Symbolic Dynamic Filtering and Language Measure for Behavior Identification of Mobile Robots

Goutham Mallapragada, Asok Ray, *Fellow, IEEE*, and Xin Jin, *Student Member, IEEE*

***Abstract*—This paper presents a procedure for behavior identification of mobile robots, which requires limited or no domain knowledge of the underlying process. While the features of robot behavior are extracted by symbolic dynamic filtering of the observed time series, the behavior patterns are classified based on language measure theory. The behavior identification procedure has been experimentally validated on a networked robotic test bed by comparison with commonly used tools, namely, principal component analysis for feature extraction and Bayesian risk analysis for pattern classification.**

***Index Terms*—Feature extraction, language measure, pattern classification, robotic signatures, symbolic dynamic filtering (SDF).**

I. Introduction

CLASSIFICATION of patterns in the behavior of dynamical systems is critical for multiagent coordination [1]. For example, a robotic platform is often required to make real-time decisions for target tracking [2]. Such decisions could be made based on the ensemble of information acquired from both mobile and stationary sensors on a distributed network. In these applications, it is necessary to rely on sensor time series because accurate and computationally tractable modeling of robot dynamics in changing environments may not be feasible solely based on the fundamental principles of physics.

Many techniques, which are largely application specific, have been reported in literature for feature extraction [3]–[5] and pattern classification [6], [7]. An example is dynamic Bayesian networks [8] that have been used to identify and track targets primarily in military applications. Here, the state machines are hand-coded, and the probabilities are estimated with expectation–maximization algorithms. Another example is conventional hidden Markov modeling (HMM) tools that have been adopted in the context of robotic soccer [9], where the states and the structure of HMM are fixed *a priori*, and the state transition and observation probabilities are identified using the Baum–Welch algorithm [10]. Chang and Huang [11] constructed an HMM to analyze human motions by making use of the domain knowledge to extract feature vectors. In contrast, this paper presents feature extraction and pattern classification for identification of mobile robot type and motion based on time-series analysis with limited or no requirements of domain knowledge about the robot kinematics. The proposed pattern classification method has general applicability and is robust relative to spurious noise and environmental uncertainties.

Three types of motion (e.g., circular, square, and random) for two different types of mobile robots (e.g., Pioneer 2AT and Segway RMP) have been investigated in this paper; they emulate both structured and unstructured movements of inanimate objects (e.g., different types of vehicles) and animate objects (e.g., human and animals) for pattern analysis from time-series data of piezoelectric sensors, installed underneath the laboratory floor, which are representatives of seismic signals. This research topic is inspired from various real-life applications of pattern classification. Examples are the following: 1) identification of enemy vehicles and their operations in air, sea, and ground battlefields [12], and 2) detection and classification of human and animal movements across borders [13].

A dynamic data-driven method, called symbolic dynamic filtering (SDF) [14], has been adopted in this paper for extraction of robot behavior features. Relying on the probabilistic finite-state automata (PFSAs) that are constructed from the observed time series, a coherent procedure is developed to extract features of robot behavior from the observed time series and to classify behavior patterns based on the language measure theory (LMT) [15] from the extracted features. While the theory of SDF has been reported in the field of anomaly detection with diverse applications [16], there has been very little work on SDF-based feature extraction and pattern classification among multiple classes. Similarly, while LMT has been applied to optimal decision and control of discrete-event systems [17], this is the first time that it has been used in the context of a pattern classification problem. The SDF-based feature extraction, presented in this paper, is different from that reported in [18] where the reference patterns are similar for all robot classes, which renders them unsuitable for LMT pattern classification. This problem is circumvented in the feature extraction method, developed in this paper, by generating different reference patterns for individual robot classes, based on a common data partitioning. From these perspectives,

Manuscript received June 19, 2010; revised May 19, 2011; accepted October 6, 2011. Date of publication November 3, 2011; date of current version May 16, 2012. This work was supported in part by the U.S. Army Research Laboratory and the U.S. Army Research Office under Grant W911NF-07-1-0376 and in part by the U.S. Office of Naval Research under Grant N00014-09-1-0688. This paper was recommended by Associate Editor J. Shamma.

G. Mallapragada was with the Department of Mechanical Engineering, The Pennsylvania State University, University Park, PA 16802 USA. He is now with Universal Robotics, Inc., Nashville, TN 37217 USA (e-mail: mr.goutham@gmail.com).

A. Ray and X. Jin are with the Department of Mechanical Engineering, The Pennsylvania State University, University Park, PA 16802 USA (e-mail: axr2@psu.edu; http://www.mne.psu.edu/Ray/; xuj103@psu.edu; http://www.personal.psu.edu/xuj103/).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TSMCB.2011.2172419

the major contributions of this paper are summarized as follows.

1) Development of a data-driven method for behavior identification of mobile robots in the following two steps, which requires limited or no domain knowledge:
   a) robust feature extraction from noise-contaminated sensor time series based on SDF [14];
   b) real-time pattern classification based on LMT [15].
2) Experimental validation on a networked robotic test bed by comparison with commonly used tools of feature extraction and pattern classification.

This paper is organized in six sections (including the present section) and an Appendix that reviews the theory of SDF for construction of feature vectors. Section II lays down a framework of feature extraction, and Section III presents pattern classification based on the extracted features. Section IV presents the algorithms to support both Sections II and III. Section V describes both hardware and software of the experimental apparatus and discusses the experimental results. This paper is summarized and concluded in Section VI along with recommendations for future research.

II. Feature Extraction from Time Series

This section presents a framework of feature extraction from time series for different classes of robot motion. Let $\mathcal{R} = \{\rho_1, \ldots, \rho_{|\mathcal{R}|}\}$ be the set of robots, and let each robot execute one or more of the different motion profiles in the set $\Phi = \{\varphi_1, \ldots, \varphi_{|\Phi|}\}$. Let the number of profiles executed by the robot ρ_i be k_i, and let the indices of the profiles executed by ρ_i be denoted as $\{\theta_1^i, \ldots, \theta_{k_i}^i\}$, i.e., ρ_i executes profiles $\{\varphi_{\theta_1^i}, \ldots, \varphi_{\theta_{k_i}^i}\}$. Thus, the total number of pattern classes is $|\Xi| = \sum_i k_i \leq |\mathcal{R}||\Phi|$. For each class, robots' behavioral patterns may vary due to, for example, changes in the payload, type of drive system, and faults.

A. SDF for Feature Extraction

This section presents SDF-based feature extraction that generates a unique reference pattern for each robot class from the respective time series. Details of the SDF theory have been reported in previous publications [14], [16], [18]. However, for completeness of this paper, the essential ingredients of SDF are presented in the Appendix, and the algorithms for SDF-based feature extraction are listed in Section IV-A. The features of robot behavior are extracted as state probability vectors of the PFSAs in the setting of SDF [14]. The performance of SDF-based feature extraction is compared with that of the principal component analysis (PCA) [6], [7].

Let $\Xi = \{\xi_i, i = 1, 2, \ldots, |\Xi|\}$ be defined as a (nonempty finite) set of pattern classes, where $|\Xi|$ is the cardinality of Ξ such that $2 \leq |\Xi| \leq n$ and n is the number of PFSA states. The training data sets for all pattern classes are combined as a single time series to construct a common partition $\wp$ for symbol generation. In the training stage, PFSAs G_i, $i = 1, 2, \ldots, |\Xi|$, are constructed from the respective symbol sequences based on the partition $\wp$. The state probability vector $\bar{\boldsymbol{p}}_i$ of each PFSA G_i is chosen as a pattern. In the sequel, $\bar{\boldsymbol{p}}_i$, $i = 1, 2, \ldots, |\Xi|$, are called reference patterns.

Let $\boldsymbol{x} \in \mathbb{R}^{1\times N}$ be a single time series of robot motion data in the testing stage, where N is the length of the time series. The goal here is to obtain a feature vector $\boldsymbol{p}(\boldsymbol{x}) \in \mathbb{R}^{1\times m}$ that is a low-dimensional representation (i.e., $m \ll N$) of the data set $\boldsymbol{x}$.

The salient steps for construction of the patterns (i.e., state probability vectors of PFSA) for SDF-based feature extraction are delineated in the following:

1) identification of a set of pattern classes spanning the robot behavior under different operational scenarios;
2) generation of sets of time series for all pattern classes;
3) generation of a reference partition that is subsequently used to obtain the unique probability vectors representing the patterns.

B. PCA for Feature Extraction

Let the training data sets (i.e., time series) of a given robot class be organized as an $(M \times N)$-dimensional data matrix, where M is the number of data sets and N is the length of each (1-D) time series as stated earlier. Let $\mathbf{X}$ be the centered version of the original $(M \times N)$ data matrix, where each row of $\mathbf{X}$ is an individual data sample $\boldsymbol{x}$ with zero mean. For $M < N$ (which is the situation in the robot experiments conducted in this paper), it is numerically efficient [7] to analyze the $(M \times M)$ matrix $(1/M)\mathbf{X}\mathbf{X}^{\mathrm{T}}$ that has the same nonzero eigenvalues as the $(N \times N)$ computed covariance matrix $(1/M)\mathbf{X}^{\mathrm{T}}\mathbf{X}$.

Let $\{\mathbf{v}_i\}$ be the (normalized) eigenvectors of the real symmetric matrix $(1/M)\mathbf{X}\mathbf{X}^{\mathrm{T}}$ corresponding to the (real positive) eigenvalues $\{\lambda_i\}$ that are arranged in the decreasing order of magnitude, i.e., $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_M$. The smallest integer m is selected such that $\sum_{i=1}^{m} \lambda_i > \eta \sum_{i=1}^{M} \lambda_i$, where $1 \leq m \leq M$ and the threshold parameter η is a real positive fraction close to one. The corresponding (normalized) eigenvectors $\mathbf{u}_i \in \mathbb{R}^N$ in the original data space are obtained as follows [7]:

$$\mathbf{u}_i = \frac{1}{\sqrt{(M\lambda_i)}}\mathbf{X}^{\mathrm{T}}\mathbf{v}_i, \qquad i = 1, 2, \ldots, m. \tag{1}$$

The m eigenvectors obtained from (1) are grouped to form the $(N \times m)$ projection matrix

$$\mathbf{W}_{\mathrm{PCA}} \triangleq [\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_m]. \tag{2}$$

An $(M \times m)$ data matrix $\mathbf{Y}_{\mathrm{train}}$ is generated from the training data set $\mathbf{X}_{\mathrm{train}}$ that has a similar structure as $\mathbf{X} \in \mathbb{R}^{M\times N}$

$$\mathbf{Y}_{\mathrm{train}} \triangleq \mathbf{X}_{\mathrm{train}}\mathbf{W}_{\mathrm{PCA}}. \tag{3}$$

Each of the m columns of the matrix $\mathbf{Y}_{\mathrm{train}}$ is considered as a feature vector. Subsequently, for pattern classification, a $(1 \times m)$ reference pattern is defined for each class as

$$\bar{\boldsymbol{p}} \triangleq \frac{1}{M}\mathbf{1}\mathbf{Y}_{\mathrm{train}}, \qquad \text{with } \mathbf{1} \triangleq [1\ 1\ \ \cdots\ 1] \in \mathbb{R}^{1\times M} \tag{4}$$

where (2) and (3), respectively, yield the projection matrix $\mathbf{W}_{\mathrm{PCA}}$ and the low-dimensional data matrix $\mathbf{Y}_{\mathrm{train}}$ from the

training data matrix $\mathbf{X}_{\text{train}}$. Similarly, in the testing stage, a time series $\boldsymbol{x}_{\text{test}} \in \mathbb{R}^{1\times N}$ yields the $(1 \times m)$ feature vector as

$$\boldsymbol{p} \triangleq \boldsymbol{x}_{\text{test}} \mathbf{W}_{\text{PCA}}. \tag{5}$$

The algorithms for PCA-based feature extraction are described in Section IV-A.

III. Pattern Classification in Mobile Robots

This section presents two methods of online pattern classification based on the extracted features $\boldsymbol{p}(\boldsymbol{x})$ that are obtained from the observed time-series data $\boldsymbol{x}$ by SDF and PCA. The first method of pattern classification is constructed based on the measure of formal languages [15] and, thus, is called LMT classification. Its performance is compared with that of Bayesian risk analysis (BRA), which is a commonly used tool for pattern classification [6]. Both pattern classification methods make use of the extracted features $\boldsymbol{p}(\boldsymbol{x})$ that are computed online by either SDF or PCA from the observed time series $\boldsymbol{x}$ (see algorithms in Section IV-B).

A. LMT for Pattern Classification

The underlying concept of the LMT pattern classification is built upon signed real measures of formal languages [15] as an alternative to existing pattern classification tools [6], [7]. In general, LMT classification is computationally inexpensive because it is a deterministic method based on orthogonal projection in a Hilbert space, and LMT does not require identification of *a priori* and *a posteriori* probability distributions.

It is noted that LMT classification is a natural extension of SDF-based feature extraction because the underlying measure is directly applied to the language of PFSAs that are generated from the symbol sequences obtained by partitioning the time series. In essence, the PFSA constructed from a symbol sequence acts as a language generator, and a signed real measure of the language is obtained by assigning a weight to each of the states of the constructed PFSAs [15]. In this setting, the task of pattern classification reduces to selection of the state weight vector for each pattern class, and this decision is based on maximizing the language measure generated by different pattern classes. The concept is formally presented hereinafter.

Let a deterministic finite-state automaton (DFSA) be represented as $G \triangleq (Q, \Sigma, \delta, q_{\text{init}}, Q_m)$, where Q is the finite set of states with $|Q| = n$; $q_{\text{init}} \in Q$ is the initial state; Σ is the (finite) alphabet of symbols, i.e., $|\Sigma| \in \mathbb{N}_1 \triangleq \{1, 2, 3, \ldots\}$; the Kleene closure of Σ is denoted as $\Sigma^\star$ that is the set of all finite-length symbol strings including the empty string ϵ (with $|\epsilon| = 0$); the (possibly partial) function $\delta : Q \times \Sigma \to Q$ represents state transitions; $\delta^\star : Q \times \Sigma^\star \to Q$ is an extension of δ; and $Q_m \subseteq Q$ is the set of marked (i.e., accepting) states.

An observed time series is first converted to a symbol sequence that, in turn, is modeled as a PFSA. This PFSA is a DFSA augmented by a *state weight vector* and *symbol generation probabilities* [15] as described hereinafter.

Definition 1: The state weight vector $\chi : Q \to [-1, 1]$ assigns a signed real weight to each state $q_i, i = 1, 2, \ldots, n$ such that relatively more positive weights are assigned to relatively more desirable states. The $(1\times n)$ state weight vector is defined as

$$\boldsymbol{\chi} = [\chi_1\ \chi_2\ \cdots\ \chi_n] \text{ where } \chi_j \triangleq \chi(q_j). \tag{6}$$

Definition 2: The symbol generation probabilities are specified as $\bar{\pi} : \Sigma^\star \times Q \to [0, 1]$ such that $\forall q_j \in Q$, $\forall \sigma_k \in \Sigma$, and $\forall s \in \Sigma^\star$, the following statements are true, respectively.

1) $\tilde{\pi}[\sigma_k, q_j] \triangleq \tilde{\pi}_{jk} \in [0, 1]$, and $\sum_k \tilde{\pi}_{jk} = 1$.
2) $\tilde{\pi}[\sigma_k, q_j] = 0$ if $\delta(q_j, \sigma_k)$ is undefined, and $\tilde{\pi}[\epsilon, q_j] = 1$.
3) $\tilde{\pi}[\sigma_k s, q_j] = \tilde{\pi}[\sigma_k, q_j]\tilde{\pi}[s, \delta(q_j, \sigma_k)]$.

Elements of the $\widetilde{\mathbf{\Pi}}$ matrix are defined as $\widetilde{\mathbf{\Pi}}_{jk} \triangleq \tilde{\pi}(q_j, \sigma_k)$, where $q_j \in Q$ and $\sigma_k \in \Sigma$.

Remark 1: The $\widetilde{\mathbf{\Pi}}$ matrix is analogous to the morph matrix of a Markov chain in the sense that an element $\tilde{\pi}_{ij}$ represents the probability of the jth symbol occurring at the ith state.

Definition 3: The probabilistic state transition map of the PFSA is defined as a function $\pi : Q \times Q \to [0, 1]$ such that

$$\pi(q_j, q_k) = \begin{cases} 0, & \text{if } \{\sigma \in \Sigma : \delta(q_j, \sigma) = q_k\} = \emptyset \\ \displaystyle\sum_{\sigma \in \Sigma : \delta(q_j, \sigma) = q_k} \tilde{\pi}(\sigma, q_j) \triangleq \pi_{jk}, & \text{otherwise.} \end{cases} \tag{7}$$

The $\mathbf{\Pi}$ matrix is defined as $\mathbf{\Pi}_{ij} = \pi(q_j, q_k)$, where $q_j, q_k \in Q$.

Remark 2: The $\mathbf{\Pi}$ matrix is the state transition probability matrix of an irreducible Markov chain in the sense that an element π_{jk} is the transition probability from state q_j to state q_k. The language generated by the PFSA under consideration is a sublanguage of the Kleene closure Σ^* of the alphabet Σ. In the SDF setting, (quasi-)stationarity of time series implies that the associated DFSA is independent of the initial state $q_{\text{init}} \in Q$ and every state is a marked state, i.e., $Q_m = Q$. Therefore, the resulting PFSA could be represented as a 4-tuple $(Q, \Sigma, \Pi, \boldsymbol{\chi})$.

Definition 4: (Language Measure [15]): A measure of the generated language of a PFSA with respect to a given state weight vector is defined as

$$\bar{\boldsymbol{\nu}}(\theta) = \theta \left[I - (1 - \theta)\mathbf{\Pi}\right]^{-1} \boldsymbol{\chi}^{\mathrm{T}} \tag{8}$$

where the scalar parameter $\theta \in (0, 1)$ ensures invertibility of the matrix on the right-hand side of (8) and implies a terminating automaton. As $\theta \to 0^+$, the terminating automaton converges to a nonterminating automaton [15].

Proposition 1: Following Definition 4, the limiting measure vector $\bar{\boldsymbol{\nu}}(0) \triangleq \lim_{\theta \to 0^+} \bar{\boldsymbol{\nu}}(\theta)$ exists and $\|\bar{\boldsymbol{\nu}}(0)\|_\infty \leq 1$.

Proof: See [15]. ■

Proposition 2 (Scalar Measure [15]): Given a primitive (i.e., irreducible and acyclic) state transition matrix $\mathbf{\Pi}$, the measure vector in Definition 8 is given by the expression $\bar{\boldsymbol{\nu}}(0) = \nu\mathbf{1}$, where $\mathbf{1} \triangleq [1\ 1\ \cdots\ 1]^{\mathrm{T}}$. Then, the scalar measure ν is denoted as

$$\nu = \boldsymbol{p}\boldsymbol{\chi}^{\mathrm{T}} \quad \text{or} \quad \nu = \boldsymbol{\chi}\boldsymbol{p}^{\mathrm{T}} \tag{9}$$

where $\boldsymbol{p}$ is the $(1 \times n)$ state probability vector which is the (sum-normalized) left eigenvector of $\mathbf{\Pi}$ corresponding to its unique unity eigenvalue [19]. (Note that each element of $\boldsymbol{p}$ is strictly positive due to irreducibility of $\mathbf{\Pi}$.)

Proof: See [15]. ■

Now, the language measure classifier is synthesized by computing a state weight vector $\boldsymbol{\chi}$ for each pattern class $\xi_i \in \Xi$ from the respective $(1 \times n)$ reference probability vector $\{\bar{\boldsymbol{p}}_i, i = 1, \ldots, |\Xi|\}$. Given the reference patterns $\bar{\boldsymbol{p}}_i$'s, let

$$\mathcal{H}_i \triangleq \left[\bar{\boldsymbol{p}}_1^{\mathrm{T}} \cdots \bar{\boldsymbol{p}}_{i-1}^{\mathrm{T}} \bar{\boldsymbol{p}}_{i+1}^{\mathrm{T}} \cdots \bar{\boldsymbol{p}}_{|\Xi|}^{\mathrm{T}}\right]. \tag{10}$$

Corresponding to each pattern ξ_i, the state weight vector $\boldsymbol{\chi}_i$ is assigned as

$$\boldsymbol{\chi}_i \triangleq \bar{\boldsymbol{p}}_i \left[I - \mathcal{H}_i \left(\mathcal{H}_i^{\mathrm{T}} \mathcal{H}_i\right)^{-1} \mathcal{H}_i^{\mathrm{T}}\right]. \tag{11}$$

Remark 3: The $n \times (|\Xi| - 1)$ matrix $\mathcal{H}_i$ in (10) has rank $(|\Xi| - 1)$ because the $(1 \times n)$ reference probability vectors $\bar{\boldsymbol{p}}_i$'s are linearly independent and $|\Xi| \leqq n$ (see Section II-A). Therefore, the $(|\Xi| - 1) \times (|\Xi| - 1)$ matrix $(\mathcal{H}_i^{\mathrm{T}} \mathcal{H}_i)$ is invertible.

The state weight $\boldsymbol{\chi}_i$ in (11) is also interpreted based on the concept of parity space [20]. In this setting, the projection matrix $[I - \mathcal{H}_i(\mathcal{H}_i^{\mathrm{T}}\mathcal{H}_i)^{-1}\mathcal{H}_i^{\mathrm{T}}]$ in (11) is expressed as

$$V_i^{\mathrm{T}} V_i = \left[I - \mathcal{H}_i(\mathcal{H}_i^{\mathrm{T}}\mathcal{H}_i)^{-1}\mathcal{H}_i^{\mathrm{T}}\right] \tag{12}$$

where V_i is known as the $(n - |\Xi| + 1) \times n$ parity matrix for the pattern class ξ_i and $V_i \boldsymbol{p}^{\mathrm{T}}$ is the corresponding $(n - |\Xi| + 1) \times 1$ parity vector for a time-series data set that generates the stationary probability vector $\boldsymbol{p}$.

Let $\psi_i \triangleq V_i \bar{\boldsymbol{p}}_i^{\mathrm{T}}$ and $\phi_i \triangleq V_i \boldsymbol{p}^{\mathrm{T}}$ be the $((|\Xi| - 1) \times 1)$ parity vectors for the reference probability vector $\bar{\boldsymbol{p}}_i$ and the observed probability vector $\boldsymbol{p}$, respectively. Then, the inner product is computed as

$$\langle \psi_i, \phi_i \rangle = \psi_i^{\mathrm{T}} \phi_i = \bar{\boldsymbol{p}}_i V_i^{\mathrm{T}} V_i \boldsymbol{p}^{\mathrm{T}}. \tag{13}$$

Following (11) and (12), the state weights are assigned as

$$\boldsymbol{\chi}_i = \bar{\boldsymbol{p}}_i V_i^{\mathrm{T}} V_i, \qquad i = 1, \ldots, |\Xi| \tag{14}$$

and a combination of (14) and (13) yields

$$\langle \psi_i, \phi_i \rangle = \boldsymbol{\chi}_i \boldsymbol{p}^{\mathrm{T}} = \boldsymbol{p} \boldsymbol{\chi}_i^{\mathrm{T}}. \tag{15}$$

Equivalently, the inner product $\langle \psi_i, \phi_i \rangle$ is the same as the language-theoretic measure defined in (9).

The magnitude $\|V_i \boldsymbol{p}^{\mathrm{T}}\|$ of the parity vector determines whether $\boldsymbol{p}$ belongs to the pattern class ξ_i. Following the concept of orthogonal projection related to the parity space, it follows that $\|V_i \boldsymbol{p}^{\mathrm{T}}\|$ is large if $\boldsymbol{p} = \bar{\boldsymbol{p}}_i$ is correct, i.e., ξ_i is the correct pattern, and that $\|V_i \boldsymbol{p}^{\mathrm{T}}\|$ is small if $\boldsymbol{p} = \bar{\boldsymbol{p}}_i$ is incorrect, i.e., ξ_i is an incorrect pattern.

Let $\boldsymbol{p}$ be the stationary $(1 \times n)$ state probability vector of the PFSA constructed from an observed time series that is symbolized by using a given partition $\wp$. Then, the decision for pattern classification is made by maximizing the (scalar) language measure (9) in terms of the state weight vector $\boldsymbol{\chi}_i$ in (11) as

$$d^\star = \arg\max_i \boldsymbol{p} \boldsymbol{\chi}_i^{\mathrm{T}}. \tag{16}$$

Remark 4: It follows from (10) and (11) that the state weight vectors $\boldsymbol{\chi}_i$, $i = 1, 2, \ldots, |\Xi|$, can be assigned in terms of the precomputed features (i.e., reference patterns) $\bar{\boldsymbol{p}}_j$, $j = 1, \ldots, |\Xi|$. Therefore, the execution time of (16) for making an online decision is very small.

B. BRA for Pattern Classification

This section presents BRA classification of robot behavior patterns. The set $\{\xi_1, \xi_2, \ldots, \xi_{|\Xi|}\}$ of pattern classes is selected as the image of identity mapping from the set $\{d_1, d_2, \ldots, d_{|\Xi|}\}$ of decisions. The objective is to autonomously identify the behavior patterns (e.g., type, payload, and the kind of motion) of mobile robots, in which respective partitions are generated for *a priori* determined classes of behavioral patterns. In the sequel, it is shown how the pattern vectors identify the robot type (e.g., Segway RMP or Pioneer 2AT) and motion profile (e.g., random motion, circular motion, or square motion).

Let $\boldsymbol{x}$ be a time series that truly belongs to the pattern class ξ_j for some $j \in \{1, \ldots, |\Xi|\}$. The following definition formalizes the notion of (nonnegative real) deviation measure of a decision d_i, $i \in \{1, \ldots, |\Xi|\}$.

Definition 5: Given a time series $\boldsymbol{x}$ whose true pattern is ξ_j, the *deviation measure* $m_{ij}(\boldsymbol{x})$ of the decision d_i is defined in terms of the respective pattern $\boldsymbol{p}_j(\boldsymbol{x})$ and the reference pattern $\bar{\boldsymbol{p}}_i$ as

$$m_{ij}(\boldsymbol{x}) \triangleq \Theta\left(\bar{\boldsymbol{p}}_i, \boldsymbol{p}_j(\boldsymbol{x})\right) \tag{17}$$

where $\Theta(\bullet, \bullet)$ is a distance function that can be selected as a norm of the difference between the pattern vectors $\bar{\boldsymbol{p}}_i$ and $\boldsymbol{p}_j(\boldsymbol{x})$.

Due to noise and uncertainties prevalent in both the physical process and sensing devices, it is reasonable to treat the time series $\boldsymbol{x}$ as a random vector. It follows from (17) that the deviation measure m_{ij} compresses the random vector $\boldsymbol{x}$ as a (scalar) random variable and that $m_{ij}(\boldsymbol{x})$ would not be identically equal to zero in the almost sure sense, regardless of whether the decision d_i is correct or not. Nevertheless, it is expected that $m_{ij}(\boldsymbol{x})$ would be relatively small if d_i is the correct decision, i.e., if $i = j$.

Let $\mathcal{M}_{ij}(\boldsymbol{x})$ denote the random variable associated with the deviation measure $m_{ij}(\boldsymbol{x})$ for a time series $\boldsymbol{x}$ for the decision d_i (i.e., $\boldsymbol{x}$ belonging to the pattern class ξ_i), while $\boldsymbol{x}$ truly belongs to the jth pattern class ξ_j. Then, realization of $\mathcal{M}_{ij}(\boldsymbol{x})$ is the nonnegative real $m_{ij}(\boldsymbol{x})$ in (17). Hence, for each pattern class ξ_i, there could be decisions d_i, $i = 1, 2, \ldots, |\Xi|$, that give rise to realizations of $\mathcal{M}_{ij}(\boldsymbol{x})$, $i, j = 1, 2, \ldots, |\Xi|$, as $m_{ij}(\boldsymbol{x})$, and there would be a total of $|\Xi| \times |\Xi| = |\Xi|^2$ random variables $\mathcal{M}_{ij}$. In the sequel, the probability distribution of $\mathcal{M}_{ij}$ is denoted as $p_{\mathcal{M}_{ij}}$.

Let $P[\boldsymbol{x}|\xi_i, d_j]$ be the *a priori* probability distribution of the random vector $\boldsymbol{x}$ conditioned on the true pattern ξ_j and the decision d_i of choosing the reference probability vector $\bar{\boldsymbol{p}}_i$ that represents the pattern ξ_i. The next task is to identify an *a priori* probability distribution $P[\boldsymbol{x}|\xi_i, d_j]$ for each pair (ξ_i, d_j) by statistically (e.g., in the χ^2 sense) fitting the observed data of $\mathcal{M}_{ij}(\boldsymbol{x})$ at an allowable significance level, i.e.,

$$P[\boldsymbol{x}|\xi_j, d_i] \approx p_{\mathcal{M}_{ij}}\left(m_{ij}(\boldsymbol{x})\right) \tag{18}$$

and the *a posteriori* probabilities are given by

$$P[\xi_j|\boldsymbol{x}, d_i] = \frac{P[\boldsymbol{x}|\xi_j, d_i]P[\xi_j|d_i]}{P[\boldsymbol{x}|d_i]}. \quad (19)$$

Equation (19) is expressed in a different form as

$$P[\xi_j|\boldsymbol{x}, d_i] = \frac{P[\boldsymbol{x}|\xi_j, d_i]P[\xi_j]}{\sum_k P[\boldsymbol{x}|\xi_k, d_i]P[\xi_k]} \quad (20)$$

based on the following two assumptions.

1) The pattern classes ξ_j's form a mutually exclusive and exhaustive set. It follows from the total probability theorem that

$$P[\boldsymbol{x}|d_i] = \sum_k P[\boldsymbol{x}|\xi_k, d_i]P[\xi_k|d_i].$$

2) The prior probability of a pattern ξ_j is independent of the process of making the decision d_i, i.e.,

$$P[\xi_j|d_i] = P[\xi_j].$$

Substitution of (18) in (20) yields

$$P[\xi_j|\boldsymbol{x}, d_i] \approx \frac{p_{\mathcal{M}_{ij}}\left(m_{ij}(\boldsymbol{x})\right)P[\xi_j]}{\sum_k p_{\mathcal{M}_{ik}}\left(m_{ij}(\boldsymbol{x})\right)P[\xi_k]}. \quad (21)$$

Let the risk of making a decision d_i when truly the pattern class is ξ_j be specified as λ_{ij}. Then, the total risk of making a decision d_i becomes [6]

$$R(d_i|\boldsymbol{x}) = \sum_{j=1}^{|\Xi|} \lambda_{ij} P(\xi_j|\boldsymbol{x}, d_i) \quad (22)$$

and the decision on pattern identification is made by minimizing the risk in (22) as

$$d^\star = \arg\min_i R(d_i|\boldsymbol{x}). \quad (23)$$

The following assumptions could be made in the absence of any specific information on $P[\xi_j]$ in (21) and λ_{ij} in (22):

1) uniform probability of the prior probabilities of occurrence of the pattern classes ξ_j's, i.e., $P[\xi_j] = (1/|\Xi|)\ \forall j \in \{1, \ldots, |\Xi|\}$;
2) uniform nonzero risk for all wrong decisions and zero risk for correct decisions, i.e., $\lambda_{ij} = 1 - \delta_{ij}$, where $\delta_{ij} = \begin{cases} 1, & \text{if } i = j \\ 0, & \text{if } i \neq j \end{cases}$ is the Kronecker-delta function.

With the aforementioned choices of λ_{ij}'s and $P[\xi_j]$'s, risk minimization in (23) is equivalent to having the maximum likelihood estimate of the pattern class as

$$\arg\max_i \left(P[\boldsymbol{x}|d_i, \xi_i]\right) = \arg\max_i \left(p_{\mathcal{M}_{ii}}(m_{ij}(\boldsymbol{x}))\right). \quad (24)$$

Assuming that the prior probabilities for all ξ_j's are equal, (21) reduces to

$$P[\xi_j|\boldsymbol{x}, d_i] = \frac{p_{\mathcal{M}_{ij}}\left(m_{ij}(\boldsymbol{x})\right)}{\sum_k p_{\mathcal{M}_{ik}}\left(m_{ij}(\boldsymbol{x})\right)}. \quad (25)$$

The risk of of making decision d_i when the true hypothesis is ξ_j is chosen as $\lambda_{ij} = (1 - \delta_{ij})$. With this choice of the risk parameters, the total risk of making the decision d_i given by (22) becomes

$$R(d_i|\boldsymbol{x}) = \sum_{j \neq i} P(\xi_j|\boldsymbol{x}, d_i). \quad (26)$$

IV. Algorithms for Behavior Identification

This section lists the algorithms for both feature extraction and pattern classification from the time series of robot classes.

A. Feature Extraction Algorithms: SDF and PCA

For SDF feature extraction (see Section II-A), the partitioning $\wp$ is constructed by combining training data sets of all robot classes, followed by generation of reference pattern vectors $\bar{\boldsymbol{p}}_j$, $j = 1, \ldots, |\Xi|$, belonging to each pattern class ξ_j. Algorithm 1 describes the SDF procedure to construct a feature vector from time series, which makes use of Algorithm 2 for maximum entropy partitioning (MEP) [16], [21]. In this way, a single common partition $\wp$ and the reference patterns are constructed from the training data. Similarly, for PCA feature extraction (see Section II-B), the procedure first generates a projection matrix $\mathbf{W}_{\text{PCA}}^j$ for each pattern class $j = 1, \ldots, |\Xi|$ and then obtains the respective reference pattern vectors $\bar{\boldsymbol{p}}_j$'s. Algorithm 3 describes the procedure.

Algorithm 1: SDF for feature extraction

Input: Time-series data $\boldsymbol{x}_j$, $j = 1, 2, \ldots, |\Xi|$
Output: Reference probability vectors $\bar{\boldsymbol{p}}_j$, $j = 1, \ldots, |\Xi|$, and the common partitioning $\wp$

1: $\boldsymbol{x} = \emptyset$
2: **for** $j = 1$ to $|\Xi|$ **do**
3: $\quad \boldsymbol{x} = \boldsymbol{x} \bigcup \boldsymbol{x}_j$
4: **end for**
5: Partition $\boldsymbol{x}$ using Algorithm 2 to obtain the common partition vector $\wp$
6: **for** $j = 1$ to $|\Xi|$ **do**
7: $\quad$ Partition $\boldsymbol{x}_j$ using $\wp$ to obtain the symbol sequence $\boldsymbol{s}$
8: $\quad$ Construct PFSA G using $\boldsymbol{s}$
9: $\quad$ Compute reference probability vector $\bar{\boldsymbol{p}}_j$ for G
10: **end for**

Algorithm 2: MEP

Input: Time-series data $\boldsymbol{x}$, number of symbols $|\Sigma|$
Output: Partitioning $\wp$

1: Sort $\boldsymbol{x}$ in ascending order
2: Let $K = \text{length}(\boldsymbol{x})$
3: $\wp(1) = \boldsymbol{x}(1)$, i.e., the minimum element of $\boldsymbol{x}$
4: **for** $i = 2$ to $|\Sigma|$ **do**
5: $\quad \wp(i) = \boldsymbol{x}\ (ceil(i-1) * K/|\Sigma|))$
6: **end for**
7: $\wp(|\Sigma| + 1) = \boldsymbol{x}(K)$; maximum of $\boldsymbol{x}$

Algorithm 3: PCA for feature extraction

Input: Training time-series data sets $X^j \in \mathbb{R}^{M^j \times N^j}$, $j = 1, 2, \ldots, |\Xi|$, tolerance $\eta \in (0, 1)$
Output: Reference patterns $\{\bar{\boldsymbol{p}}_j\}$ and the projection matrices $\{\mathbf{W}^j_{\rm PCA}\}$
1: Find the minimum sample size $M = min(M^1, M^2, \ldots, M^{|\Xi|})$
2: Find the minimum time-series length $N = min(N^1, N^2, \ldots, N^{|\Xi|})$
3: **for** $j = 1$ to $|\Xi|$ **do**
4: **for** $\ell = 1$ to M^j **do**
5: Obtain the training data $\boldsymbol{x}^j_\ell$ as the ℓth row of the matrix X^j
6: Remove the DC component of $\boldsymbol{x}^j_\ell$, i.e., $\boldsymbol{x}^j_\ell \leftarrow (\boldsymbol{x}^j_\ell - \bar{\boldsymbol{x}}^j_\ell)$
7: **end for**
8: Generate the $(M^j \times N)$ training data matrix $\mathbf{X}^j_{\rm train} = [\boldsymbol{x}^j_1, \ldots, \boldsymbol{x}^j_{M^j}]$
9: the covariance matrix $\mathbf{S}^j = (1/M^j)(\mathbf{X}^j_{\rm train})^{\rm T}\mathbf{X}^j_{\rm train}$
10: Compute the (normalized) eigenvectors $\{\mathbf{v}^j_i\}$ of $\mathbf{S}^j$ with their corresponding eigenvalues $\{\lambda^j_i\}$ in the decreasing order of magnitude
11: Compute the normalized eigenvectors $\mathbf{u}^j_i = (1/\sqrt{M^j\lambda^j_i})(\mathbf{X}^j_{\rm train})^{\rm T}\mathbf{v}^j_i$
12: Find the smallest $m^j \leq M$ such that $\sum_{i=1}^{m^j} \lambda^j_i > \eta \sum_{i=1}^{M^j} \lambda^j_i$
13: **end for**
14: Compute the feature space dimension $m = max(m^1, m^2, \ldots, m^{|\Xi|})$
15: **for** $j = 1$ to $|\Xi|$ **do**
16: Construct $(N \times m)$ projection matrices $\mathbf{W}^j_{\rm PCA} \triangleq [\mathbf{u}^j_1, \mathbf{u}^j_2, \ldots, \mathbf{u}^j_m]$
17: Construct $(M^j \times m)$ data matrices $\mathbf{Y}^j_{\rm train} \triangleq \mathbf{X}^j_{\rm train}\mathbf{W}^j_{\rm PCA}$
18: Generate $(1 \times m)$ reference patterns $\bar{\boldsymbol{p}}_j = (1/M^j)\mathbf{1}\boldsymbol{Y}^j_{\rm train}$, where $\mathbf{1} \triangleq [1\ 1\ \cdots\ 1] \in \mathbb{R}^{1\times M^j}$
19: **end for**

B. Pattern Classification Algorithms: LMT and BRA

For LMT classification based on SDF feature extraction, Algorithm 4 presents the procedure of computing the state weight vectors, where the inputs are reference pattern vectors $\bar{\boldsymbol{p}}_j$, $j = 1, 2, \ldots, |\Xi|$, generated by Algorithm 1. This information is used as inputs in Algorithm 6 to generate the decision $d^\star$ for LMT classification. The decision is made by maximizing the language measure in line 7 of Algorithm 7 that is executed online.

For BRA classification based on SDF feature extraction, Algorithm 5 estimates the probability densities $p_{\mathcal{M}_{ij}}$'s. With these estimated $p_{\mathcal{M}_{ij}}$'s as inputs, Algorithm 7 performs BRA classification of patterns. In the event of a correct decision, the generated pattern vector should be very close to the chosen reference pattern, implying that the deviation measure is very small in (17). The *a posteriori* probabilities and the Bayes risk functions are then computed from $p_{\mathcal{M}_{ij}}(m_{ij})$ via (21) and (22), respectively, as shown in lines 10–12 of Algorithm 7 that is executed online. The decision $d^\star$ is to minimize the risk in line 14 of Algorithm 7.

For LMT classification based on PCA feature extraction, Algorithm 9 computes the language measures ν_j, $j = 1, \ldots, |\Xi|$, in line 4, where the inputs are the projection matrices $\{\mathbf{W}^j_{\rm PCA}\}$ generated by Algorithm 3 and the state weights $\{\boldsymbol{\chi}_i\}$ generated by Algorithm 4. This information is used as inputs to make the decision $d^\star$ for LMT classification by maximizing the language measure in line 6 of Algorithm 8 that is executed online.

For BRA classification based on PCA feature extraction, Algorithm 9 first computes the feature vectors $\{\boldsymbol{p}_i\}$ and their deviation measures $\{m_{ij}\}$ from the reference vectors $\{\bar{\boldsymbol{p}}_j\}$. Then, the probability densities $p_{\mathcal{M}_{ij}}(\bullet)$ are estimated by Algorithm 5. With these estimated probability densities as inputs, Algorithm 9 performs BRA classification of patterns similar to what is done in Algorithm 7.

In all cases of pattern classification (i.e., Algorithms 6, 7, 8, and 9), the pattern vector index for the identified decision $d^\star$ is converted into the corresponding indices of the robot i and the movement profile θ^i_ℓ (see the beginning of Section II). Upon arriving at the classification decision $d^\star$, the information on the robot ρ_i and its motion profile $\varphi_{\theta^i_\ell}$ is retrieved as follows.

1) Compute the cumulative sequence $\{0, k_1, k_1 + k_2, \ldots, \sum_{i=1}^{i=|\mathcal{R}|} k_i\}$ to form the new sequence $\{N_0, N_1, N_2, \ldots, N_{|\mathcal{R}|}\}$ from the sequence $\{k_1, k_2, \ldots, k_{|\mathcal{R}|}\}$.
2) Find i such that $N_{i-1} < d^\star \leq N_i$ and $\ell = N_i - d^\star$.
3) Conclude that the robot ρ_i has executed the $\varphi_{\theta^i_\ell}$ profile.

Algorithm 4: Computation of state weights for LMT classification

Input: reference pattern vectors $\bar{\boldsymbol{p}}_i$, $i = 1, 2, \ldots, |\Xi|$
Output: state weight vectors $\{\boldsymbol{\chi}_i\}$
1: **for** $i = 1$ to $|\Xi|$ **do**
2: Compute $\mathcal{H}_i = [\bar{\boldsymbol{p}}^{\rm T}_1 \cdots \bar{\boldsymbol{p}}^{\rm T}_{i-1}\bar{\boldsymbol{p}}^{\rm T}_{i+1} \cdots \bar{\boldsymbol{p}}^{\rm T}_{|\Xi|}]$ [See (10)]
3: Compute $\boldsymbol{\chi}_i = \bar{\boldsymbol{p}}_i[I - \mathcal{H}_i(\mathcal{H}^{\rm T}_i\mathcal{H}_i)^{-1}\mathcal{H}^{\rm T}_i]$ [See (11)]
4: **end for**

Algorithm 5: Probability density estimation for BRA classification

Input: Time-series data sets, common partitioning $\wp$, and # of sample data sets L_j, $j = 1, \ldots, |\Xi|$
Output: Probability densities $p_{\mathcal{M}_{ij}}(\bullet)\ \forall\ i, j = 1, \ldots, |\Xi|$
1: **for** $j = 1$ to $|\Xi|$ **do**
2: **for** $\ell = 1$ to L_j **do**

3: Collect time-series data $\boldsymbol{x}_j^{\ell}$
4: **for** $i = 1$ to $|\Xi|$ **do**
5: Partition $\boldsymbol{x}_j^{\ell}$ using $\wp$ to obtain symbol sequence $\boldsymbol{s}$
6: Construct PFSA G using $\boldsymbol{s}$
7: Compute state probability vector $\boldsymbol{p}_j$ for G
8: Compute deviation measure $m_{ij}^{\ell} = \Theta(\boldsymbol{p}_j, \bar{\boldsymbol{p}}_i)$
9: **end for**
10: **end for**
11: **end for**
12: From realizations $\{m_{ij}^1, \ldots, m_{ij}^{L_j}\}$ estimate the probability density for $p_{\mathcal{M}_{ij}}(\bullet)\ \forall\ i, j = 1, \ldots, |\Xi|$

Algorithm 6: Decision making for LMT classification using SDF

Input: Time series $\boldsymbol{x}$, partitioning $\wp$, state weight vectors $\boldsymbol{\chi}_i$, $i = 1, \ldots, |\Xi|$
Output: Identified pattern i.e., decision $d^{\star}$
1: Partition $\boldsymbol{x}$ using $\wp$ to generate symbol sequence $\boldsymbol{s}$
2: Construct PFSA using $\boldsymbol{s}$
3: Compute state probability vector $\boldsymbol{p}_{j^{\star}}$ for G, where $j^{\star}$ corresponds to the unknown pattern $\xi_{j^{\star}}$ that is yet to be identified
4: **for** $i = 1$ to $|\Xi|$ **do**
5: Compute the language measure $\nu_i = \boldsymbol{p}_{j^{\star}} \boldsymbol{\chi}_i^{\mathrm{T}}$
6: **end for**
7: Compute $d^{\star} = \arg\max_i \nu_i$

Algorithm 7: Decision making for BRA classification using SDF

Input: Time series $\boldsymbol{x}$, reference patterns $\bar{\boldsymbol{p}}_j$, $j = 1, \ldots, |\Xi|$, common partitioning $\wp$, and density estimates $p_{\mathcal{M}_{ij}}$, $i, j = 1, \ldots, |\Xi|$
Output: Identified pattern, i.e., decision $d^{\star}$
1: Partition $\boldsymbol{x}$ using $\wp$ to generate symbol sequence $\boldsymbol{s}$
2: Construct PFSA G using $\boldsymbol{s}$
3: Compute state probability vector $\boldsymbol{p}_{j^{\star}}$ for G, where $j^{\star}$ corresponds to the unknown pattern $\xi_{j^{\star}}$ that is yet to be identified
4: **for** $i = 1$ to $|\Xi|$ **do**
5: Compute the deviation measure $m_{ij^{\star}} = \Theta(\boldsymbol{p}_{j^{\star}}, \bar{\boldsymbol{p}}_i)$
6: **for** $j = 1$ to $|\Xi|$ **do**
7: Compute $P[\boldsymbol{x}|d_i, \xi_j]$ as $p_{\mathcal{M}_{ij}}(m_{ij^{\star}}(\boldsymbol{x}))$ [see (18)]
8: **end for**
9: **for** $j = 1$ to $|\Xi|$ **do**
10: Compute $P[\xi_j|\boldsymbol{x}, d_i]$ using (21)
11: **end for**
12: Compute the Bayes risk $R(\boldsymbol{x}|d_i)$ using (22)
13: **end for**
14: Compute $d^{\star} = \arg\min_i R(\boldsymbol{x}|d_i)$

Algorithm 8: Decision making for LMT classification using PCA

Input: Time series $\boldsymbol{x}$, projection matrices $\mathbf{W}_{\mathrm{PCA}}^j$, $j = 1, \ldots, |\Xi|$, and state weight vectors $\boldsymbol{\chi}_j$, $j = 1, \ldots, |\Xi|$
Output: Identified pattern i.e., decision $d^{\star}$
1: Remove the DC component of $\boldsymbol{x}$, i.e., $\boldsymbol{x} \leftarrow (\boldsymbol{x} - \bar{\boldsymbol{x}})$
2: **for** $i = 1$ to $|\Xi|$ **do**
3: Compute $(1 \times m)$ feature vectors $\boldsymbol{p}_i = \boldsymbol{x}\mathbf{W}_{\mathrm{PCA}}^i$
4: Compute the language measure $\nu_i = \boldsymbol{p}_i \boldsymbol{\chi}_i^{\mathrm{T}}$
5: **end for**
6: Compute $d^{\star} = \arg\max_i \nu_i$

Algorithm 9: Decision making for BRA classification using PCA

Input: Time series $\boldsymbol{x}$, reference patterns $\bar{\boldsymbol{p}}_j j = 1, \ldots, |\Xi|$, projection matrices $\mathbf{W}_{\mathrm{PCA}}^j$, $j = 1, \ldots, |\Xi|$, and density estimates $p_{\mathcal{M}_{ij}}$, $i, j = 1, \ldots, |\Xi|$
Output: Identified pattern, i.e., decision $d^{\star}$
1: Remove the DC component of $\boldsymbol{x}$, i.e., $\boldsymbol{x} \leftarrow (\boldsymbol{x} - \bar{\boldsymbol{x}})$
2: **for** $i = 1$ to $|\Xi|$ **do**
3: $(1 \times m)$ feature vectors $\boldsymbol{p}_i = \boldsymbol{x}\mathbf{W}_{\mathrm{PCA}}^i$
4: **for** $j = 1$ to $|\Xi|$ **do**
5: Compute the deviation measure $m_{ij} = \Theta(\boldsymbol{p}_i, \bar{\boldsymbol{p}}_j)$
6: Compute $P[\boldsymbol{x}|d_i, \xi_j]$ as $p_{\mathcal{M}_{ij}}(m_{ij}(\boldsymbol{x}))$ [see (18)]
7: **end for**
8: **for** $j = 1$ to $|\Xi|$ **do**
9: Compute $P[\xi_j|\boldsymbol{x}, d_i]$ using (21)
10: **end for**
11: Compute the Bayes risk $R(\boldsymbol{x}|d_i)$ using (22)
12: **end for**
13: Compute $d^{\star} = \arg\min_i R(\boldsymbol{x}|d_i)$

V. Experimental Results and Discussion

This section describes the experimental procedure and validation of the feature extraction and pattern classification methods based on the experimental results. The objective here is to identify the statistical patterns of robot behavior, which include both parametric and nonparametric uncertainties such as follows:

1) small variations in the robot mass that includes unloaded base weights of the platform itself and its payload;
2) uncertainties in friction coefficients for robot traction;
3) fluctuations in the robot motion due to small delays in commands due to communication delays and computational delays particularly if the processor is heavily loaded;
4) sensor uncertainties due to random noise in the A/D channels of the microprocessor.

A family of patterns is generated for behavior recognition of each of the robot classes. Each member of a family represents

TABLE I
PARAMETERS USED FOR VARIOUS TYPES OF MOTION

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

the pattern measure 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., square), the pattern classification algorithm is shown to be capable of detecting this change after reaching a (statistically quasi-stationary) steady state. During the brief transient period, the pattern classification is not applicable as the resulting time series is not likely to be sufficiently long and statistically stationary for accurate feature extraction.

The Segway RMP and Pioneer 2AT robots are commanded to execute three different motion trajectories, namely, *random motion*, *circular motion*, and *square motion*. Both robots were made to execute each of the three different types of trajectories on a pressure-sensitive floor of the laboratory environment for about an hour. The voltage generated by the piezoelectric pressure sensors, embedded under the floor, is converted to a digital signal by using a 10-b A/D converter over the range of 0–5 V to generate readings in the range of 0–1023. The data from the 144 sensors in the (9×16) grid floor were collected at the sampling rate of approximately 10 Hz. The 144 readings were stacked row-wise to create a 1-D array of length 144. This configuration facilitates identification of the type of robot and its motion profile, but not its exact location. A total of approximately 1440 points per second are collected for an hour to create a sufficiently long 1-D time-series data set. This procedure was repeated to collect six data sets for two different robots executing each of the three different motion behaviors. It is assumed that, during the execution of each motion, the statistical behavior of the robot is stationary and does not switch behaviors in between.

Since the robot movements influence those sensors that surround its location, only a few sensors generate significantly higher readings than the remaining sensors. A simple background subtraction was used to eliminate the readings from sensors away from the robot's location. Since the spatial distribution of the pressure-sensitive coils underneath the floor is statistically homogenous, the decisions on detection of robot behavior patterns are assumed to be statistically independent of the robot location (e.g., center of the circle, center and orientation of the square, and mean of the distribution for random motion). Table I lists the parameters for the three types of robot motion.

For the two types of robots and three types of motion, a total of six different behavior patterns are defined. The data sets for each of the six pattern classes are collected and processed to create the respective reference probability vectors $\bar{\boldsymbol{p}}_i$, $i \in 1, \ldots, |\Xi|$, where $|\Xi| = 6$. The statistics of a particular type of a robot and its motion with different parameters (e.g., Segway robot's circular motion of two different diameters) should be statistically identical. However, in reality, some differences are expected due to statistical inhomogeneity of the sensor field layout, imperfection in robot motion due to kinematic constraints, and finite length of the available time-series data.

For all cases considered in this paper, the following options have been used in the SDF procedure for construction of D-Markov machines.

1) *Partitioning method parameters*: For the Hilbert-transform-based analytical signal space partitioning (ASSP) [21] (see also the Appendix), # of radial direction cells $|\Sigma_R| = 3$ and # of angular direction cells $|\Sigma_A| = 4$; the alphabet size $|\Sigma| = |\Sigma_R| \cdots |\Sigma_A| = 12$. In this paper, MEP has been used for both radial and angular directions. Uniform probability distribution of the partitioning blocks is a consequence of MEP, where the time series is partitioned such that the regions with more information are partitioned finer and those with sparse information are partitioned coarser [22]; conceptually, MEP serves as a nonlinear discriminant that is effective for separation of pattern classes.
2) *D-Markov machine parameters*: With the alphabet size $|\Sigma| = 12$ and depth $D = 1$, the number of PFSA states $n = |\Sigma|^D = 12$.
3) *Distance function for computation of deviation measure*: Standard Euclidean norm of the difference between the pair of patterns. Since the space of pattern vectors is finite dimensional, all norms are topologically equivalent, and any other norm could have been used [14].

The aforementioned combination of the parameters $|\Sigma|$ and D is found to be adequate in this application to successfully recognize all six behavioral patterns with only eight states and was computationally very fast in the sense that the code execution time is orders of magnitude smaller than the process response time. Further increase of the alphabet size $|\Sigma|$ did not provide any noticeable improvement in the results because a finer partitioning did not generate any significant new information. Increasing the value of D beyond one was also found to be ineffective, which increases the number of states of the finite-state automaton, many of them having near-zero or zero probabilities, and requires a larger data set for computational convergence of the state probability vectors.

The efficacy of SDF for feature extraction is evaluated by comparison with PCA. The PCA method, described in Algorithm 3 in Section IV, is implemented to identify the eigendirections of the time-series data sets and to obtain the orthogonal linear operators that project the time series onto a low-dimensional compressed feature space; the feature space dimension $m = 8$ is obtained for the threshold parameter $\eta \geq 0.9$ that satisfies all six classes (see Algorithm 3).

A. Generation of Statistical Patterns

A set of $L = 50$ experiments was conducted and exhaustively divided into half training and half testing. This section describes the procedure for generation of statistical patterns for both LMT and BRA classifiers.

1) *Language measure*: As shown in Algorithms 6 and 8, the LMT classifier is deterministic and does not need any

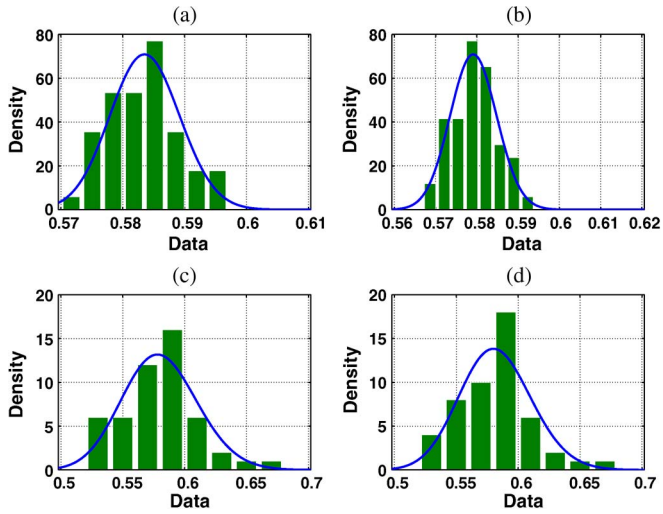


Fig. 1. Typical probability histograms of $\mathcal{M}_{ij}$ and their *lognormal* fits.

statistical computation. It acts directly upon the patterns provided by the feature extractor that could be either SDF (see Algorithm 1) or PCA (see Algorithm 3).

2) *BRA classifier*: A set of $L = 50$ experiments was conducted to generate an ensemble of realizations for each of these random variables. To compute a realization m_{ij}^{ℓ}, where $\ell \in \{1 \cdots L\}$, the following procedure is adopted.
   a) Partition the ℓth data set for pattern j using the partition vector $\wp$.
   b) Construct a D-Markov machine (of state cardinality less than or equal to $|\Sigma|^D$) for each generated symbol sequence, and compute the state probability vector $\boldsymbol{p}_j$.
   c) Compute the realization m_{ij}^{ℓ} as the distance between $\boldsymbol{p}_j$ and reference probability vector $\bar{\boldsymbol{p}}_i$.

Thus, an ensemble consisting of $L = 50$ realizations $\{m_{ij}^1, \ldots, m_{ij}^L\}$ is created for each $\mathcal{M}_{ij}$. A two-parameter *lognormal* distribution was hypothesized for each $\mathcal{M}_{ij}$. The rationale for selecting *lognormal* distribution of $\mathcal{M}_{ij}$, as opposed to other distributions (e.g., Weibull), is stated as follows.

1) The fact that the *lognormal* distribution is one directional on the position axis is consistent with the deviation measure Θ that is nonnegative [see (17)].
2) For a sample data set using the correct reference probability vector, the probability of the deviation measure being extremely close to zero is very small, but it is very likely to have a relatively large positive value for a certain range and then gradually decreases as the deviation measure increases. This is easily modeled by a *lognormal* distribution.
3) Since the random variable $\ell n(\mathcal{M}_{ij})$ is Gaussian, many standard statistical tools are available for data analysis.

Each *lognormal* distribution satisfied the 10% significance level which suffices for the conventional standard of 5% significance level. Fig. 1 shows four typical histograms of $\mathcal{M}_{ij}$ (out of a total of 36). The goodness of fit of these histograms evinces that the *lognormal* distribution is an adequate approximation for the statistics of $\mathcal{M}_{ij}$.

B. Performance Comparison

Table II presents the confusion matrices [23] to compare the performance of feature extraction (i.e., SDF and PCA) and pattern classification (i.e., LMT and BRA) algorithms for identification of both robot type and motion profile. The left column in Table II shows the confusion matrices for SDF-based feature extraction and LMT/BRA classification with LMT at the top left and BRA at the bottom left. The right column in Table II shows the confusion matrices for PCA-based feature extraction and LMT/BRA classification with LMT at the top right and BRA at the bottom right. The diagonal blocks shaded in both light and dark gray in all four blocks in Table II represent test samples with correct robot-type classification, while the diagonal elements shaded only in dark gray represent the test samples with correct motion classification. Table III summarizes the results of classification accuracy of the robot type and motion profile. Table IV presents a comparison of the computation time for SDF and PCA feature extraction (in both training and testing stages) and for LMT and BRA pattern classification, when these two classifiers are used in conjunction with either SDF or PCA feature extraction. The amount of computation time required for the feature extraction is relatively large. This is particularly true for SDF due to symbolization of data sequences and construction of PFSA. The following statements are noted: 1) LMT is an order of magnitude faster than BRA for pattern classification, when operating in conjunction with either SDF or PCA feature extraction, and 2) PCA is faster than SDF in both training and testing stages at the cost of significantly lower performance for classification of robot motion patterns.

The lower performance of PCA can be attributed to the fact that principal components (i.e., eigenvectors with largest eigenvalues) may not be the best discriminating features when pattern classes are close to each other in the feature space. It also explains why PCA performs better in classifying robot type, where pattern classes are relatively far apart; however, PCA performs significantly worse when classifying different motion profiles for a robot, where pattern classes are close together. Another possible reason for relatively superior performance of SDF is the MEP (see Algorithm 2 in Section IV) that induces a nonlinear discriminant for class separation, whereas PCA is a linear operation on the covariance matrix that separates the space of eigenvectors based on their respective eigenvalues. The results of comparison of the feature extraction and pattern classification methods are summarized as follows.

1) The feature extraction performance of PCA is close to that of SDF in robot-type classification for both LMT and BRA classifiers. However, SDF outperforms PCA in motion classification in both cases of LMT and BRA.
2) The performances of both LMT and BRA classifiers are comparable for both types of feature extraction, with BRA being modestly superior. However, the LMT classifier is significantly simpler in terms of computational complexity, as shown by inspection of the algorithms in Section IV. This computational advantage accrues from the fact that the LMT classifier does not require

TABLE II
CONFUSION MATRICES OF ROBOT BEHAVIOR RECOGNITION WITH SDF/PCA FEATURE EXTRACTION AND LMT/BRA CLASSIFICATION

SDF + LMT		Segway			Pioneer		
		Rand.	Circ.	Sq.	Rand.	Circ.	Sq.
Segway	Rand.	16	0	1	0	7	1
	Circ.	0	19	0	6	0	0
	Sq.	7	0	15	0	2	1
Pioneer	Rand.	0	5	0	20	0	0
	Circ.	1	0	0	0	23	1
	Sq.	1	0	1	0	9	14

PCA + LMT		Segway			Pioneer		
		Rand.	Circ.	Sq.	Rand.	Circ.	Sq.
Segway	Rand.	14	0	0	11	0	0
	Circ.	8	0	0	16	1	0
	Sq.	14	0	0	10	0	1
Pioneer	Rand.	3	0	0	22	0	0
	Circ.	0	0	0	24	1	0
	Sq.	1	0	0	24	0	0

SDF + BRA		Segway			Pioneer		
		Rand.	Circ.	Sq.	Rand.	Circ.	Sq.
Segway	Rand.	21	0	3	0	0	1
	Circ.	0	21	0	4	0	0
	Sq.	7	0	16	0	0	2
Pioneer	Rand.	0	9	0	16	0	0
	Circ.	0	0	0	0	23	2
	Sq.	1	0	6	0	3	15

PCA + BRA		Segway			Pioneer		
		Rand.	Circ.	Sq.	Rand.	Circ.	Sq.
Segway	Rand.	3	4	14	0	2	2
	Circ.	3	2	16	1	2	1
	Sq.	0	3	20	1	1	0
Pioneer	Rand.	3	6	4	0	7	5
	Circ.	0	2	0	1	12	10
	Sq.	1	1	1	1	14	7

TABLE III
COMPARISON OF THE CLASSIFICATION ACCURACY BY USING SDF/PCA FEATURE EXTRACTION AND LMT/BRA CLASSIFICATION

Classification Accuracy	SDF		PCA	
	LMT	BRA	LMT	BRA
Robot Type Pattern	83.3%	84.7%	71.3%	81.3%
Robot Motion Pattern	71.3%	74.7%	24.7%	29.3%

TABLE IV
COMPUTATION TIME FOR EXECUTION OF THE ALGORITHMS (25×6 TRAINING SAMPLES AND 25×6 TESTING SAMPLES)

Computation Time	SDF		PCA	
	LMT	BRA	LMT	BRA
Feature Extraction (train)	284.2 s	396.8 s	34.8 s	38.9 s
Feature Extraction (test)	112.1 s	112.1 s	3.60 s	3.60 s
Pattern Classification	0.03 s	0.37 s	0.04 s	0.42 s

estimation of the probability density functions for the random variables $\mathcal{M}_{ij}$'s (see Section III-B).

VI. SUMMARY, CONCLUSION, AND FUTURE WORK

This paper has presented a dynamic data-driven method for identification of behavior patterns in autonomous agents such as mobile robots. The proposed method utilizes SDF [14] to model the statistical behavior patterns of mobile robots that, in turn, serve as extracted features for classification of robot behavior patterns (e.g., the type of robot and the kind of robot motion) in real time. The decision of pattern classification is based on time-series data collected from an array of sensors on a pressure-sensitive floor, over which the mobile robots make movements. The SDF-based feature extraction method has the following distinct properties in contrast to standard feature extraction tools (e.g., PCA) [6], [7].

1) Fully automated pattern identification in the symbol space via coarse graining of the transformed time series. This property allows usage of relatively low-precision and inexpensive commercially available sensors.
2) Robustness to parametric and nonparametric uncertainties due to, for example, phase distortion and imprecise initial conditions.
3) Insensitivity to environmental and spurious disturbances due to the inherent noise suppression capability of SDF [14].

Upon extraction of the robot behavior features, based on SDF and PCA, an LMT method [15] of pattern classification is investigated by comparison with BRA [6]. These two methods of pattern classification have been experimentally evaluated on a networked robotic test bed to identify the type and motion profile of the robots under consideration. Their detection performance is comparable (with BRA being marginally superior to LMT) for identification of both robot type and motion profile if SDF-based feature extraction is adopted; however, LMT classification is computationally significantly faster than BRA classification. The rationale is that the LMT classification is deterministic and does not require estimation of probability distributions of the random variables. The performance of PCA as a feature extractor is found to be inferior to that of SDF when used for classification of robot type and is much worse for classification of robot motion.

Further theoretical and experimental research is needed before its implementation beyond the laboratory environment. Topics of future research include the following.

1) *Comparison of SDF with other existing feature extraction tools for robot behavior identification under different operating (e.g., change of payload) and environmental (e.g., change of terrain) conditions*: Performance evaluation of SDF has been reported for other applications such as fatigue damage detection and nonlinear active electronic systems [24] for different operating conditions. Similar investigation is needed for robot behavior identification.
2) *Behavior identification of multiple robots simultaneously operating in a dynamic environment*: This scenario poses

a challenging problem because the time-series data are intermingled and signal separation becomes very difficult, particularly in real time. A possible solution to circumvent this problem is spatial localization of data by noting the initial positions and keeping track of the robot motion throughout the operation.

3) *Learning and pattern discovery from time series*: The pattern classification method, presented in this paper, makes online decisions on *a priori* determined classes of robot type and motion. It does not address discovery of new pattern classes (i.e., encountering a new type of robot or an unknown motion profile). Future research should address the research problems in learning and pattern discovery, which are important in different applications such as autonomy of multivehicle operation in changing environments. This research should also explore alternative methods of pattern classification such as additive support vector machines [25] that combine the simplicity and transparency/interpretability of linear classifiers with the generalizing performance of nonlinear models.
4) *Augmentation of data-driven analysis with model-based information*: If the domain knowledge is available for dynamic models of robot motion and the flexible pressure-sensitive floor, integration of the model-based and data-driven information would improve the performance of pattern classification.

APPENDIX
SDF

This appendix succinctly reviews the theory and salient properties of SDF based on the previous work that has been reported in recent literature [14], [21], [22], [24]. The core concept of SDF is built upon the fundamental principles of symbolic dynamics, finite-state automata, and pattern recognition.

Symbol sequence generation from time series is an essential ingredient of SDF, where the length of the time series is determined based on a stopping rule [26]. A data sequence (e.g., time series) is converted to a symbol sequence by partitioning into finitely many discrete cells, $\Phi_1, \Phi_2, \ldots, \Phi_m$. These cells form an exhaustive and mutually exclusive set, i.e.,

$$\bigcup_{j=1}^{|\Sigma|} \Phi_j = \Omega \text{ and } \Phi_j \bigcap \Phi_k = \emptyset \; \forall j \neq k. \tag{27}$$

Each cell is labeled as a symbol $\sigma \in \Sigma$, where the symbol set Σ is called the alphabet, consisting of different symbols such as the alphabet cardinality $|\Sigma|$. If a data point falls in a particular partitioning cell, then it is assigned the corresponding symbol of that cell; thus, the data sequence is transformed into a symbol sequence in this manner.

A. ASSP

Several partitioning techniques have been reported in literature for symbol generation; examples are *symbolic false nearest neighbor* [27] and *wavelet space partitioning* [22]. The method adopted in this paper is the ASSP [21] that is briefly described hereinafter.

Let $\{\widetilde{x}_k\}$ be the Hilbert transform [28] of a real-valued time series $\{x_k\}$, which is defined as its convolution product (denoted by $\star$) with $(1/\pi)\{1/k\}$, i.e., $\{\widetilde{x}_k\} \triangleq \{x_k\} \star (1/\pi)\{1/k\}$, which is represented in the Fourier domain as

$$\mathcal{F}[\widetilde{x}](\xi) = -i\text{sgn}(\xi)\mathcal{F}[x](\xi) \tag{28}$$

where

$$\text{sgn}(\xi) = \begin{cases} +1, & \text{if } \xi > 0 \\ -1, & \text{if } \xi < 0. \end{cases} \tag{29}$$

The corresponding complex-valued analytic signal sequence $\{\mathcal{A}x_k\}$ is defined as

$$\mathcal{A}x_k \triangleq (x_k + i\widetilde{x}_k) \Leftrightarrow \mathcal{A}x_k = a_k \exp(i\varphi_k) \tag{30}$$

where a_k and φ_k are called the instantaneous amplitude and instantaneous phase of $\mathcal{A}x_k$, respectively.

It follows from (28)–(30) that the analytic signal sequence $\{\mathcal{A}[x]_k\}$ is obtained from the inverse Fourier transform of $\mathcal{F}[\mathcal{A}x](\xi)$ that is directly obtained from the Fourier transform $\mathcal{F}[x](\xi)$ of the real-valued time series $\{x_k\}$ as

$$\mathcal{F}[\mathcal{A}x](\xi) = \begin{cases} 2\mathcal{F}[x](\xi), & \text{if } \xi > 0 \\ 0, & \text{if } \xi < 0. \end{cases} \tag{31}$$

Thus, given a set of real-valued time-series data, the Hilbert transform of this data set yields a pseudophase plot that is constructed from the analytic signal sequence by a bijective mapping of the complex domain onto the $\mathbb{R}^2$, e.g., by plotting the magnitude and phase parts of the analytic signal sequence as a polar plot. The time-dependent analytic signal in (30) is now represented as a (1-D) trajectory in the 2-D pseudophase space.

The magnitude and phase of the time-dependent analytic signal in (30) can be partitioned separately according to either uniform partitioning, MEP, or any other type of partitioning; the type of partitioning may depend on the characteristics of the physical process. In this way, each point in the data set is represented by a pair of symbols—one belonging to the alphabet Σ_R based on the magnitude (i.e., in the radial direction) and the other belonging to the alphabet Σ_A based on the phase (i.e., in the angular direction). The analytic signal sequence is eventually partitioned as a symbol sequence by associating each pair of symbols into a symbol from a new alphabet Σ as

$$\Sigma \triangleq \{(\sigma_i, \sigma_j) : \sigma_i \in \Sigma_R, \sigma_j \in \Sigma_A\} \text{ and } |\Sigma| = |\Sigma_R| \cdot |\Sigma_A|.$$

Once the symbol sequence is generated, the next step is to construct a PFSA as described in the next section.

B. Construction of PFSA

This section describes the construction of finite-state automata from a symbol sequence, where the quasi-stationary histograms of the state probability distribution represent patterns that are indicative of the nominal (or reference) and anomalous behavior of the dynamical system.

A PFSA is constructed, where the PFSA states are defined corresponding to a given alphabet Σ and window length D. The alphabet size $|\Sigma|$ is the total number of partitions, while the window length D is the length of consecutive symbol words forming the states of the automaton [14]. The states of the automaton are chosen as words of length D from the symbol sequence, thereby making the number n of states to be less than or equal to the total permutations of the symbols within words of length D (i.e., $n \leq |\Sigma|^D$). The choice of $|\Sigma|$ and D depends on specific experiments, noise level, partitioning, and the available computation power. A large alphabet may be noise sensitive, while a small alphabet could miss the details of signal dynamics. Similarly, a high value of D is capable of capturing more dynamic information but would lead to larger number of states requiring more computation power and longer data sets. For PFSA construction, the window of length D on the symbol sequence $\cdots \sigma_{i_1}\sigma_{i_2}\cdots\sigma_{i_k}\cdots$ is shifted to the right by one symbol, such that it retains the last $(D-1)$ symbols of the previous state and appends it with the new symbol σ_{i_ℓ} at the end. The symbolic permutation in the current window gives rise to a new state. The constructed automaton is called D-Markov because of its Markov properties [14].

Definition 6: A symbolic stationary process is called D-Markov if the probability of the next symbol depends only on the previous D symbols, i.e.,

$$P\left(\sigma_{i_0}|\sigma_{i_{-1}}\cdots\sigma_{i_{-D}}\sigma_{i_{-D-1}}\cdots\right) = P\left(\sigma_{i_0}|\sigma_{i_{-1}}\cdots\sigma_{i_{-D}}\right). \tag{32}$$

The PFSA constructed earlier has D-Markov properties because the probability of occurrence of symbol σ_{i_ℓ} on a particular state depends only on the configuration of that state, i.e., previous D symbols. Once the alphabet Σ and word length D are assigned at the nominal condition at the time epoch t_0, they are kept invariant for subsequent (slow-scale) epochs $\{t_1, t_2, \ldots, t_k, \ldots\}$, i.e., the PFSA structure is kept fixed, although its arc probabilities may vary. The PFSA states are marked with the corresponding symbolic word permutation, and edges joining the states indicate occurrence of a symbol σ_{i_ℓ}.

Definition 7: The probability of transitions from state q_j to state q_k belonging to the set Q of states under a transition $\delta : Q \times \Sigma \rightarrow Q$ is defined as

$$\pi_{jk} = P\left(\sigma \in \Sigma | \delta(q_j, \sigma) \rightarrow q_k\right); \sum_k \pi_{jk} = 1. \tag{33}$$

For a D-Markov machine under quasi-stationary conditions, the irreducible[1] stochastic matrix $\mathbf{\Pi} \equiv [\pi_{ij}]$ describes all transition probabilities between states such that it has at most $|\Sigma|^{D+1}$ nonzero entries. The left eigenvector $\mathbf{p}$ corresponding to the unique unity eigenvalue of $\mathbf{\Pi}$ is the state probability vector under the (fast time scale) stationary condition of the dynamical system [14]. For computation of state transition probabilities from a given symbol sequence $\cdots \sigma_{i_1}\sigma_{i_2}\cdots\sigma_{i_l}\cdots$ generated from the time-series data collected at slow-time epoch t_k, a D-block (i.e., a window of length D) is moved by counting occurrences of symbol blocks $\sigma_{i_1}\cdots\sigma_{i_D}\sigma_{i_{D+1}}$ and $\sigma_{i_1}\cdots\sigma_{i_D}$ which are, respectively, denoted by $N(\sigma_{i_1}\cdots\sigma_{i_D}\sigma_{i_{D+1}})$ and $N(\sigma_{i_1}\cdots\sigma_{i_D})$. Note that if $N(\sigma_{i_1}\cdots\sigma_{i_D}) = 0$, then the state $q \equiv \sigma_{i_1}\cdots\sigma_{i_D} \in Q$ has zero probability of occurrence. For $N(\sigma_{i_1}\cdots\sigma_{i_D}) \neq 0$, the transition probabilities are then obtained by these frequency counts as follows:

$$\begin{aligned}\pi_{jk} \triangleq P\left[q_k|q_j\right] &= \frac{P\left(\sigma_{i_1}\cdots\sigma_{i_D}\sigma_{i_{D+1}}\right)}{P\left(\sigma_{i_1}\cdots\sigma_{i_D}\right)} \\ &\approx \frac{N\left(\sigma_{i_1}\cdots\sigma_{i_D}\sigma_{i_{D+1}}\right)}{N\left(\sigma_{i_1}\cdots\sigma_{i_D}\right)}\end{aligned} \tag{34}$$

where the corresponding states are denoted by $q_j \triangleq \sigma_{i_1}\sigma_{i_2}\cdots\sigma_{i_D}$ and $q_k \triangleq \sigma_{i_2}\cdots\sigma_{i_D}\sigma_{i_{D+1}}$.

[1] A square nonnegative real matrix A is called irreducible [19] if, for every i and j, there exists a positive integer k such that the ijth element of the kth power of A is strictly positive. The integer k may vary with i and j.

ACKNOWLEDGMENT

The authors would like to thank Dr. E. Keller for providing technical assistance for design and construction of the laboratory facility. Any opinions, findings, and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the sponsoring agencies.

REFERENCES

[1] R. Brooks, J. Schweir, and C. Griffin, "Behavior detection using confidence intervals of hidden Markov models," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 39, no. 6, pp. 1484–1492, Dec. 2009.

[2] F. Zhang, Y. Xi, Z. Lin, and W. Chen, "Constrained motion model of mobile robots and its applications," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 39, no. 3, pp. 773–787, Jun. 2009.

[3] H. Liu, *Feature Extraction, Construction and Selection*. Boston, MA: Kluwer, 1998.

[4] I. Guyon, *Feature Extraction: Foundations and Applications*. New York: Springer-Verlag, 2006.

[5] M. Nixon, *Feature Extraction and Image Processing*, 2nd ed. Boston, MA: Academic, 2008.

[6] R. Duda, P. Hart, and D. Stork, *Pattern Classification*, 2nd ed. New York: Wiley-Interscience, 2001.

[7] C. M. Bishop, *Pattern Recognition and Machine Learning*. New York: Springer-Verlag, 2006.

[8] N. Denis and E. Jones, "Spatio-temporal pattern detection using dynamic Bayesian networks," in *Proc. 42nd IEEE Conf. Decision Control*, Maui, HI, Dec. 2003, vol. 5, pp. 4533–4538.

[9] K. Han and M. Veloso, "Automated robot behavior recognition applied to robotic soccer," in *Proc. IJCAI Workshop Team Behaviors Plan Recog.*, 1999, pp. 47–52.

[10] L. Rabiner, "A tutorial on hidden Markov models and selected applications in speech processing," *Proc. IEEE*, vol. 77, no. 2, pp. 257–286, Feb. 1989.

[11] C. Chang and C.-L. Huang, "The model-based human body motion analysis system," *Image Vis. Comput.*, vol. 18, no. 14, pp. 1067–1083, Nov. 2000.

[12] B. Guo, M. Nixon, and T. Damarla, "Acoustic vehicle classification by fusing with semantic annotation," in *Proc. 12th Int. Conf. Inf. Fusion*, Seattle, WA, Jul. 2009, pp. 232–239.

[13] T. Damarla, "Performance of sensors mounted on a robotic platform for personnel detection," in *Proc. SPIE—Sensors, and Command, Control, Communications, and Intelligence (C3I) Technologies for Homeland Security and Homeland Defense VII*, 2008, vol. 6943, p. 694309-1.

[14] A. Ray, "Symbolic dynamic analysis of complex systems for anomaly detection," *Signal Process.*, vol. 84, no. 7, pp. 1115–1130, Jul. 2004.

[15] I. Chattopadhyay and A. Ray, "Renormalized measure of regular languages," *Int. J. Control*, vol. 79, no. 9, pp. 1107–1117, Sep. 2006.

[16] S. Gupta and A. Ray, "Symbolic dynamic filtering for data-driven pattern recognition," in *Pattern Recognition: Theory and Application*. Hauppage, NY: Nova, 2007, ch. 5, pp. 17–71.
[17] I. Chattopadhyay and A. Ray, "Language-measure-theoretic optimal control of probabilistic finite-state systems," *Int. J. Control*, vol. 80, no. 8, pp. 1271–1290, Aug. 2007.
[18] G. Mallapragada, I. Chattopadhyay, and A. Ray, "Automated behavior recognition in mobile robots using symbolic dynamic filtering," *Proc. I Mech E Part I: J. Syst. Control Eng.*, vol. 222, no. 6, pp. 409–434, Sep. 2008.
[19] R. B. Bapat and T. E. S. Raghavan, *Nonnegative Matrices and Applications*. Cambridge, U.K.: Cambridge Univ. Press, 1997.
[20] J. Potter and M. Suman, "Thresholdless redundancy management with arrays of skewed instruments," *Integrity Electron. Flight Control Syst.*, vol. AGRADOGRAPH-224, pp. 15-1–15-25, 1977.
[21] A. Subbu and A. Ray, "Space partitioning via Hilbert transform for symbolic time series analysis," *Appl. Phys. Lett.*, vol. 92, no. 8, pp. 084107-1–084107-3, Feb. 2008.
[22] V. Rajagopalan and A. Ray, "Symbolic time series analysis via wavelet-based partitioning," *Signal Process.*, vol. 86, no. 11, pp. 3309–3320, Nov. 2006.
[23] R. Kohavi and F. Provost, "Glossary of terms," *Mach. Learn.*, vol. 30, no. 2/3, pp. 271–274, 1998.
[24] C. Rao, A. Ray, S. Sarkar, and M. Yasar, "Review and comparative evaluation of symbolic dynamic filtering for detection of anomaly patterns," *Signal, Image, Video Process.*, vol. 3, no. 2, pp. 101–114, Jun. 2009.
[25] M. Doumpos, C. Zopounidis, and V. Golfinopoulou, "Additive support vector machines for pattern classification," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 37, no. 3, pp. 540–550, Jun. 2007.
[26] Y. Wen and A. Ray, "A stopping rule for symbolic dynamic filtering," *Appl. Math. Lett.*, vol. 23, no. 9, pp. 1125–1128, Sep. 2010.
[27] M. Buhl and M. Kennel, "Statistically relaxing to generating partitions for observed time-series data," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 71, no. 4, pp. 046213-1–046213-14, Apr. 2005.
[28] L. Cohen, *Time-Frequency Analysis*. Upper Saddle River, NJ: Prentice-Hall, 1995.

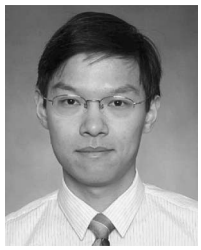
Goutham Mallapragada received the Ph.D. degree in mechanical engineering from The Pennsylvania State University, University Park, in 2009 and the M.S. degrees in electrical engineering and mechanical engineering.

He is currently a Machine Learning Scientist with Universal Robotics, Inc., Nashville, TN. His research interests include machine learning, artificial intelligence, and robotics.

Asok Ray (SM'83–F'02) received the Ph.D. degree in mechanical engineering from Northeastern University, Boston, MA, and the M.S. degrees in electrical engineering, mathematics, and computer science.

Since July 1985, he has been with The Pennsylvania State University (Penn State), University Park, where he is currently a Distinguished Professor of mechanical engineering and a Graduate Faculty of electrical engineering. Prior to joining Penn State, he also held research and academic positions with Massachusetts Institute of Technology, Cambridge, and Carnegie-Mellon University, Pittsburgh, PA, as well as research and management positions with GTE Strategic Systems Division, Charles Stark Draper Laboratory, and MITRE Corporation. He has authored or coauthored over 500 research publications, including about 250 scholarly articles in refereed journals and research monographs.

Dr. Ray is a Fellow of the American Society of Mechanical Engineers and World Innovative Foundation. He had been a Senior Research Fellow at NASA Glenn Research Center under a National Academy of Sciences award.



Xin Jin (S'09) received the B.S. degree in mechanical engineering from Shanghai Jiao Tong University, Shanghai, China, in 2007 and the M.S. degrees in electrical engineering and mechanical engineering from The Pennsylvania State University, University Park, where he is currently working toward the Ph.D. degree in mechanical engineering.

His research interests include energy management, machine learning, control systems, and robotics.

Mr. Jin is a Student Member of the American Society of Mechanical Engineers, ISA, and ANS.