbf{p}_j^{\mathcal{C}_i}, j=1,2,\ldots,n_i$ are generated from the training data sets of class $\mathcal{C}_i$, where n_i is the number of samples in class $\mathcal{C}_i$. The same procedure is carried out for all other classes. In the testing stage, a testing feature vector $\mathbf{p}_{test}$ with unknown class labels is generated using SDF. Two examples of using the pattern classifiers with SDF are provided here. For k-NN algorithm, the estimated class label of a testing feature vector $\mathbf{p}_{test}$ is equal to the most frequent class among the k-nearest training features [29]. For SVM, a separating hyperplane/hypersurface is generated based on training feature vectors ($\mathbf{p}_j^{\mathcal{C}_i}, j=1,2,\ldots,n_i$). The estimated class label of the testing feature vector $\mathbf{p}_{test}$ depends on which side of the hyperplane/hypersurface the testing feature vector falls [14].

5. Application I: damage detection in structural materials

Behavioral pattern changes may take place in dynamical systems due to accumulation of faults and progression of anomalies (i.e., deviations of the evolving patterns from the nominal pattern). The pattern changes are characterized by a scalar-valued non-negative function, called *anomaly measure* (μ). In this application, efficacy of the symbolic dynamic filtering (SDF) tool is demonstrated for damage detection in polycrystalline alloys.

The proposed algorithm has been experimentally validated on a laboratory apparatus for damage detection in 7075-T6 aluminium alloy specimens [31]. The apparatus is a special-purpose uniaxial fatigue damage testing machine that is instrumented with ultrasonic flaw detectors and an optical traveling microscope, as shown in Fig. 6(a). The experimental apparatus is operated under load control or strain control at speeds up to 12.5 Hz. The tests were conducted using center notched 7075-T6 aluminium specimens at a constant amplitude sinusoidal load, where the maximum and minimum loads were kept constant at 87 and 4.85 MPa. The specimens used are 3 mm thick and 50 mm wide, and have a slot of 1.58 mm $\times$ 4.5 mm at the center. The central notch is made to increase the stress concentration factor that ensures crack initiation and propagation at the notch ends. The test apparatus is equipped with two types of sensors that have been primarily used for damage detection: traveling optical microscope and ultrasonic flaw detector [19]. Fig. 6(b) shows the aluminum specimen and an illustration of the ultrasonic flaw detector mounted on the specimen. The sampling frequency of the ultrasonic sensing device is 20 MHz, while the maximum pseudo-frequency $f_p^{\max}$ is 5 MHz (see Section 2.2).

5.1. Notion of two-time scales

Anomaly detection from sensor time series data is posed as a two-time-scale problem as depicted in Fig. 7. The *fast time scale* is related to the response time of the process dynamics. Over the span of data acquisition, the dynamic behavior of the system is assumed to remain invariant, i.e., the process is quasi-stationary at the fast time scale. On the other hand, the *slow time scale* is related to the time span over which the dynamical system undergoes non-stationary evolution due to gradual growth of anomalies. SDF detects the statistical changes in behavioral patterns over slow-time-scale epochs, that are simply referred to as *epochs* in the sequel.

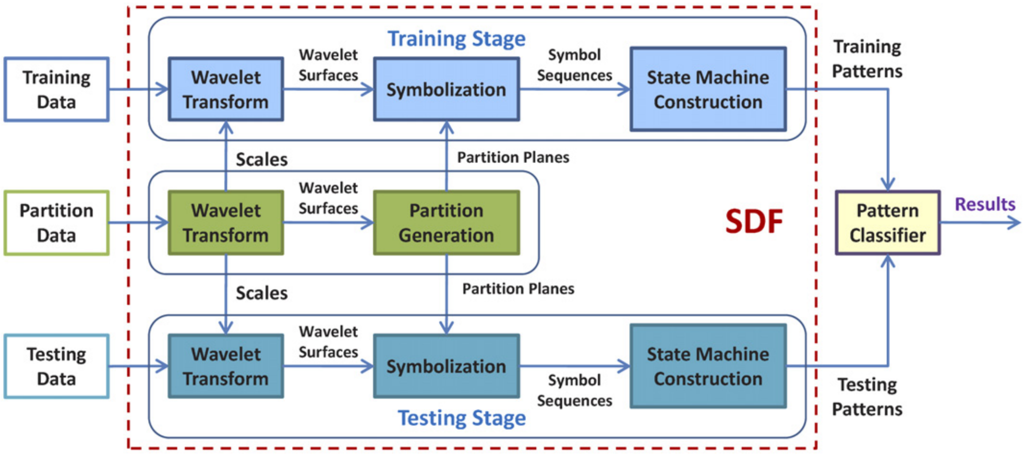

Fig. 5. Flow chart of the proposed methodology.

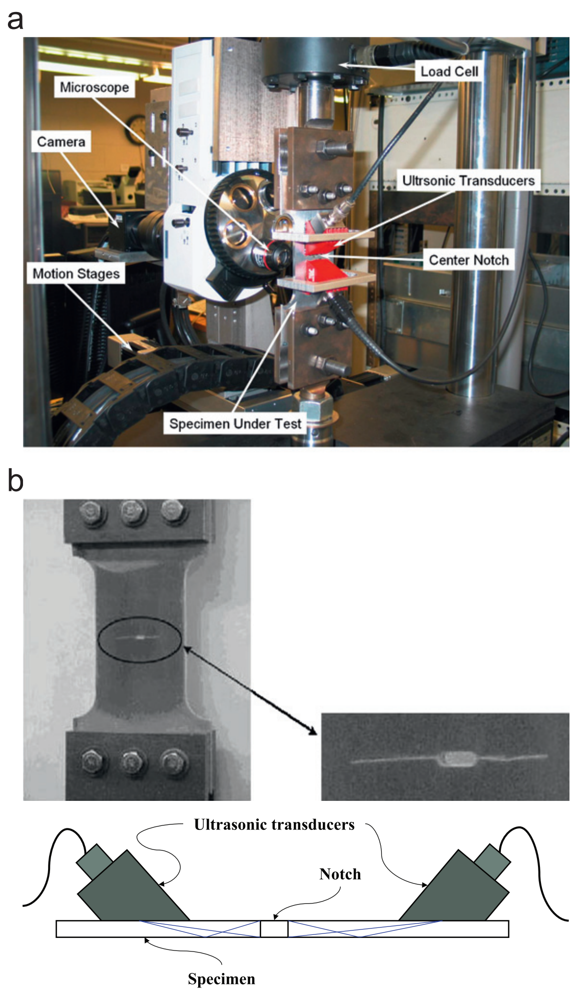

Fig. 6. Computer-instrumented apparatus for fatigue testing and schematic of ultrasonic sensors on a test specimen. (a) Fatigue testing apparatus and (b) specimen and mounting of the ultrasonic sensors.

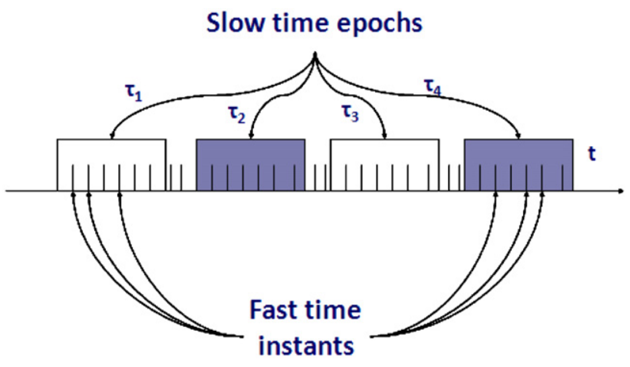

Fig. 7. Pictorial view of the two time scales: (i) *slow time scale* of anomaly evolution and (ii) *fast time instants* of data acquisition.

5.2. Experimental procedure

The ultrasonic sensing device was triggered at a frequency of 5 MHz at each peak of the ($\sim$12.5 Hz) sinusoidal load. The slow-time epochs were chosen to be 1000 load cycles (i.e., $\sim$80 s) apart. At the onset of each slow-time epoch, the ultrasonic data points were collected on the fast scale of 50 cycles (i.e., $\sim$4 s) which produced a string of 15,000 data points. It is assumed that during this fast scale, no major changes occurred in the crack behavior. The nominal condition at the slow-time epoch τ_0 was chosen to be 1.0 kcycles to ensure that the electro-hydraulic system of the test apparatus had come to a steady state and that no significant damage occurred till that point. The anomalies at subsequent slow-time epochs, $\tau_1, \tau_2, \ldots, \tau_k \ldots$, were then calculated with respect to the nominal condition at τ_0. There are in total 46 epochs, which are taken every 1 kcycle in the fatigue testing.

5.3. Pattern analysis for anomaly detection

This section explains how the anomaly detection algorithms are formulated using SDF and are compared with several other pattern recognition tools, including principal component analysis (PCA), radial basis function Neural Networks (rbfNN), and multilayer perceptron Neural Networks (mlpNN). For the anomaly detection problem, the training information is chosen to be the ultrasonic data collected at the nominal condition τ_0 that generates the reference pattern, and is also used for construction of the partitioning. Subsequently, the testing is conducted at all other time epochs for anomaly detection. Note that '*anomaly*' is defined as a continuous deviation of the pattern obtained at the current epoch with the reference pattern. In this regard, the different pattern analysis methods mentioned above are used for feature extraction from sensor data at different slow time epochs. Once the features (i.e., patterns) are obtained, then a divergence measure, as shown later in Eq. (10), is used for continuous quantification of anomaly as the distance between the extracted patterns from the nominal pattern.

For SDF the parameters are chosen as alphabet size $|\Sigma| = 8$, window size $\ell \times \ell = 3 \times 3$, and reduced state length $m=1$ (see Section 3.1); the wavelet basis function is chosen to be *gaus*3 because it closely matches the shape of the ultrasonic response signals. Absolute values of the wavelet data are used to generate the partition, because of the symmetry of the data sets about their mean. This combination of parameters is capable of capturing the anomalies significantly earlier than the images created by the optical microscope. Increasing the alphabet size $|\Sigma|$ further does not improve the results and increasing the reduced state length m creates a larger number of states of the probabilistic finite state automata (PFSA), many of them having very small or near-zero probabilities. An increased number of states also requires a larger data set at each epoch to (numerically) stabilize the state probability vectors of PFSA. The state probability vector $\mathbf{p}^0$ is obtained at the nominal condition of time epoch τ_0 and the state probability vectors $\mathbf{p}^1, \mathbf{p}^2, \ldots, \mathbf{p}^k \ldots$ are obtained at subsequent epochs $\tau_1, \tau_2, \ldots, \tau_k \ldots$.

The mlpNN consists of three hidden layers with 50 neurons in each one of them and an output layer with one neuron. Tangent sigmoid functions have been used in the hidden layers as transfer functions, while the output layer uses a linear function. On the other hand, the rbfNN uses only one hidden layer and one output layer (with one neuron). Optimal training was obtained using 100 neurons in the hidden layer. The hidden layer uses a radial basis function, whereas the output layer uses linear function as transfer functions. Several other variants of these combinations have also been used; however, these choices yielded the best results.

Several divergence measures can be defined for anomaly detection using the SDF [19] and the other pattern analysis methods. One such measure is called the anomaly measure μ that is computed from the state probability vector $\mathbf{p}$ at each time epoch τ_i such that:

$$\mu^i \equiv d(\mathbf{p}^i, \mathbf{p}^0), \tag{10}$$

where $d(\bullet,\bullet)$ is an appropriately defined divergence function and $\mathbf{p}^i$ and $\mathbf{p}^0$ are the state probability vectors for the ith and the nominal data set, respectively. A possible choice for $d(\bullet,\bullet)$ is $\mu^i \equiv \|\mathbf{p}^i - \mathbf{p}^0\|_r$ where $\|\bullet\|_r$ for $r \in [1,\infty)$ is the Hölder norm of $\bullet$. In this paper, $r=2$ (i.e., the Euclidean norm) is used. It is emphasized that the anomaly measure is relative to the nominal condition which is fixed in advance and should not be confused with the actual damage at an absolute level.

5.4. Experimental results and discussion

The three rows of plates in Fig. 8, respectively, show (i) two-dimensional microscopic images of a specimen surface, (ii) time series data from ultrasonic sensors, and (iii) corresponding histograms of probability distribution of PFSA states. The four columns of plates in Fig. 8 correspond to four different (slow-time) epochs, approximately at 1, 15, 23 and 35 kcycles, exhibiting gradual evolution of damage. In each column from (a) to (d), the top plate exhibits the surface image of the test specimen as seen by the optical microscope. As exhibited on the top plate in each of the four columns, the crack originated and developed on the right side of the notch at the center. The time series exhibited in the middle plate of each of the four columns displays the evolution in ultrasonic signals in slow time scale. The histogram in the bottom plate of each of the four columns show the state probability vector corresponding to damage growth on the test specimen at the respective (slow-time) epoch, signifying how the probability distribution gradually changes from nearly uniform distribution (i.e., minimal information) to delta distribution (i.e., maximum information).

The top plate in Fig. 8(a) shows the image at the nominal condition (i.e., 1 kcycles) with no indication of surface damage as seen by the optical microscope, which is considered to be the reference point with the anomaly measure equal to zero. The top plate in Fig. 8(b) still does not have any indication of surface crack (as seen by the optical microscope). Although the middle plate in

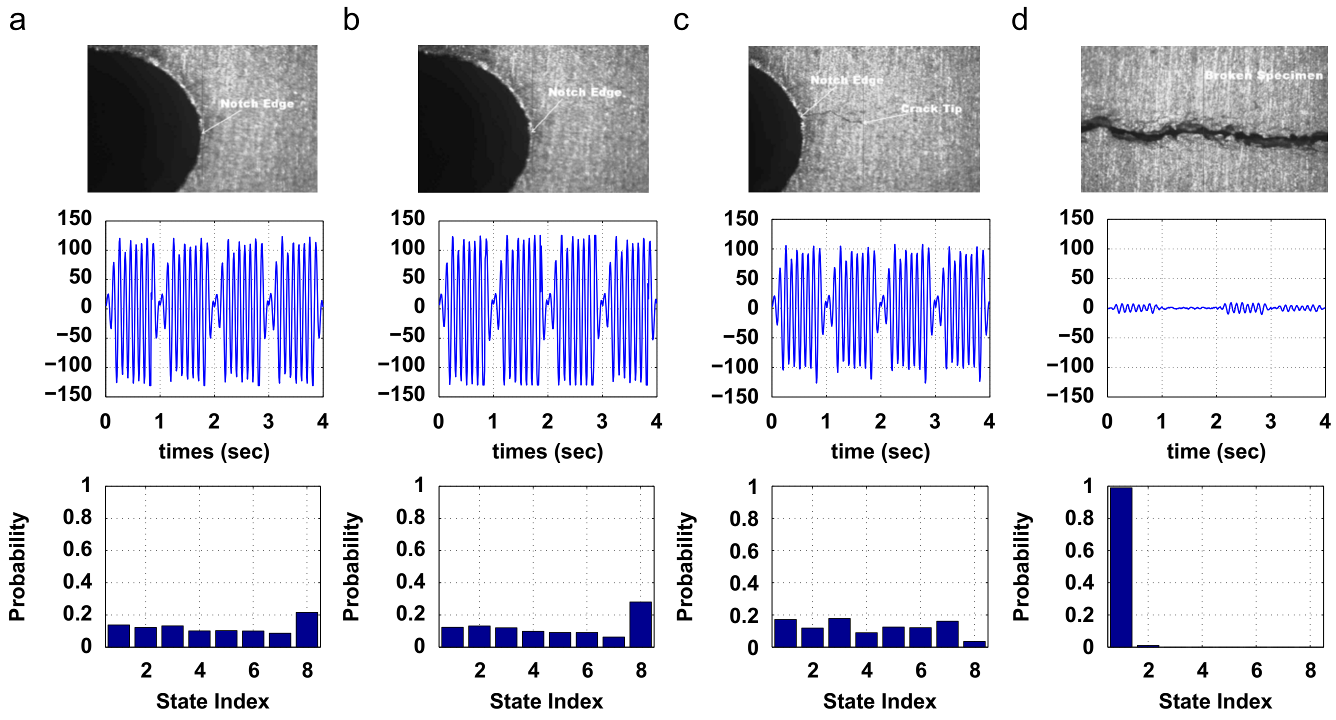

Fig. 8. Pictorial view of crack damage, corresponding ultrasonic sensor measurements and probability distribution (from top to bottom). (a) nominal condition (1 kcycles), (b) internally damaged (15 kcycles), (b) appearance of surface crack (23 kcycles), and (d) broken specimen (35 kcycles).

Fig. 8(b) indicates only an insignificant attenuation in the amplitude of the ultrasonic signal compared to the middle plate in Fig. 8(a), the bottom plate of Fig. 8(b) does exhibit a noticeable deviation from the nominal condition in the bottom plate of Fig. 8(a). This is an evidence that the analysis, based on SDF using ultrasonic sensor data, produces damage information during crack initiation, which is not available by simple optical microscopy. The top plate in Fig. 8(c) exhibits the first noticeable appearance of a crack on the specimen surface, which may be considered as the boundary of the crack initiation and crack propagation phases. The histogram of probability distribution in the bottom plate of Fig. 8(c) at 23 kcycles shows further deviation from the corresponding distribution in Fig. 8(b) at 15 kcycles. The top plate in Fig. 8(d) at 35 kcycles exhibits the image of a completely broken specimen.

Fig. 9 compares SDF with Multi-Layer Perceptron Neural Network (mlpNN), Radial Basis Function Neural Network (rbfNN), and Principal Component Analysis (PCA) for detection of anomaly patterns. Each of the normalized anomaly measure curves in Fig. 9 shows a transition from the crack initiation phase to the crack propagation phase, where the slope of the anomaly measure changes dramatically indicating onset of the crack propagation phase. First appearance of a crack on the specimen surface, as detected by the optical microscope at approximately 23 kcycles, is indicated by the dashed vertical line in Fig. 9. The critical information lies in the region to the left of the vertical line where no crack is visible on the specimen surface by the optical microscope. This is the region where multiple damage sites are possibly formed inside the specimen, which cause small changes in the ultrasonic signal profile. An abrupt change in the slope (i.e., a sharp decrease in the curvature) of the anomaly measure profile provides a clear insight into the forthcoming failure.

The family of anomaly measure profiles μ in Fig. 9 exhibits gradual increase until after the phase transition at $\sim$23 kcycles. Changes in the value of μ, its slope, and its curvature provide early warnings for a forthcoming major change in the system dynamics. From this perspective, the performance of SDF is superior to that of both types of Neural networks (i.e., rbfNN and mlpNN) and PCA, especially in the crack initiation phase that lies on the left side of the vertical line. Table 1 presents a numerical comparison of execution time and memory requirement of the afore-mentioned methods for computation of the anomaly measure μ. In each case, the execution time for a single operation cycle at a time epoch is obtained from the average of execution times for operation cycles at 46 consecutive slow-time epochs on a 2.83 GHz Quad Core CPU desktop computer in the Matlab 7.9.0 environment.

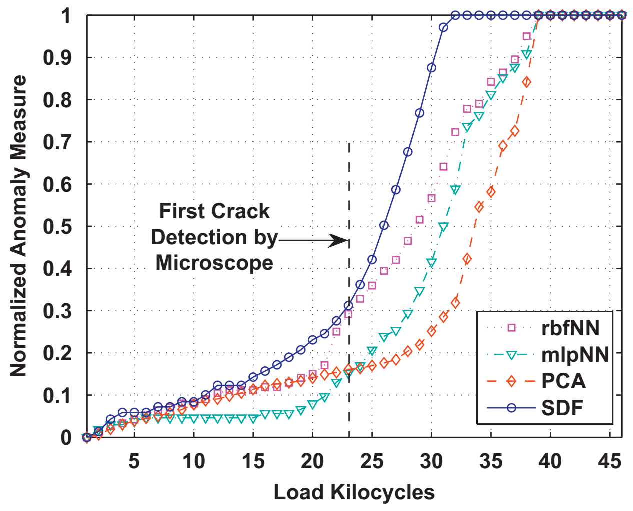

Fig. 9. Performance comparison for fatigue damage detection.

Table 1
Comparison of computational complexity of anomaly detection methods.

Anomaly detection method	Training stage		Testing stage	
	Execution time (s)	Memory requirement (MB)	Execution time (s)	Memory requirement (MB)
PCA	1.06×10^{0}	126.3	9.85×10^{-3}	87.8
SDF	3.51×10^{-1}	87.3	2.94×10^{-1}	78.5
mlpNN	3.02×10^{2}	175.0	2.20×10^{-1}	120.0
rbfNN	6.90×10^{2}	814.5	4.50×10^{-1}	95.8

Table 1 also shows that, in the testing stage, the execution time varies from about ten milliseconds for PCA to about half a second for rbfNN. All methods shown in Table 1 are implementable in real time since the execution time is much shorter than the time spent in data acquisition. SDF is the fastest among all anomaly detection methods in the training stage, and has similar execution time in the testing stage. PCA is at least one order faster than the other methods, while rbfNN, mlpNN and SDF have comparable execution time in the testing stage. As for the memory requirement, SDF requires the least memory among the four methods, as shown in Table 1. The memory requirement in other methods in the testing stage is more or less similar (around 100 MB), which is insignificant for a commercially available computer. However, for rbfNN and mlpNN, the training stage requires significantly longer time (6.90×10^2 s for rbfNN and 3.02×10^2 s for mlpNN) and more memory (814.5 MB for rbfNN and 175.00 MB for mlpNN) than the other methods.

6. Application II: classification problems in mobile robots

This section presents the experimental validation of the proposed wavelet-based feature extraction method for behavior recognition in mobile robots. The objective here is to identify the robot type and the behavior (i.e., the motion type) based on the time series data obtained from the pressure sensitive floor. These experiments are inspired from various real-life applications of pattern classification, such as (i) classification of enemy vehicles across the battlefield through analysis of seismic and acoustic time series data and (ii) classification of human and animal movements through analysis of seismic time series data.

6.1. Experimental procedure

The experimental set up consists of a wireless network incorporating mobile robots, robot simulators, and distributed sensors as shown in Figs. 10 and 11. A major component of the experimental set up is the pressure sensitive floor that consists of distributed piezoelectric wires installed underneath the floor to serve as arrays of distributed pressure sensors. A coil of piezoelectric wire is placed under a 0.65 m × 0.65 m square floor tile as shown in Fig. 11(a) such that the sensor generates an analog voltage due to pressure applied on it. This voltage is sensed by a *Brainstem*™ microcontroller using one of its 10-bit A/D channels thereby yielding sensor readings in the range of 0 to 1023. The sampling frequency of the pressure sensing device that captures the dynamics of robot motion is 10 Hz, while the maximum pseudo-frequency $f_p^{\max}$ is 4.44 Hz (see Section 2.2). A total of 144 sensors are placed in a 9 × 16 grid to cover the entire laboratory environment as shown in Fig. 11(b). The sensors are grouped into four quadrants, each being connected to a stack consisting of 8 networked *Brainstem* microcontrollers for data acquisition. The microcontrollers are, in turn, connected to two laptop computers running *Player* [32]

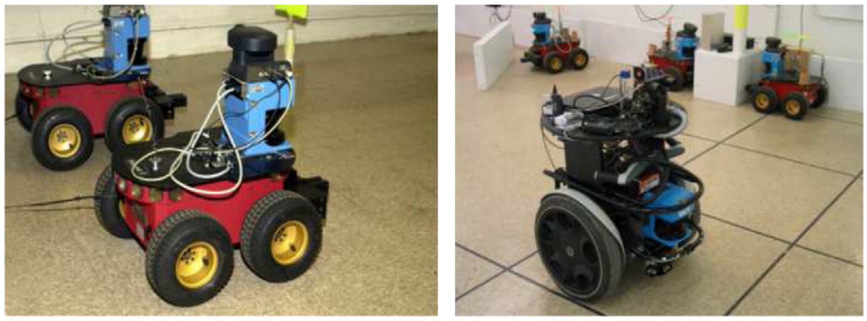

Fig. 10. The Robot Hardware: Pioneer 2AT and Segway RMP.

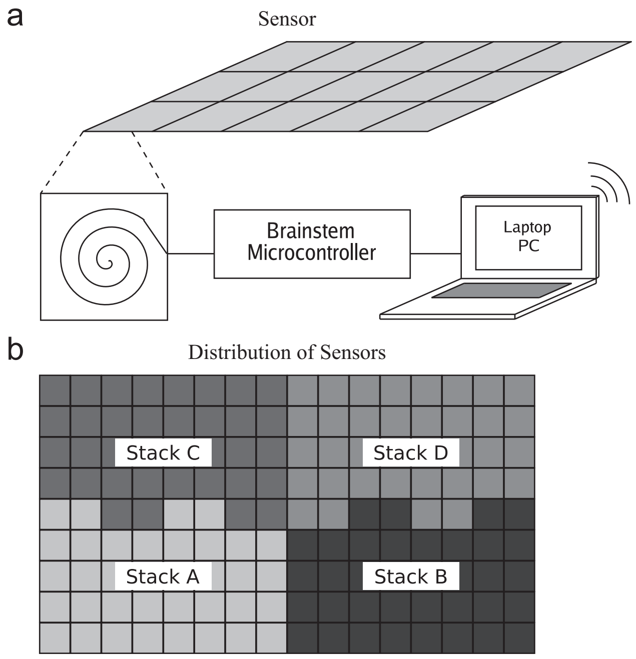

Fig. 11. Sensor layout in the laboratory environment: (a) sensor and (b) distribution of sensors.

server that collects the raw sensor data and distributes to any client over the wireless network for further processing.

Fig. 10 shows a pair of Pioneer robots and a Segway RMP that have the following features:

- Pioneer 2AT is a four-wheeled robot that is equipped with a differential drive train system and has an approximate weight of 35 kg.
- Segway RMP is a two-wheeled robot (with inverted pendulum dynamics) that has a zero turn radius and has an approximate weight of 70 kg.

Since Pioneer is lighter than Segway and Pioneer's load on the floor is more evenly distributed, their statistics are dissimilar. Furthermore, since the kinematics and dynamics of the two types of robots are different, the texture of the respective pressure sensor signals are also different.

The objective is to identify the robot type and motion type from the time series data. The Segway RMP and Pioneer 2AT robots are commanded to execute three different motion trajectories, namely, *random motion*, *circular motion* and *square motion*. Table 2 lists the parameters for the three types of robot motion. In the presence of uncertainties (e.g., sensor noise and fluctuations in robot motion), a complete solution of the robot type and motion identification

Table 2
Parameters used for various types of motion.

Motion type	Parameter	Value
Circular	Diameter	4 m
Square	Edge length	3 m
Random	Uniform distribution	*x*-dir 1 to 7 m y-dir 1 to 4 m

problem may not be possible in a deterministic setting because the patterns would not be identical for similar robots behaving similarly. Therefore, the problem is posed in the statistical setting, where a family of patterns is generated from multiple experiments conducted under identical operating conditions. The requirement is to generate a family of patterns for each class of robot behavior that needs to be recognized. Therefore, both Segway RMP and Pioneer 2AT robots were made to execute several cycles of each of the three different types of motion trajectories on the pressure sensitive floor of the laboratory environment. Each member of a family represents the pattern of a single experiment of one robot executing a particular motion profile. As a robot changes its type of motion from one (e.g., circular) to another (e.g., random), the pattern classification algorithm is capable of detecting this change after a (statistically quasi-stationary) steady state is reached. During the brief transient period, the analysis of pattern classification may not yield accurate results because the resulting time series may not be long enough to extract the features correctly.

Fig. 12(a) shows an example of the sensor reading when the robot moves over it. The voltage generated by the piezoelectric pressure sensor gradually increases as the robot approaches the sensor, and discharge occurs in the sensor when the robot moves away from the sensor and hence the voltage resumes to be 0. The choice of mother wavelet depends on the shape of the sensor signal; the mother wavelet should match the shape of the sensor signal in order to capture the signature of the signal. Haar wavelet (*db*1), as shown in Fig. 12(b), is chosen to be the mother wavelet in this application. The sensor data collected by the 9×16 grid is stacked sequentially to generate a one-dimensional time series. For each motion trajectory consisting of several cycles, the time series data collected from the pressure sensors was divided into 40 to 50 data sets. The length of each data set is 3.0×10^5 data points, which corresponds to about three minutes of the experiment time. The data sets are randomly divided into half training and half testing. Among the training data, 10 sets are chosen to serve as the partitioning data sets as well.

6.2. *Pattern analysis for robot and motion classification*

This subsection provides a description of the application of different pattern analysis methods to time series data of pressure sensors for classification of the robots and their motion types.

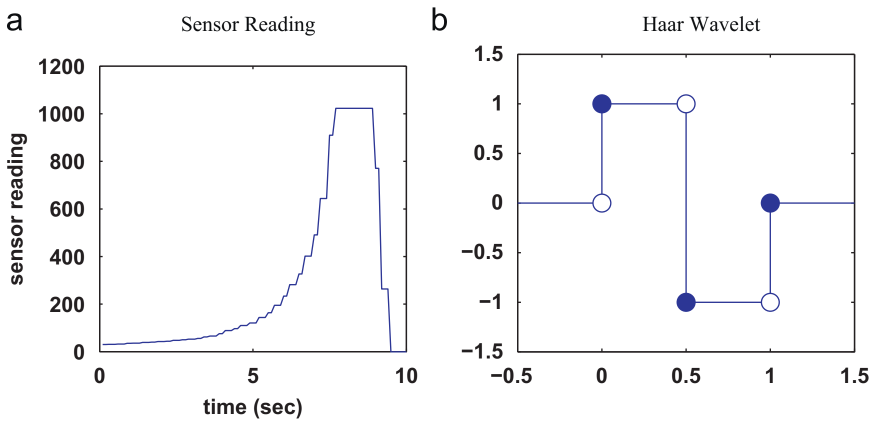

Fig. 12. Example of sensor readings and plot of Haar wavelet: (a) Sensor reading and (b) Haar wavelet.

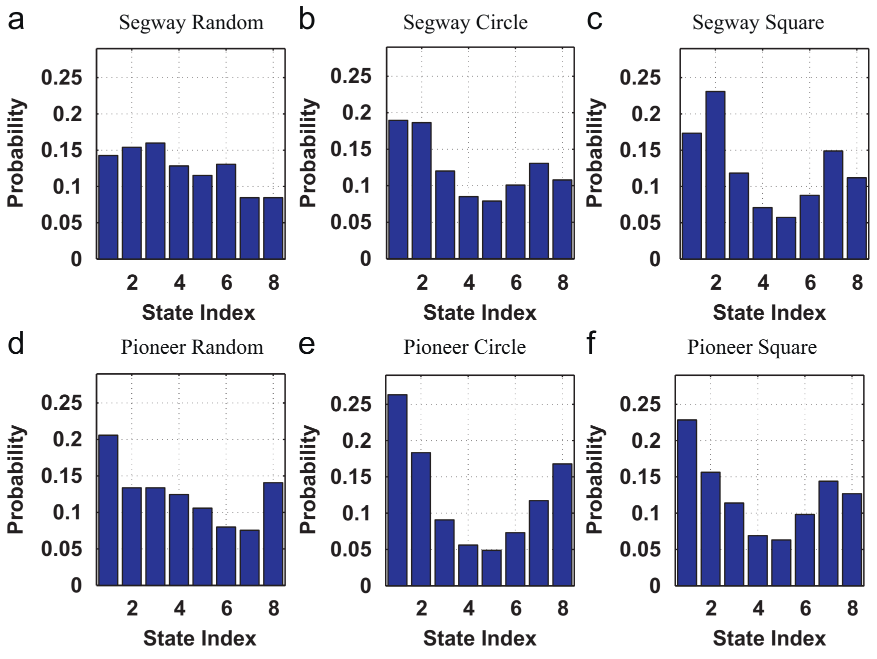

Fig. 13. Ensemble mean of the state probability vectors (feature vectors) for pattern classification: (a) Segway random, (b) Segway circle, (c) Segway square, (d) Pioneer random, (e) Pioneer circle, and (f) Pioneer square.

For feature extraction using SDF, each data set of a family (or class) is analyzed to generate the corresponding state probability vectors (i.e., patterns). Thus, the patterns $\mathbf{p}_j^{\mathcal{C}_i}$, $j=1,2,\ldots,n_i$, are generated for n_i samples in each class $\mathcal{C}_i$ corresponding to robot type and motion. Following the SDF procedure, each time-series data set is analyzed using $|\Sigma|=8, \ell=2$ and $m=1$. Ensemble mean of pattern vectors for different motion profiles of Segway and Pioneer robots is shown in Fig. 13. It can be observed in Fig. 13 that the state probability vectors of Segway and Pioneer robots are quite distinct. Following Fig. 5, for each motion type, the state probability vectors $\mathbf{p}_j^{\mathcal{C}_i}$ were equally divided into training sets and testing sets.

In this application, the efficacy of SDF for feature extraction is evaluated by comparison with PCA. The time series data are transformed to the frequency domain for noise mitigation and then the standard PCA method is implemented to identify the eigen-directions of the transformed data and to obtain an orthogonal linear operator that projects the frequency-domain features onto a low-dimensional compressed-feature space. For the purpose of comparison, the dimension of this compressed feature space is chosen to be the same as that of the feature vectors obtained by SDF. In this application, the support vector machine (SVM), k-NN algorithm, radial basis Neural Network (rbfNN), and multilayer perceptron Neural Network (mlpNN) have been used as the pattern classifiers to identify different classes of feature vectors extracted by SDF and PCA. The pattern classifiers identify the type of the robot and its motion profile, based on the acquired statistical patterns. Since, in this pattern classification problem, there are two robots and each robot has three different types of motion profiles, it is natural to formulate this problem as a two-layer classification problem, where the robot type is identified in the first layer followed by identification of the motion type in the second layer. Thus, the above problem is formulated using a tree-structure classification as shown in Fig. 14.

6.3. Experimental results and discussion

The performance comparison between SDF and PCA that are used in conjunction with different classifiers is presented in Table 3. The left part of Table 3 shows the results of robot type and robot motion classification using SDF for feature extraction, and the right part shows the corresponding results using PCA for feature extraction. As stated earlier, SVM, k-NN, rbfNN, and mlpNN have been used as pattern classifiers in both cases. The *polynomial* kernel is used in SVM [33], and a neighbor size of $k=5$ is used in the k-NN classifier. The rbfNN uses one hidden layer and one output layer with a single neuron. Optimal training is obtained with 100 neurons in the hidden layer that uses a radial basis function, while the output layer uses a linear transfer function. The mlpNN utilizes a feed-forward back-propagation network that consists of one hidden layer with 50 neurons and an output layer with a single neuron; the tangent sigmoid function has been used in the hidden layers as a transfer function, while the output layer uses a linear function.

It is noted that since the tree structure is used for pattern classification, the motion recognition results are affected by the robot recognition results. For example, the samples that are incorrectly classified in the robot recognition stage will be incorrectly classified for motion recognition also. However, this particular aspect is application dependent and the tree structure for classification can be redesigned accordingly. The classification results are presented in Table 3 that show the accuracy percentage equal to (# correct classifications/# total data sets $\times$ 100). In the left part of Table 3, the combination of SDF with all four classifiers yield good accuracy in recognizing the robot type, namely, 100% for Segway and more than 94% for Pioneer. Although not explicitly shown in Table 3, but all four classifiers successfully identified the three types of motions of the Pioneer robot with 100% accuracy in the robot motion classification stage. The errors in the motion recognition, as seen in Table 3, originate from the robot recognition stage. The success rate in recognizing Segway motion is slightly lower due to the following possible reasons: (i) complicated kinematics of the Segway robot, and (ii) the non-stationarity in the samples due to uncertainties in the laboratory environment (e.g., floor friction). It is expected that this accuracy would further improve if the number of stationary samples is increased. The right part of Table 3 shows that PCA yields slightly worse (but still comparable) results than SDF in robot recognition. However, SDF significantly outperforms PCA in motion recognition, because the feature vectors extracted by PCA lack class separability among different types of motions, which in turn yields poor motion recognition accuracy.

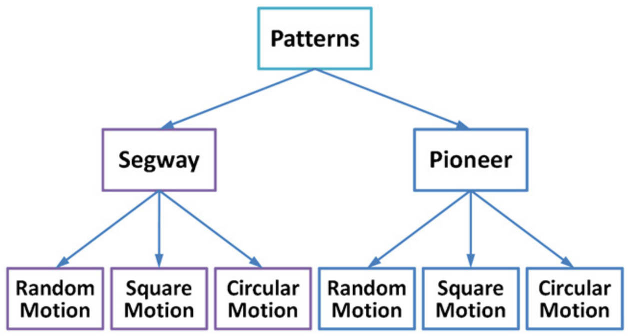

Fig. 14. Tree structure for pattern classification.

Fig. 15 exhibits the effects of changing the number of neighbors in the k-NN classifier on the performance (i.e., error magnitude) and robustness (i.e., error fluctuation) by comparing the SDF-based and PCA-based features. Fig. 15 shows that SDF-based classification is consistently superior to PCA-based classification in terms of both performance and robustness. For $k \geq 3$, the error in robot identification using SDF-based features has a decreasing trend and

Table 3
Results of robot and motion classification.

Feature extraction using SDF						**Feature extraction using PCA**					
Pattern	Robot recognition		Motion recognition			Pattern	Robot recognition		Motion recognition		
Classifier	Robot	Result	Motion	Result	Total	Classifier	Robot	Result	Motion	Result	Total
SVM	Segway	**100%**	Random	92% ($\frac{23}{25}$)	**95%**	SVM	Segway	**91%**	Random	84% ($\frac{21}{25}$)	**81%**
		($\frac{58}{58}$)	Circular	92% ($\frac{12}{13}$)	($\frac{55}{58}$)			($\frac{53}{58}$)	Circular	77% ($\frac{10}{13}$)	($\frac{47}{58}$)
			Square	100% ($\frac{20}{20}$)					Square	80% ($\frac{16}{20}$)	
	Pioneer	**94%**	Random	100% ($\frac{20}{20}$)	**94%**		Pioneer	**100%**	Random	20% ($\frac{4}{20}$)	**65%**
		($\frac{61}{65}$)	Circular	90% ($\frac{18}{20}$)	($\frac{61}{65}$)			($\frac{65}{65}$)	Circular	65% ($\frac{13}{20}$)	($\frac{42}{65}$)
			Square	92% ($\frac{23}{25}$)					Square	100% ($\frac{25}{25}$)	
k-NN	Segway	**100%**	Random	88% ($\frac{22}{25}$)	**91%**	k-NN	Segway	**100%**	Random	84% ($\frac{21}{25}$)	**52%**
		($\frac{58}{58}$)	Circular	85% ($\frac{11}{13}$)	($\frac{53}{58}$)			($\frac{58}{58}$)	Circular	69% ($\frac{9}{13}$)	($\frac{30}{58}$)
			Square	100% ($\frac{20}{20}$)					Square	0% ($\frac{0}{20}$)	
	Pioneer	**94%**	Random	95% ($\frac{19}{20}$)	**94%**		Pioneer	**88%**	Random	0% ($\frac{0}{20}$)	**51%**
		($\frac{61}{65}$)	Circular	100% ($\frac{20}{20}$)	($\frac{61}{65}$)			($\frac{57}{65}$)	Circular	55% ($\frac{11}{20}$)	($\frac{33}{65}$)
			Square	88% ($\frac{22}{25}$)					Square	88% ($\frac{22}{25}$)	
rbfNN	Segway	**100%**	Random	84% ($\frac{21}{25}$)	**91%**	rbfNN	Segway	**100%**	Random	92% ($\frac{23}{25}$)	**72%**
		($\frac{58}{58}$)	Circular	92% ($\frac{12}{13}$)	($\frac{53}{58}$)			($\frac{58}{58}$)	Circular	31% ($\frac{4}{13}$)	($\frac{42}{58}$)
			Square	100% ($\frac{20}{20}$)					Square	75% ($\frac{15}{20}$)	
	Pioneer	**97%**	Random	95% ($\frac{19}{20}$)	**97%**		Pioneer	**95%**	Random	0% ($\frac{0}{20}$)	**66%**
		($\frac{63}{65}$)	Circular	100% ($\frac{20}{20}$)	($\frac{63}{65}$)			($\frac{62}{65}$)	Circular	100% ($\frac{20}{20}$)	($\frac{43}{65}$)
			Square	96% ($\frac{24}{25}$)					Square	92% ($\frac{23}{25}$)	
mlpNN	Segway	**100%**	Random	96% ($\frac{24}{25}$)	**98%**	mlpNN	Segway	**100%**	Random	96% ($\frac{24}{25}$)	**98%**
		($\frac{58}{58}$)	Circular	100% ($\frac{13}{13}$)	($\frac{57}{58}$)			($\frac{58}{58}$)	Circular	100% ($\frac{13}{13}$)	($\frac{57}{58}$)
			Square	100% ($\frac{20}{20}$)					Square	100% ($\frac{20}{20}$)	
	Pioneer	**100%**	Random	100% ($\frac{20}{20}$)	**100%**		Pioneer	**100%**	Random	85% ($\frac{17}{20}$)	**92%**
		($\frac{65}{65}$)	Circular	100% ($\frac{20}{20}$)	($\frac{65}{65}$)			($\frac{65}{65}$)	Circular	95% ($\frac{19}{20}$)	($\frac{60}{65}$)
			Square	100% ($\frac{25}{25}$)					Square	96% ($\frac{24}{25}$)	

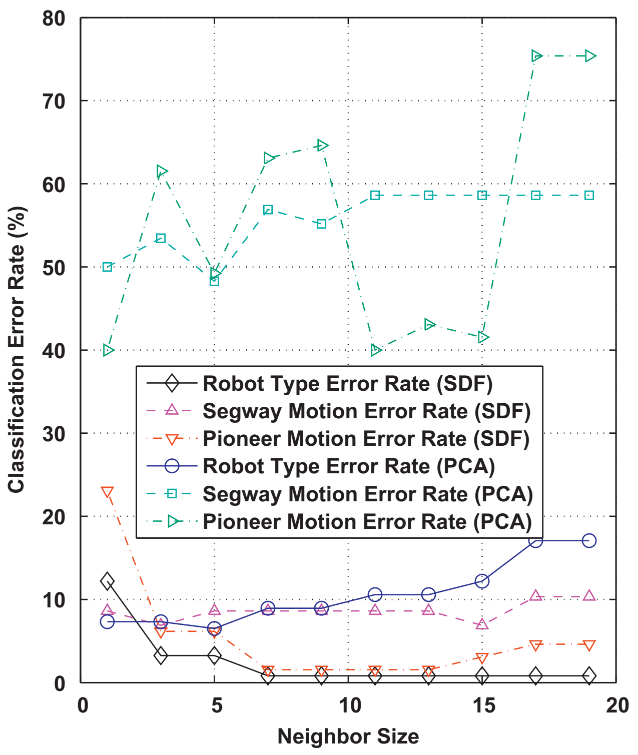

Fig. 15. Classification error vs. neighbor size in k-NN classifiers.

Table 4
Comparison of computational complexity of feature extraction methods.

Feature extraction method	Training stage		Testing stage	
	Execution time (s)	Memory requirement (MB)	Execution time (s)	Memory requirement (MB)
SDF	5.21	65.2	5.17	64.9
PCA	6.62	233.85	0.04	37.5

negligible fluctuations as the neighbor size increases. The corresponding errors of motion classification for both Segway and Pioneer are nearly consistent without any noticeable fluctuations. In contrast, the error in robot identification using PCA-based features is relatively large and has an increasing trend as the neighbor size increases. The corresponding error of motion classification for Pioneer suffers from large fluctuations, while these fluctuations for Segway are much smaller although the error magnitude is very large. These results show that the SDF-based features yield a significantly better separability among different classes as compared to PCA-based features.

The proposed SDF-based method has a computational complexity of $O(N)$ for a given algebraic structure of the PFSA, with a leading constant that is proportional to the number of scales used in the wavelet transform [34]. A comparison of the computational complexity of SDF and PCA is presented in Table 4 in terms of execution time and memory requirements for processing each data set. For the data set consisting of 3.0×10^5 data points, which is about 3.5 min of the experimentation time, it takes an average of 5.21 s for SDF and 6.62 s for PCA to process each data set in the training stage, respectively. The memory requirement is 65.2 MB for SDF, and 233.9 MB for PCA. PCA takes longer execution time and consumes more memory in the training stage because it needs to calculate the covariance matrix using all training data sets. In the testing stage, the execution time and memory requirement for SDF are almost the same as those in the training stage, while the PCA requires less time and memory than those in the training stage. Both feature extraction methods have real-time implementation capability since the execution time in the testing stage is much less than the experiment time spent for collecting each data set. The rationale for SDF taking longer time than PCA in the testing stage is that the SDF-based method involves wavelet transformation and PFSA construction from the two-dimensional wavelet image in both training and testing stages, while the PCA-based method only involves Fourier transform and finding the projection of the testing data set using the projection matrix that is already constructed in the training stage; this is a price paid for the superior performance and robustness achieved in SDF-based feature extraction (see Fig. 15 and Table 3). It is anticipated that the PCA-based method will be relatively slower if the raw time-series is (more effectively) de-noised by wavelet transform instead of Fourier transform. In these experiments, the data analysis was performed on a 2.83 GHz Quad Core CPU desktop computer with 8.0 GB of RAM.

7. Summary, conclusions and future work

This paper presents a feature extraction method for pattern classification in complex dynamical systems. These features are extracted as statistical patterns using symbolic modeling of the wavelet images, generated from sensor time series. An appropriate selection of the wavelet basis function and the scale range allows the wavelet-transformed signal to be de-noised relative to the original (possibly) noise-contaminated signal before the resulting wavelet image is partitioned for symbol generation. In this way, the symbolic images generated from wavelet coefficients capture the signal characteristics with larger fidelity than those obtained directly from the original signal. These symbolic images are then modeled using probabilistic finite state automata (PFSA) that, in turn, generate the low-dimensional statistical patterns, also called feature vectors. This process is referred to as symbolic dynamic filtering (SDF).

The proposed SDF-based feature extraction and pattern classification methodology is executable in real time on commercially available computational platforms. A distinct advantage of this method is that the low-dimensional feature vectors, generated from sensor time series in real time, can be communicated as short packets over a limited-bandwidth wireless sensor network with limited-memory nodes.

The feature extraction and pattern classification methodology has been experimentally validated on laboratory apparatuses for two different applications: (i) early detection of evolving damage in polycrystalline alloys and (ii) behavior recognition in mobile robots by identification of their type and motion profiles. Experimental results in these two specific applications show that the proposed SDF-based methodology has superior performance relative to some of the common pattern recognition tools (e.g., principal component analysis).

Further theoretical and experimental research is recommended in the following areas:

1. Exploration of other wavelet transform techniques for wavelet image generation, such as fast wavelet transform and wavelet packet analysis.
2. Optimization of the partitioning scheme for symbolization of the wavelet images.
3. Experimental validation in other applications.

References

[1] S. Gupta, A. Ray, Statistical mechanics of complex systems for pattern identification, Journal of Statistical Physics 134 (2) (2009) 337–364.
[2] T.W. Liao, Clustering of time series data—a survey, Pattern Recognition 38 (2005) 1857–1874.
[3] K. Fukunaga, Statistical Pattern Recognition, second ed., Academic Press, Boston, USA, 1990.
[4] R.O. Duda, P.E. Hart, D.G. Stork, Pattern Classification, second ed., Wiley Interscience, New York, NY, 2001.
[5] T. Lee, Independent Component Analysis: Theory and Applications, Kluwer Academic Publishers, Boston, USA, 1998.
[6] R. Rosipal, M. Girolami, L. Trejo, Kernel PCA feature extraction of event-related potentials for human signal detection performance, in: Proceedings of the International Conference on Artificial Neural Networks Medicine Biology, 2000, pp. 321–326.
[7] D.J. Berndt, J. Clifford, Using dynamic time warping to find patterns in time series, in: Proceedings of the AAAI Workshop on Knowledge Discovery in Databases, 1994, pp. 359–370.
[8] F. Gullo, G. Ponti, A. Tagarelli, S. Greco, A time series representation model for accurate and fast similarity detection, Pattern Recognition 42 (7) (2009) 2998–3014.
[9] O.R. Lautour, P. Omenzetter, Damage classification and estimation in experimental structures using time series analysis and pattern recognition, Mechanical Systems and Signal Processing 24 (2010) 1556–1569.
[10] W. Zucchini, I.L. MacDonald, Hidden Markov Models for Time Series: An Introduction Using R, CRC Press, 2009.
[11] K.P. Zhu, Y.S. Wong, G.S. Hong, Wavelet analysis of sensor signals for tool condition monitoring: a review and some new results, International Journal of Machine Tools and Manufacture 49 (2009) 537–553.
[12] D.B. Percival, A.T. Walden, Wavelet Methods for Time Series Analysis, Cambridge University Press, 2000.
[13] S. Pittner, S.V. Kamarthi, Feature extraction from wavelet coefficient for pattern recognition tasks, IEEE Transactions on Pattern Analysis and Machine Intelligence 21 (1) (1999) 83–88.
[14] C.M. Bishop, Pattern Recognition and Machine Learning, Springer, 2006.
[15] D. Lind, M. Marcus, An Introduction to Symbolic Dynamics and Coding, Cambridge University Press, United Kingdom, 1995.
[16] C.S. Daw, C.E.A. Finney, E.R. Tracy, A review of symbolic analysis of experimental data, Review of Scientific Instruments 74 (2) (2003) 915–930.
[17] K. Keller, H. Lauffer, Symbolic analysis of high-dimensional time series, International Journal of Bifurcation Chaos 13 (9) (2003) 2657–2668.
[18] A. Ray, Symbolic dynamic analysis of complex systems for anomaly detection, Signal Processing 84 (7) (2004) 1115–1130.
[19] S. Gupta, A. Ray, Symbolic dynamic filtering for data-driven pattern recognition, in: E.A. Zoeller (Ed.), Pattern Recognition: Theory and Application, Nova Science Publishers, Hauppage, NY, 2007, pp. 17–71 (Chapter 2).
[20] C. Rao, A. Ray, S. Sarkar, M. Yasar, Review and comparative evaluation of symbolic dynamic filtering for detection of anomaly patterns, Signal, Image, Video Processing 3 (2009) 101–114.
[21] M. Buhl, M. Kennel, Statistically relaxing to generating partitions for observed time-series data, Physical Review E 71 (4) (2005) 046213.
[22] V. Rajagopalan, A. Ray, Symbolic time series analysis via wavelet-based partitioning, Signal Processing 86 (11) (2006) 3309–3320.
[23] A. Subbu, A. Ray, Space partitioning via Hilbert transform for symbolic time series analysis, Applied Physics Letters 92 (8) (2008) 084107-1–084107-3.
[24] S.G. Mallat, A Wavelet Tour of Signal Processing: The Sparse Way, third ed., Academic Press, 2009.
[25] C.J. Veenman, M.J.T. Reinders, E.M. Bolt, E. Baker, A maximum variance cluster algorithm, IEEE Transactions on Pattern Analysis and Machine Intelligence 24 (9) (2002) 1273–1280.
[26] T. Chau, A.K. Wong, Pattern discovery by residual analysis and recursive partitioning, IEEE Transactions on Knowledge and Data Engineering 11 (6) (1999) 833–852.
[27] Y. Kakizawa, R. Shumway, N. Taniguchi, Discrimination and clustering for multivariate time series, J. Amer. Stat. Assoc. 93 (441) (1999) 328–340.
[28] P. Abry, Ondelettes et turbulence, multiŕesolutions, algorithmes de décomposition, invariance déchelles, Diderot Editeur, Paris, 2000.
[29] T. Cover, P. Hart, Nearest neighbor pattern classification, IEEE Transactions on Information Theory 13 (1) (1967) 21–27.
[30] S.S. Haykin, Neural Networks and Learning Machines, third ed., Prentice-Hall, New York, 2009.
[31] S. Gupta, A. Ray, E. Keller, Symbolic time series analysis of ultrasonic data for early detection of fatigue damage, Mechanical Systems and Signal Processing 21 (2007) 866–884.
[32] B. Gerkey, R. Vaughan, A. Howard, The Player/Stage project: tools for multi-robot and distributed sensor systems, in: Proceedings of the International Conference on Advanced Robotics, Coimbra, Portugal, June 30–July 3, 2003, pp. 317–323.
[33] S. Canu, Y. Grandvalet, V. Guigue, A. Rakotomamonjy, SVM and kernel methods Matlab toolbox, Perception Systèmes et Information, INSA de Rouen, Rouen, France, 2005.
[34] A. Muñoz, R. Ertlé, M. Unser, Continuous wavelet transform with arbitrary scales and o(n) complexity, Signal Processing 82 (5) (2002) 749–757. doi:10.1016/S0165-1684(02)00140-8.

Xin Jin received the Bachelor's degree in Mechanical Engineering from Shanghai Jiao Tong University (SJTU), Shanghai, China in July 2007. Subsequently, he received the M.S. degree in Electrical Engineering in December 2009 and the M.S. degree in Mechanical Engineering in August 2010, both from the Pennsylvania State University, where he is currently a Ph.D. candidate in Mechanical Engineering. His research interests include: instrumentation and control, machine learning, and robotics. He has been a student member of the Institute of Electrical and Electronics Engineers (IEEE), the American Society of Mechanical Engineers (ASME), and the American Nuclear Society (ANS) since 2009.

Shalabh Gupta obtained his Ph.D. in Mechanical Engineering from the Pennsylvania State University, at University Park, PA, in 2006. He also obtained Master of Science degrees in Mechanical Engineering and Electrical Engineering from the Pennsylvania State University. He is currently a research associate in Department of Mechanical Engineering at the Pennsylvania State University. Dr. Gupta's research interests include the Science of Autonomy, Swarm Robotics, Intelligent Systems, Machine Learning, Network Science, and Fault Detection & Isolation in Complex Systems. His research efforts have been instrumental in opening new fields of data understanding and pattern discovery via interfacing multidisciplinary concepts derived from Statistical Mechanics, Symbolic Dynamics, and Languages & Automata Theory. He is a member of the American Society of Mechanical Engineers (ASME) and the Institute of Electrical and Electronics Engineers (IEEE).

Kushal Mukherjee received his Bachelor's degree in mechanical engineering from the Indian Institute of Technology (IIT), Roorkee, India in 2006. Subsequently, he obtained an M.S. in Mechanical Engineering and an M.S. in Electrical Engineering from the Pennsylvania State University in 2009, where, he is currently pursuing a Ph.D. in Mechanical Engineering. His areas of interest include symbolic dynamics, sensor networks, and multi-agent systems.

Asok Ray received the Ph.D. degree in Mechanical Engineering from Northeastern University, Boston, MA, and also graduate degrees in each discipline of Electrical Engineering, Mathematics, and Computer Science. Dr. Ray joined the Pennsylvania State University in July 1985, and is currently a Distinguished Professor of Mechanical Engineering. Prior to joining Penn State, Dr. Ray also held research and academic positions at Massachusetts Institute of Technology and Carnegie-Mellon University as well as research and management positions at GTE Strategic Systems Division, Charles Stark Draper Laboratory, and MITRE Corporation. Dr. Ray had been a Senior Research Fellow at NASA Glenn Research Center under a National Academy of Sciences award. Dr. Ray's research experience and interests include: Control and optimization of continuously varying and discrete-event dynamical systems; Intelligent instrumentation for real-time distributed systems; and Modeling and analysis of complex dynamical systems from thermodynamic perspectives in both deterministic and stochastic settings, as applied to Robotics, Undersea autonomous vehicles, Aeronautics and astronautics, and Fossil-fueled and nuclear power plants. Dr. Ray has authored or co-authored over four hundred and fifty research publications including about two hundred and fifty scholarly articles in refereed journals such as transactions of IEEE, ASME, and AIAA, and research monographs. Dr. Ray is a Fellow of IEEE, a Fellow of ASME, and a Fellow of World Innovative Foundation (WIF). Further details of Dr. Ray's credentials are available in the web address http://www.mne.psu.edu/Ray/.