Detection of Anomalous Trajectory Patterns in Target Tracking via Stochastic Context-Free Grammars and Reciprocal Process Models

Mustafa Fanaswala, *Student Member, IEEE*, and Vikram Krishnamurthy, *Fellow, IEEE*

***Abstract*—On meta-level time scales, anomalous trajectories can signify target intent through their shape and eventual destination. Such trajectories exhibit complex spatial patterns and have well defined destinations with long-range dependencies implying that Markov (random-walk) models are unsuitable. How can estimated target tracks be used to detect anomalous trajectories such as circling a building or going past a sequence of checkpoints? This paper develops context-free grammar models and reciprocal Markov models (one dimensional Markov random fields) for modeling spatial trajectories with a known end point. The intent of a target is assumed to be a function of the shape of the trajectory it follows and its intended destination. The stochastic grammar models developed are concerned with trajectory shape classification while the reciprocal Markov models are used for destination prediction. Towards this goal, Bayesian signal processing algorithms with polynomial complexity are presented. The versatility of such models is illustrated with tracking applications in surveillance.**

***Index Terms*—Intent inference, meta-level tracking, stochastic context-free grammars, reciprocal Markov processes, trajectory models, pattern of life analysis.**

I. Introduction

CLASSICAL target tracking [1] assumes a state-space model with target maneuvers (acceleration) modeled as a finite state Markov chain. Such models are useful over short time scales (order of several seconds) and several well known target tracking algorithms have been developed in the literature. This paper is motivated by *meta-level target tracking* applications on longer time scales (order of several minutes). At such time scales, the evolution of the target trajectory might not necessarily be modeled best using Markovian models. For example, an anomalous target trajectory representing malicious intent follows a pre-meditated sequence of events. Such sequences can exhibit variable long-range dependency which cannot be suitably captured using Markovian models. In meta-level tracking, one is interested in devising automated procedures that assist the human operator to detect anomalous trajectories from tracks obtained by a conventional tracking algorithm.

Manuscript received August 09, 2012; revised November 05, 2012; accepted November 29, 2012. Date of publication December 11, 2012; date of current version January 22, 2013. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Venkatesh Saligrama.

The authors are with the Department of Electrical and Computer Engineering, University of British Columbia, Vancouver, BC V6T 1Z4, Canada (e-mail: mustafaf@ece.ubc.ca; vikramk@ece.ubc.ca).

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/JSTSP.2012.2233459

In this paper, novel stochastic models and associated signal processing algorithms are developed for detecting anomalous trajectory patterns in target tracking. The trajectory of a moving target is observed by a noisy sensor (or even a network of sensors). Sensor measurements are used by a tracker to yield a sequence of "dots" representing the position of multiple targets. In this paper, we address the issue of developing syntactic models to analyze anomalous trajectories embedded within the estimated sequence of target positions (and/or velocity, acceleration). A syntactic model uses low-level features of a signal to describe the manner in which the features can combine to form more complicated structures. The main syntactic models used in this paper are *stochastic context-free grammars* (SCFG) and *stochastic reciprocal processes* (RP). Both models can be used to model long-range dependencies in signals. SCFGs have the added advantage of dealing with variable-range dependencies. *This paper develops SCFG and RP models together with stochastic filtering algorithms to assist the human operator in determining anomalous trajectories*. The algorithms presented in this paper use the track estimates from an existing tracker to perform syntactic filtering. In this sense, they are at a higher layer of abstraction than conventional tracking and are fully compatible with existing trackers.

To motivate the paper, we describe two examples from radar tracking applications. The first example involves a forensic surveillance application called *pattern of life* analysis. A pattern of life analysis involves identification of a target's daily interaction with its environment. Such a pattern of life analysis can be used to predict a target's behavior based on habit or schedule. Moreover, anomalous behavior that deviates from routine habit usually indicates an event of interest. Consider a street like that shown in Fig. 1(a). The normal behavior of the local traffic between the points (x_1, y_1) and (x_T, y_T) is observed to flow in a straight road between the two points. However, if the local population has insider information about terrorist activity (such as installing an improvised explosive device) near the marked embassy, then the local pattern of traffic flow changes to avoid the marked embassy as shown in Fig. 1(b). The ability to detect such changes in either single or aggregate target behavior requires a parsimonious representation of target trajectories. Such a representation is provided by the models developed in this paper which are also scalable towards dealing with trajectories of different shapes and sizes.

The second example involves a real-time application of anomalous trajectory classification. Suppose that prior information about the intended destination (x_T, y_T) of a target

(a) (b)

Fig. 1. (a) A map showing the normal pattern of trajectories between (x_1, y_1) and (x_T, y_T). (b) The same map showing an anomalous pattern of trajectories between (x_1, y_1) and (x_T, y_T). The deviant path signifies a strong intent by traffic to avoid the area where the embassy is located.

starting from (x_1, y_1) is available. Knowledge about the intended destination can be obtained, for example, from flight plans or shipping registries for airplanes and ships respectively. A generative model for trajectories with known destination could provide predictive information about the path of the target's trajectory for various inference tasks. A pictorial representation of this example is shown in Fig. 3. Further details are provided in Section II.

Why Use Stochastic Context Free Grammars (SCFGs) and Reciprocal Processes (RPs)?: SCFGs and RPs provide a more powerful framework to deal with long-range correlation between points in the target trajectory. In formal language theory, grammars can be classified into four different types depending on the forms of their production rules [2]. Stochastic regular grammars or finite state automata are equivalent to hidden Markov models (HMMs). SCFGs (which will be defined in Section III-A) are a significant generalization of regular grammars. Only stochastic regular grammars and SCFGs have polynomial complexity estimation algorithms and are therefore of practical use in tracking applications. It is well known in formal language theory, that SCFGs are more general than HMMs (stochastic finite automata) and can capture long range dependencies through recursively embedded structures in trajectories. Modeling anomalous trajectories with SCFGs has several potential advantages:

i) *Ability to Model Complex Trajectories*: The recursive embedding structure of certain anomalous trajectory patterns is more naturally modeled using SCFGs. As mentioned earlier, the Markovian type model has dependency that has fixed length, and the growing state space is difficult to handle since the maximum range dependency must be considered. As shown in Section III-B, SCFGs can model arcs, rectangles, closed trajectories and other anomalous trajectories like move-stop-move. A move-stop-move trajectory is a tactic used by targets to evade radar detection. Further details are provided in Section III-B.

ii) *Predictive Capacity*: SCFGs are more efficient in modeling hidden branching processes when compared to stochastic regular grammars or hidden Markov models with the same number of parameters. The predictive power of a SCFG measured in terms of entropy is greater than that of the stochastic regular grammar [3]. An SCFG is equivalent to a multi-type Galton-Watson branching process with finite number of rewrite rules, and its entropy calculation is discussed in [4].

Literature Survey: Trajectory modeling for intent inference is mainly approached in two ways: a) anomaly detection [5] or b) model-based inference [6]. In the former, a specific trajectory is not identified. Rather, all possible trajectories of a target are categorized as either normal or anomalous. For example, in [5], a support-vector machine approach is taken to classify aberrant trajectories from normal trajectories. The latter approach of model-based intent inference is taken in this paper. It involves specifically identifying models for anomalous trajectories of interest.

A key feature of model-based inference is to obtain a semantic interpretation of a complex pattern through the use of simpler sub-patterns. For example, in [6], a dynamic Bayesian network is used to identify scenarios where a shopper either enters a retail store, leaves a store or passes by the store. In this paper, we consider tracklets as the simpler sub-patterns comprising a trajectory (which is semantically equivalent to intent). The tracklets are explained in detail in Section II.

Our work is related in spirit to the approach taken in [7] and [8]. A context-free grammar approach is taken in [7] to identify two-person interactions like hugs, hand-shakes, kicks and punches. A stochastic context-free grammar approach is taken in [8] to recognize cheating actions in card-games at casinos. Our work departs from them significantly as we consider trajectory modeling in a tracking situation and not an action recognition system. The work presented in this paper builds upon the work in [9], [10] on target tracking using stochastic context-free grammars in radar tracking applications. Our work explicitly differs from [9] in that no assumptions are made on the form the target dynamics. In [9], tracklets are called *modes* which are explicitly factored into the state dynamics of the target as parameters which affect the variance of directional noise. Our work directly uses filtered state estimates to obtain trajectory tracklets. Moreover, we also build upon the framework in [9] by incorporating awareness of the target destination in the SCFG model through constraints on the rule probabilities. The presentation is this paper is also related to the approach taken in [11] where attributes are associated with a stochastic context-free grammar to

enforce constraints on the applicability of the production rules. The domain of interest is detection of anomalous activities. In this paper, constraints are also enforced albeit on the numerical value of rule probabilities to obtain a desired expected length of the target trajectory.

The use of SCFGs as a modeling tool for detecting anomalous trajectories requires the ability to compute model likelihoods. The inside-outside (IO) algorithm [3] is the conventional method used to compute the probability of an observed trajectory belonging to a given grammar model. However, the IO algorithm requires the stochastic grammar to be in a restrictive form. The Earley-Stolcke parser [12] on the other hand is able to deal with arbitrarily structured grammars and is the algorithm used in this paper for the Bayesian estimation of the model probabilities.

The use of RPs in target tracking was first shown in [13]. A continuous-time solution based on optimal smoothing is provided to predict the final destination of a ship whose starting location and final destination are known. However, our work is significantly different, because we consider a discrete-time, discrete-space version of the problem. Moreover, we use the concept of Markov bridges in predicting a target's final destination. The optimal filters used in this paper are derived in [14].

Main Results: For a unified treatment, this paper uses a general tracking framework to develop the main ideas. The notion of tracklets and the associated modeling of trajectories using SCFGs and RPs is scalable towards many sensing modalities like radar systems, multi-camera surveillance systems, geo-positioning based trackers and cellular base-station triangulation. The main results of this paper are:

1) The detection of anomalous trajectories is formulated as a classification problem in Section II. The SCFG and RP models use the output from base-level tracking algorithms to either provide feedback to enhance the tracker or to perform higher-level inference recognizing anomalous trajectories. As a result, the tools developed in this paper are legacy-compatible.
2) SCFGs are used as a modeling framework for spatial patterns like arcs, rectangles, closed paths etc. They are also used to model move-stop-move behavior which is a common evasive tactic used by targets. Towards this end, a quantized representation of velocity directions are used as low-level features. This representation allows a novel application in modeling the destination of a target by placing constraints on the SCFG rule probabilities. As a result, both shape and destination can be captured in the same generative SCFG model.
3) While SCFGs are able to constrain the final destination in an expected sense, anomalous trajectories which are sensitive to the exact destination require an explicit position-based representation. This is carried out through the novel use of RPs and Markov bridges. Towards this end, a grid-based quantization of Cartesian space is used as low-level features. Such a representation allows destination prediction of the target which can further be used to enhance the accuracy of the base-level tracker.
4) The use of SCFGs for shape classification and the use of RPs for destination prediction is combined in a probabilistic fusion framework to provide an inference that is able to use both shape and destination cues to detect anomalous trajectories of interest.
5) The experimental simulations carried out demonstrate the discriminative power of SCFG and RP models in detecting anomalous trajectories. We compare the performance of destination-constrained SCFGs with probabilistic fusion of SCFG and RP models for target intent inference.

II. Anomalous Trajectory Classification Framework

In this section, a system-level description of the anomalous trajectory classification problem is presented. We first describe the tracking framework and provide a mathematical description of trajectory classification. We then provide specific details about our tracklet estimation extension to the classical tracking approach. A diagrammatic representation of the proposed system is shown in Fig. 3. A tracking sensor is assumed to make measurements $\mathbf{y}_t$ related to the position and velocity of targets in a particular region of interest (ROI). These measurements $\mathbf{y}_t$ are utilized by a base-level tracker $\mathcal{T}$ to estimate the actual position and velocity of the targets. Such a set-up is the conventional "tracker" module used in many tracking applications. We introduce an additional module called the tracklet estimator $\mathcal{H}$ which produces quantized position estimates and velocity directions z_t using the output $\mathcal{P}_t$ of a base-level tracker. These, in turn, are used by a meta-level inference engine to determine target intent.

The base-level tracker is a nonlinear Bayesian filter (such as a particle filter, IMM algorithm etc.) which can be represented as an operator $\mathcal{T}$ that uses sensor measurements $\mathbf{y}_t$ to update a posterior distribution $\mathcal{P}_t$ over the position and velocity of the target by

$$\mathcal{P}_t = \mathcal{T}(\mathcal{P}_{t-1}, \mathbf{y}_t). \tag{1}$$

The posterior distribution in (1) is then used by the tracklet estimator to obtain quantized estimates of the target position and its velocity direction. The tracklet estimator can be represented as an operator $\mathcal{H}_i$ such that

$$z_t = \mathcal{H}_i(\mathcal{P}_t), \tag{2}$$

where $i \in \{\text{position}, \text{velocity}\}$. We consider two different tracklets viz., the position tracklets which are quantized by the operator $\mathcal{H}_{\text{position}}$ and velocity tracklets which are quantized through $\mathcal{H}_{\text{velocity}}$.

The aim of this paper is to provide models for the process z_t that can be used to classify anomalous trajectories. The target trajectory is associated with an intent depending on its shape and/or its destination. For example, a circling behavior (closed loops, rectangles, arcs) might be indicative of a reconnaissance operation in the vicinity of a sensitive asset like a check-post. A boat that is loitering near the shore-line and heading towards a known drop-off point could also be indicative of a smuggling operation.

A unified framework is developed in this paper to deal with different kinds of anomalous trajectories. The defining features of the considered trajectories are either its shape or the destination of the target. These are further characterized

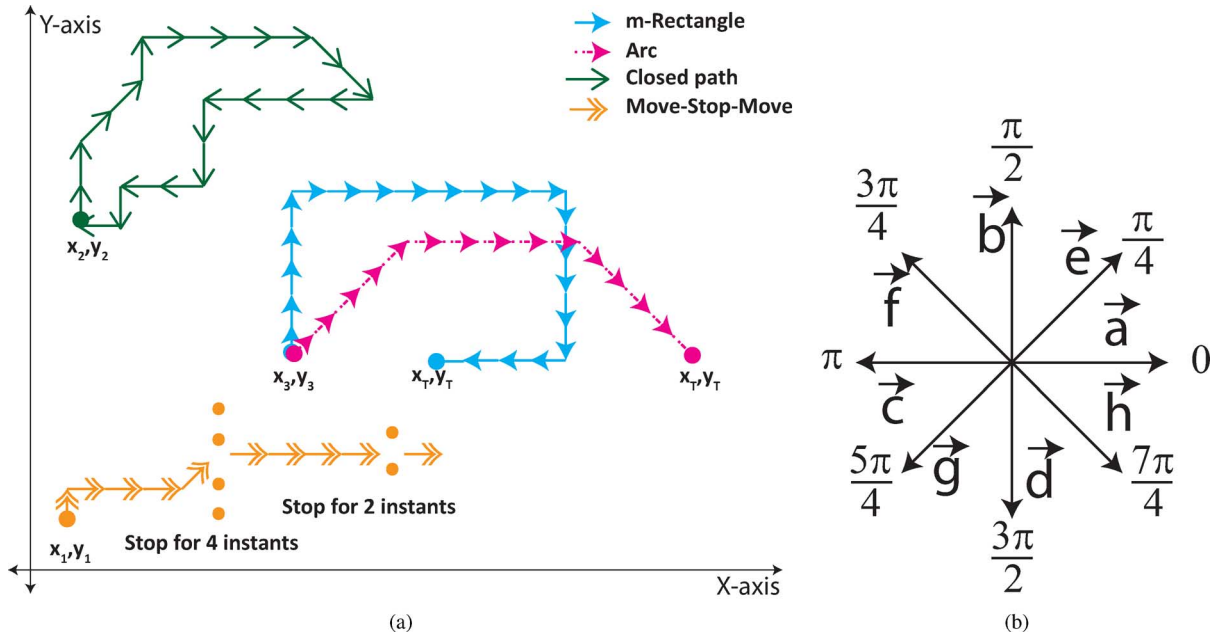


Fig. 2. (a) A rectangular trajectory and an arc trajectory are shown with different destinations. A closed trajectory is shown whose beginning and ending points coincide. A move-stop-move trajectory is shown which is a tactic used by targets to evade radar detection. (b) State estimates quantized into radial velocity directions.

through velocity or position tracklets. Each type of target intent is thus assumed to be generated by a particular model $G_k \in \mathcal{G}, k = \{1, \ldots, K\}$, where there are K different types of target intent under consideration. These models are described in Sections III-B and IV-A. As a target moves in a region of interest (ROI), it generates tracklets z_t. The anomalous trajectory classification task is then defined as finding the model G_k that has the highest probability of explaining the observed tracklet sequence $z_0, \ldots, z_t$,

$$G^* = \arg\max_{G_k \in \mathcal{G}} \mathbf{P}\{G_k \,|\, z_0, \ldots, z_t\}. \tag{3}$$

We now provide further details on the base-level tracker and tracklet estimation modules in Fig. 3. A typical tracking application involves certain assumptions on the dynamics of the target. The target dynamics are summarized using its kinematic state in the vector $\mathbf{s}_t = [x_t, y_t, \dot{x}_t, \dot{y}_t]^T$ and a description of how $\mathbf{s}_t$ evolves. The state variables (x_t, y_t) refer to the position of the target while $(\dot{x}_t, \dot{y}_t)$ refer to the velocity of the target in Cartesian co-ordinates. A tracking sensor cannot measure the kinematic state of a target directly. It can however make measurements $\mathbf{y}_t$ related to the state $\mathbf{s}_t$ which are contaminated with noise. A base-level tracker (Kalman filter, particle filter, IMM algorithm) is then used to track the state of the target $\mathbf{s}_t$ using the sensor observations $\mathbf{y}_t$. The output of such a tracker consists of a posterior probability distribution $\mathcal{P}_t = \mathbf{P}\{\mathbf{s}_t \,|\, \mathbf{y}_t\}$. The state estimate can then be evaluated from the posterior distribution using a conditional expectation. The state estimates $\mathbf{E}\{\mathbf{s}_t \,|\, \mathbf{y}_t\}$ form the input to the tracklet estimator.

The tracklet estimator $\mathcal{H}$ is a quantization module which outputs either position or velocity tracklets. Tracklets are used as sub-units which comprise target trajectories. The SCFG models utilize velocity tracklets as sub-units of the trajectory shape while RP models utilize position tracklets as sub-units of a goal-directed trajectory (with a known destination). Each type of tracklet is described below. The position tracklet estimator works with a discretized surveillance space Λ over which the target is observed. At each time instant, the position tracklet estimator quantizes the (x_t, y_t) state estimates to the closest element $\lambda_i \in \Lambda$ on the discretized 2-D grid Λ. The position tracklet estimator can be represented as

$$z_t = \arg\min_i D(x_t, y_t, \lambda_i), \tag{4}$$

where $D(x_t, y_t, \lambda_i)$ is the Euclidean distance between (x_t, y_t) and the center of the grid element λ_i.

The velocity tracklet estimator utilizes the coordinate velocity estimates $(\dot{x}_t, \dot{y}_t)$ to find the direction of motion of the target. The possible directions of motion of the target are quantized into 8 radial angular directions from the set $\mathcal{Q} = \{\vec{a} = 0, \vec{e} = (\pi)/(4), \vec{b} = (\pi)/(2), \vec{f} = (3\pi)/(4), \vec{c} = \pi, \vec{g} = (5\pi)/(4), \vec{d} = (3\pi)/(2), \vec{h} = (7\pi)/(4)\}$. The radial directions are shown in Fig. 2(b) and each is labeled for notational convenience with a lowercase alphabet and an additional $\vec{}$ to denote that is a unit directional vector. The velocity tracklet estimator thus outputs

$$z_t = \arg\max_{q \in Q} \left| \arctan\left(\frac{\dot{y}_t}{\dot{x}_t}\right) - q \right| \tag{5}$$

III. Trajectory Modeling and Inference Using Stochastic Context-Free Grammars

In this section, we present SCFG models and associated signal processing algorithms for trajectory modeling and classification. SCFGs will be the main tool that we will use to model shapes of target trajectories. The output of an SCFG is a string of terminal symbols. These terminal symbols are precisely the tracklets which we aim to model. Finally, the Earley-Stolcke

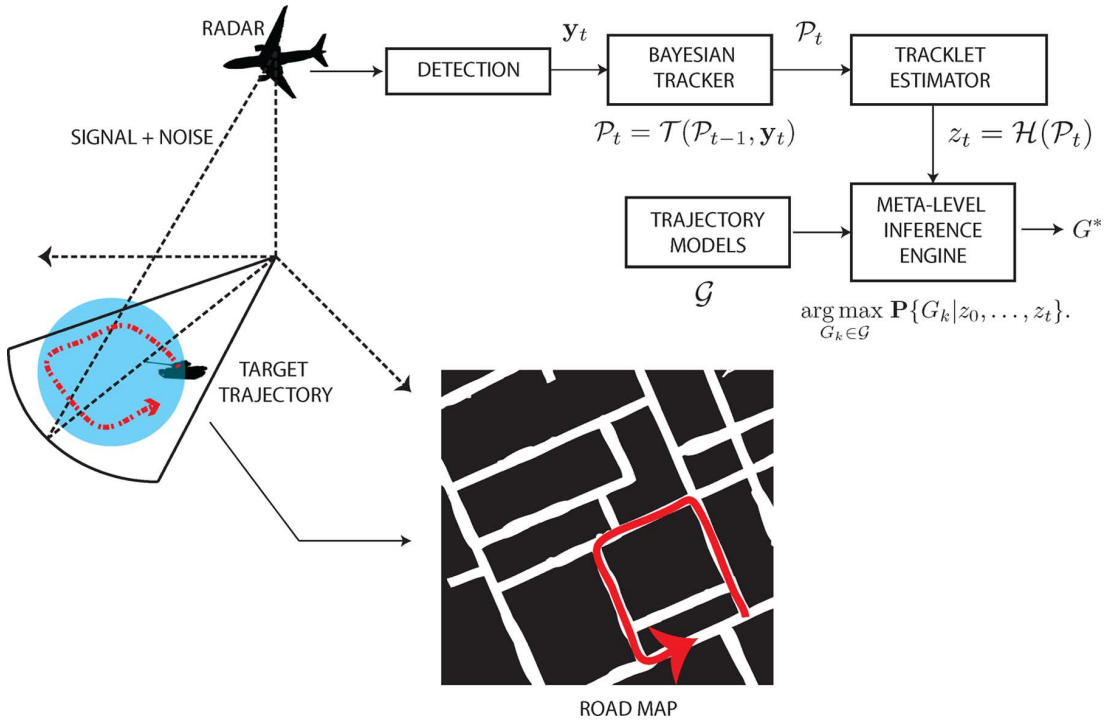


Fig. 3. The architecture of the proposed meta-level inference for target intent described in Section II.

parser is presented to perform statistical signal processing of the tracklets.

A. Review of Stochastic Context-Free Grammars

In this section, we briefly review stochastic context-free grammars. A textbook treatment can be found in [2]. A context-free grammar G_{CFG} is a 4-tuple $(\mathcal{N}, \mathcal{V}, S, \mathcal{A})$, where $\mathcal{N}$ is a finite set of non-terminals $(N_i, i = 1, \ldots, |\mathcal{N}|)$, $\mathcal{V}$ is a finite set of terminals $(v_i, i = 1, \ldots, |\mathcal{V}|)$ such that $(\mathcal{N} \cap \mathcal{V} = \emptyset)$, $S \in \mathcal{N}$ is the chosen start symbol (initial non-terminal) and $\mathcal{A}$ is a finite set of production rules a_m of the form $(A \rightarrow \alpha)$, $A \in \mathcal{N}$ and $\alpha \in (\mathcal{N} \cup \mathcal{V})^+$. The set $(\mathcal{N} \cup \mathcal{V})^+$ denotes all finite length strings of symbols in $(\mathcal{N} \cup \mathcal{V})$, excluding strings of length 0 (the case where strings of length 0 is included is indicated by $(\mathcal{N} \cup \mathcal{V})^*$). The $\rightarrow$ symbol denotes a re-write operation which replaces the non-terminal A with the string α. A stochastic context-free grammar is defined as a pair (G_{CFG}, p), where $p : \mathcal{A} \rightarrow [0, 1]$ is a probability function over the production rules $(A \rightarrow \alpha) \in \mathcal{A}$ such that $\forall A \in \mathcal{N}, \sum_{i=1}^{n_A} p(A \rightarrow \alpha_i) = 1$. The number of alternative production rules associated with A is denoted n_A.

A grammar is a generative model which produces an output sequence of terminal symbols. A symbol refers to an element of the set $(\mathcal{N} \cup \mathcal{V})$ which can be either a single non-terminal or a single terminal. A sequence of such symbols is called a string $\alpha \in (\mathcal{N} \cup \mathcal{V})^+$. When the string is complete and composed entirely of terminals (such that no further symbols can be concatenated or produced in the string), it is called a sentence. In Fig. 4, we show an example of a sentence generated by a grammar. We also demonstrate ambiguity in the generation process that can be resolved through rule probabilities.

B. SCFG Models for Anomalous Trajectories

In this section, we model various trajectories of interest using stochastic context-free grammar models. For all the SCFG models considered in this paper, each type of trajectory shape or pattern has an associated grammar model G. All the grammars have a common set of terminals $\mathcal{V} = \mathcal{Q}$ and a common start symbol S. They may have different rule spaces $\mathcal{A}$ and/or non-terminal spaces $\mathcal{N}$. While modeling trajectory shapes using grammar models, we will focus on the structure of the production rules. The non-terminal space is implicitly included when writing the production rules. In Section III-C, the production rule probabilities will be chosen to constrain the expected final destination of the target.

Line Trajectory: A target traveling in a straight path creates linear trajectories with local Markov dependency, and it is characterized by rules of the form $S \rightarrow \vec{a}S \mid \vec{a}$ with $\vec{a}$ representing the target's direction of motion. An example string of a target traveling in a straight horizontal line for four sampling instants is "$\vec{a}\vec{a}\vec{a}\vec{a}$". The production rules of a line grammar generates a language that is equivalent to that of a hidden Markov model formulation (or equivalently a regular grammar). A regular grammar is constrained to have only one non-terminal on either side of a production rule. The linear shape can be represented as the language $\mathcal{L}_{\text{line}} = \{x \in \vec{a}^n\}$. This notation implies that all strings generated by a line grammar G_{line} will have the form $\vec{a}^n$. The notation $\vec{a}^n$ implies that the terminal symbol $\vec{a}$ appears n times consecutively in a sequence. The other geometric shapes of interest are arcs, rectangles, closed loops and move-stop-move trajectories. These shapes possess long range and self-embedding dependencies that require production rules which regular grammars (and hence Markov models) cannot represent.

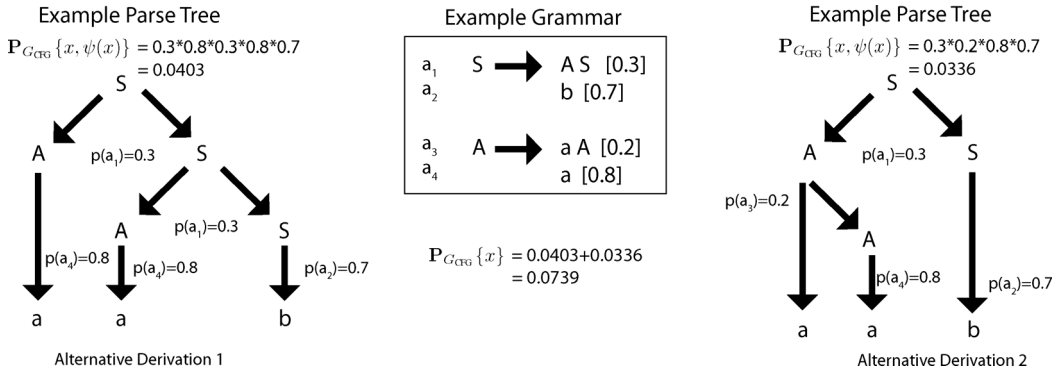


Fig. 4. An example grammar with context-free production rules and representative alternative parse trees of a generation process for the string aab. The quantities in [ ] brackets denote the probability of choosing that production rule. The individual probabilities of each parse tree is shown together with the total probability of the string.

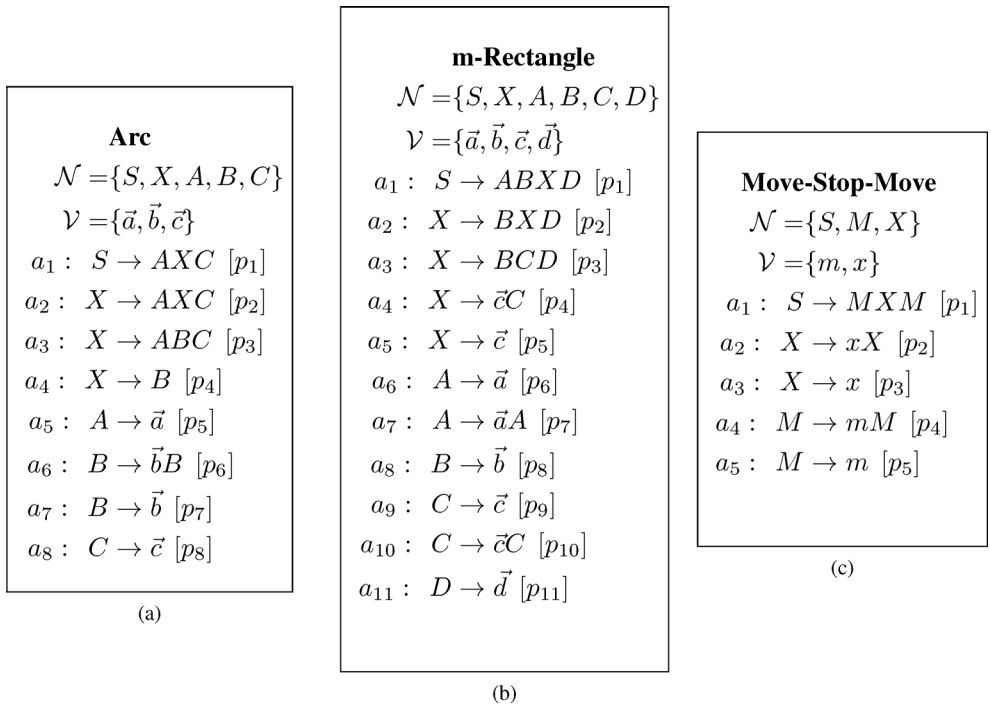


Fig. 5. An arc grammar in (a) and an m-rectangle grammar in (b). A move-stop-move grammar is shown in (c).

Arc Trajectory: An arc-shaped trajectory can be expressed as a language $\mathcal{L}_{\text{arc}} = \{x \in \vec{a}^n\vec{b}^+\vec{c}^n\}$, where there is an equal number of matching upward $\vec{a}$ and downward $\vec{c}$ tracklets and an arbitrary number of forward tracklets $\vec{b}$. The $+$ symbol denotes an arbitrary number of $\vec{b}$ symbols. The symbol x represents any arbitrary string belonging to the language. Such a language can be generated by the grammar shown in Fig. 5(a). The grammar can be constructed based on techniques reviewed in [15].

Rectangular Trajectory: The m-rectangle language (with associated grammar shown in Fig. 5(b)) is $\mathcal{L}_{\text{m-rectangle}} = \{\vec{a}^m\vec{b}^+\vec{c}^m\vec{d}^*\}$ and it can model any trajectory comprising of four sides at right angles (not necessarily a closed curve) with at least two opposite sides being of equal length. Why do we consider m-rectangles instead of rectangles? This is because the language comprising of only rectangles is not context-free. The language comprising of only rectangles can be generated by a more specific class of grammars called context-sensitive grammars. A proof of this can be found in [15]. As a result, algorithms of polynomial complexity for recognizing such trajectories cannot be constructed. However, we can construct heuristic rules by fixing the number of $\vec{b}$'s and $\vec{d}$'s in the grammar to generate rectangles and squares.

Closed Trajectory: Consider the closed trajectory in Fig. 2(a). We can resolve each directional vector onto the unit directions represented by $\vec{a}$ and $\vec{b}$. A closed figure then comprises of an equal number of $\vec{a}$ (up) and $\vec{c}$ (down) movements together with an equal number of $\vec{b}$ (left) and $\vec{d}$ (right) movements. Such a trajectory also comprises an arc-like language where an equal number of opposing movements is represented by the language $\mathcal{L}_{\text{equal}} = \{\vec{k}^n\vec{l}^n\}$, where k and l refer to opposite movements from the set $\mathcal{Q}$.

Move-Stop-Move Trajectory: A move-stop-move trajectory results from a coarse representation of the tracker state estimates which allows us to model a common evasion tactic used by targets. If the target stops moving (or its velocity drops below a threshold), then ground moving target indication (GMTI)

trackers are unable to track it [9]. As a result, targets seeking to evade a radar often intersperse periods of movement with periods of no movements. The sporadic stopping between two periods with movement can be modeling as a self-embedding grammar of the form shown in Fig. 5(c). A move-stop-move trajectory in which the target stops for four sampling instants would take the form "$mmmmxxxxmmmm$", where each m refers to a movement in any one of the directions in $\mathcal{Q}$ and x refers to a stop.

C. Constraints on SCFG Rule Probabilities

In this section, we describe two constraint conditions which ensure that the SCFG is well-posed and also allows us to constrain the expected destination of a target. The first condition is called the consistency constraint (described in [2]) which ensures that the grammar is able to terminate in a finite length string. The second condition constrains the final destination of the target in an expected sense.

In the following, we describe the manner in which the target destination is modeled. The production rule probabilities of a grammar can be chosen such that the expected value of the final destination is equal to the intended final destination. We assume that the target maintains a constant speed in the direction of motion. If the target speed is $\kappa\ m/s$, then each estimated tracklet z_t represents the movement of the target by κ meters in a radial direction represented by z_t. The total distance S_t traveled by the target until time t is given by

$$S_t = \sum_{\tau=1}^{t} \kappa z_\tau. \tag{6}$$

The co-ordinate system used is the Cartesian plane which is represented by the unit vectors $\vec{a}$ and $\vec{b}$ as shown in Fig. 2(b). Consequently, every trajectory consisting of a sequence of tracklets z_t can be written as an linear combination of $\vec{a}$ and $\vec{b}$. Each of the other unit directional vectors can also be written as a linear combination of $\vec{a}$ and $\vec{b}$ using vector addition.

Consider a target moving in an m-rectangle trajectory like that in Fig. 3. Let's assume that the target starts at an initial position (x_1, y_1) which represents the position vector $x_1\vec{a}+y_1\vec{b}$. We wish to constrain the final destination of the target to be at (x_T, y_T) representing the position vector $x_T\vec{a} + y_T\vec{b}$. This implies that the total distance traveled by the target is $(x_T - x_1)\vec{a} + (y_T - y_1)\vec{b}$. The total distance S_T can be computed as the sum of the number of times the target moves in each of the directions in $\mathcal{Q}$ (because of the constant speed assumption). This is given by

$$\begin{aligned} S_T &= \sum_{t=1}^{T} z_t = (x_T - x_1)\vec{a} + (y_T - y_1)\vec{b}, \\ &= \left(N_{\vec{a}}\vec{a} + N_{\vec{b}}\vec{b} + N_{\vec{c}}\vec{c} + \cdots + N_{\vec{h}}\vec{h}\right), \end{aligned} \tag{7}$$

where T is the total length of the string and N_{q_j} is the number of times the target moves in direction $q_j \in \mathcal{Q}$. Our interest is from a modeling perspective and hence we would like to constrain the total distance the target travels. It turns out that the probabilistic nature of the grammar production rules allows us to constrain the total distance traveled if we can bound the expected value of the total distance traveled. This is obtained from (7) by replacing each quantity with its expected value viz., the expected number of times $\mathbf{E}\{N_{q_j}\}$ that the target travels in each of the directions $q_j \in \mathcal{Q}$. To compute these expected values, we need to define the following matrices.

The matrix E has $|\mathcal{N}|$ rows indexed by non-terminals and $|\mathcal{P}|$ columns indexed by production rules. An element $E_{i,j}$ of the matrix E has value p_j if production rule a_j has non-terminal N_i on its left and 0 otherwise. Here p_j is the probability of choosing rule a_j as shown in Fig. 5.

The matrix C has $|\mathcal{P}|$ rows indexed by production rules and $|\mathcal{N}|$ columns indexed by non-terminals. An element $C_{i,j}$ of the matrix C is the number of occurrences of non-terminal N_j on the right hand side of production rule a_i.

The stochastic expectation matrix $N = E \cdot C$ is an $|\mathcal{N}| \times |\mathcal{N}|$ matrix indexed by non-terminals. An element $N_{i,j}$ of the matrix N is the expected number of times a non-terminal N_j will occur when N_i is re-written using exactly one production rule. The stochastic expectation matrix N plays an important part in determining the consistency of a grammar. A grammar is consistent if it terminates in a finite length terminal string. A grammar G is consistent if the largest eigenvalue $\rho(N)$ of its stochastic expectation matrix N is less than one [2].

We are also interested in the matrix N^∞ for any number of production rules being applied. This is given by $N^\infty = \sum_{i=0}^{\infty} N^i$. This series converges as shown in [16] to

$$N^\infty = (I - N)^{-1}, \tag{8}$$

where I is a $|\mathcal{N}| \times |\mathcal{N}|$ identity matrix. The matrix N^∞ is called the non-terminal expectation matrix. Each element $N^\infty_{i,j}$ represents the expected number of non-terminals N_j that could result from a non-terminal N_i taking into account an arbitrary number of production rules being applied.

The matrix T has $|\mathcal{P}|$ rows indexed by production rules and $|\mathcal{Q}|$ columns indexed by terminals. An element $T_{i,j}$ of the matrix T represents the number of times the terminal q_j appears on the right hand side of the production rule a_i.

The matrix $W = E \cdot T$ has $|\mathcal{N}|$ rows indexed by non-terminals and $|\mathcal{Q}|$ columns indexed by terminals. An element $W_{i,j}$ of the matrix W represents the expected number of instances of terminal q_j resulting from one re-write of the non-terminal N_i.

The terminal expectation matrix $W^\infty = N^\infty \cdot W$ has $|\mathcal{N}|$ rows indexed by non-terminals and $|\mathcal{Q}|$ rows indexed by terminals. Each element $W^\infty_{i,j}$ represents the number of instances of terminal q_j resulting from an arbitrary application of production rules starting from a non-terminal N_i.

Using the matrices defined above, we can now compute the expected word length of the grammar G. The expected word length is defined as the total number of terminals q_j derived starting from a particular non-terminal N_i. Since we are interested in the final word length, we consider only the non-terminal S which is the starting symbol (with index 1). This can be represented as

$$\mathbf{E}\left\{N_{q_j}\right\} = W^\infty_{1,j}. \tag{9}$$

Using the expected values from (9), the total distance traveled by the target can be constrained which is the same as con-

straining the end-point of the trajectory given that we know the initial starting point. The constraint equation is given by

$$\sum_j W_{1,j}^{\infty}\, \vec{q_j} = (x_T - x_1)\vec{a} + (y_T - y_1)\vec{b}, \qquad (10)$$

after resolving the 8 radial directions into constituent combinations of $\vec{a}$ and $\vec{b}$. The value of each element $W_{i,j}^{\infty}$ is a function of the production rule probabilities $p_1, \ldots, p_{|\mathcal{P}|}$. Using the constraint in (10) and the consistency constraint, we obtain one inequality constraint (the consistency constraint) and one equality constraint (the expected word length constraint). The exact form of these constraints depends on the form of the grammatical rules. In the case of the grammars described in Fig. 5, the consistency constraint turns out to be a trivial constraint. The eigenvalues of the stochastic expectation matrix N are specific rule probabilities. Since a rule probability is $0 \leq p_m \leq 1$, the consistency criteria is trivially satisfied for these grammars. As a result, we are only left with one equality constraint to satisfy.

In the case of the grammar in Fig. 5(b), the expected word length constraint results in a multi-linear equation (after multiplying by the common factor throughout) of the form

$$\frac{p_2 + p_3 - 1}{p_2 - 1} - \frac{1}{p_6 - 1} + \frac{p_3 + p_4}{(p_2 - 1)(p_9 - 1)} - \kappa = 0, \qquad (11)$$

where κ is the distance between x_T and x_1. Since we are faced with only one equation in 5 unknowns, where each unknown $0 \leq p_m \leq 1$, a brute-force approach is used to obtain a feasible solution. A 5-dimensional grid is created and the constraint equation in (11) is swept to find a feasible solution set.

D. Bayesian Signal Processing of SCFG Models

This section deals with the solution of the following sequential classification problem. Given an observation sequence of the target's estimated velocity directions $z = z_1, \ldots, z_t$, can we classify the target's trajectory? The set of permissible grammar models is given by $\mathcal{G}$ which contains all the anomalous trajectories described in Section III.B. Mathematically, we seek the grammar posterior probability

$$G^{\star} = \arg\max_{G_k} \in \mathbf{P}\{G_k \,|\, z_1, \ldots, z_t\}, \qquad (12)$$

where $G^{\star}$ is the grammar model (or corresponding anomalous trajectory) with the maximum probability given the observation sequence. The computation of the likelihoods using partial sentences $z_1, \ldots, z_t$ rather than a complete trajectory $z_1, \ldots, z_T$ is a non-trivial exercise and requires the computation of *prefix* probabilities. The prefix probability $\mathbf{P}\{z_1, \ldots, z_t; G_k\}$ of the string $z_1, \ldots, z_t$ is the probability that grammar G_k derives the string $z_1, \ldots, z_t, y$ which has $z_1, \ldots, z_t$ as its prefix and $y \in (\mathcal{N} \cup \mathcal{V})^*$ is an arbitrary suffix. The prefix probability is defined as

$$\mathbf{P}\{z_1, \ldots, z_t; G_k\} = \sum_{y \in (\mathcal{N} \cup \mathcal{V})^*} \mathbf{P}\{z_1, \ldots, z_t, y; G_k\}. \qquad (13)$$

The computation of the prefix probabilities is carried out using the Earley-Stolcke parser which is described next.

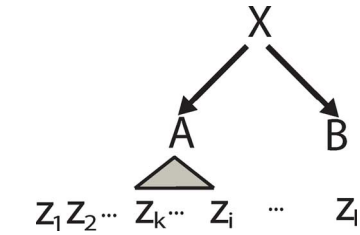


Fig. 6. A graphical example of an Earley state. The non-terminals are represented by the upper case letters. The lower case alphabets are the terminal symbols. The dot "·" is a marker representing the current position of the input string.

Bayesian Methods of Grammatical Inference: The inference of SCFG models is predominantly carried out using the inside-outside algorithm [3]. The inside-outside algorithm is very similar to the forward-backward algorithm used in the inference of hidden Markov models. In the case of SCFGs, the inside-outside algorithm restricts the grammar to have a particular form called the Chomsky normal form (CNF). However, any grammar can be transformed into the Chomsky normal form [15]. We begin by defining the inside probability for a stochastic grammar model. The inside probability is denoted by $\gamma(i, j, n) = \mathbf{P}\{n \stackrel{*}{\Rightarrow} z_i, \ldots, z_j\}$. This is described as the probability that non-terminal $n \in \mathcal{N}$ derives the observations (terminals) from $z_i, \ldots, z_j$. The inside-outside algorithm computes this probability recursively by first computing the inside probability of single observations $\gamma(i, i, n)\ \forall\ n, i$ and then incrementally computing the inside probability for all sequences of length two and so on.

In this paper, we choose to work with unconstrained grammars that are not in Chomsky normal form. We prefer this approach because it preserves the intuitive meaning of production rules. The Earley-Stolcke algorithm also computes inside probabilities but it uses a top-down approach which is different from the bottom-up approach that the inside-outside algorithm uses.

The Earley-Stolcke Parser: The Earley-Stolcke parser scans an input string $z = z_1, \ldots, z_t, \ldots, z_T$ from left to right and is able to compute the probability of the string $\mathbf{P}\{z \,|\, G_k\}$ given the parameters of the SCFG. As each symbol z_t is scanned, a set of states u_t is created which represents the condition of the inference process at that point in the scan. Each state in u_t represents (1) a production rule $a_m \in \mathcal{A}$ such that we are currently scanning a portion of the input string which is derived from its right hand side, (2) a point (marker) in that production rule which shows how much of that rule's right side we have recognized so far and (3) a pointer back to the position in the input string at which we began to look for that instance of the production rule. Each state is an incomplete portion of the parse tree which generated the input string x. These states are referred to as the control structure used by the Earley-Stolcke parser to store the incomplete parse trees and are represented as ${}_i^j X \rightarrow \lambda \cdot Y \mu[\alpha, \gamma]$.

The upper-case letters X and Y are non-terminals, λ and μ are substrings of non-terminals and terminals, "·" is the marker that specifies the end position j for the partially parsed input, i is the starting index of the substring that is generated by the non-terminal X. Fig. 6 illustrates an example state ${}_i^j X \rightarrow A \cdot B$, where A and B are non-terminals; the indices i and j specify the beginning and the end of the substring respectively which the non-terminal X can "explain" so far, and the index marker "·"

demarcates the part of X's production rule that has been applied to explain the substring. With the marker in front of the non-terminal B, B is not yet applied, and the state is still incomplete. Each state is also associated with a forward probability α and an inside probability γ which are explained in more detail later. For the purposes of dealing with the start symbol, the Earley-Stolcke uses a dummy state ${}_0^0S' \rightarrow \cdot S[1,1]$ which is the initial state of the Earley Stolcke parser.

Earley-Stolcke Operations: In general, we operate on a state set u_t as follows: the states in the set are processed in order, by performing one of three operations on each depending on the form of the state. These operations may add more states to u_t and may also put states in a new state set u_{t+1}. Whenever an operation attempts to add a new state, it is linked to an existing state. The *predictor* operation is applicable to a state when there is a non-terminal to the right of the dot. It causes the addition of one new state to u_t for each alternative production rule of that non-terminal. The dot is placed at the beginning of the production rule in each new state. The pointer is set to t, since the state was created in u_t. Thus the predictor adds to u_t all productions which might generate sub-strings beginning at x_{t+1}. More formally, for a state ${}_s^tX \rightarrow \lambda \cdot Y\mu$ in the state set u_t, the predictor adds a new state ${}_t^tY \rightarrow \cdot\nu$ for each of the alternative production rules $(Y \rightarrow \nu) \in \mathcal{A}$. A link is thus created between these states. The state ${}_t^tY \rightarrow \cdot\nu$ is called a *predicted* state.

The *scanning* operation, on the other hand, is applicable just in the case when there is a terminal to the right of the dot. The scanner compares that symbol with z_{t+1}, and if they match, it adds the state to $u(t+1)$, with the dot moved over one symbol in the state to indicate that that terminal symbol has been scanned. If ${}_s^tX \rightarrow \lambda \cdot a\mu$ exists and $z_{t+1} = a$, the scanning operation adds a new state ${}_s^{t+1}X \rightarrow \lambda a \cdot \mu$ to state set $u(t+1)$ which is called a *scanned* state. A link is also created between these states.

The third operation, the *completer*, is applicable to a state if its dot is at the end $({}_s^tX \rightarrow \lambda Y\mu\cdot)$ of its production. Such a state is called a "complete" state. For every complete state, the completer goes back to the state set s indicated by the pointer in the complete state, and adds all states from $u(s)$ to $u(t)$ which have X (the non-terminal corresponding to that production) to the right of the dot. It moves the dot over X in these states. Intuitively, $u(t)$ is the state set we were in when we went looking for that X. We have now found it, so we go back to all the states in $u(s)$ which caused us to look for a X, and we move the dot over the X in these states to show that it has been successfully scanned. A completion operation adds a new state ${}_s^tX \rightarrow \lambda Y \cdot \mu$ (called a completed state) using ${}_s^rX \rightarrow \lambda \cdot Y\mu$ and ${}_r^tY \rightarrow \nu\cdot$. A link pointing from ${}_r^tY \rightarrow \nu\cdot$ to ${}_s^tX \rightarrow \lambda Y \cdot \mu$ is also created. In such a manner, the Earley-Stoclke parser continues until all the observation symbols have been scanned. If the final state set u_T contains the state ${}_0^0S' \rightarrow S\cdot$, then the algorithm terminates successfully. It represents a successful parse of the sentence $z_1, \ldots, z_T$.

Earley-Stolcke Probabilities: We mentioned earlier that each state is associated with a forward probability $\alpha({}_i^jX \rightarrow \lambda \cdot \mu)$ which is the sum of the probabilities of all paths of length i which end in the state ${}_i^jX \rightarrow \lambda \cdot \mu$ and generate observations $z_1, \ldots, z_i$. The inner probability $\gamma({}_i^jX \rightarrow \lambda \cdot \mu)$ of a state is defined as the sum of the probability of all paths of length $k-i$ which pass through the state ${}_i^jX \rightarrow \lambda\cdot\mu$ and hence derive the observations $z_i, \ldots, z_{k-1}$. A path is simply a sequence of Earley states linked through the operations of prediction, scanning and completion as mentioned above. The length of a path is defined as the number of scanned states in it. Since a state can be scanned only when the terminal to the right of the dot matches the input symbol, there is a one-to-one correspondence between scanned states and length of the input. In Fig. 8, we show the concept on an Earley path and the manner in which they can split and merge via the operations of prediction, scanning and completion. The updates of the forward probability α and inside probability γ for each of the state operations is summarized in Fig. 7.

Prediction

$${}_s^tX \rightarrow \lambda \cdot Y\mu \ [\alpha,\gamma] \Rightarrow {}_t^tY \rightarrow \cdot\nu \ [\alpha',\gamma']$$
$$\alpha' += \alpha\mathbf{P}\{Y \rightarrow \nu\} \text{ (a sum over all duplicate states)},$$
$$\gamma' = \mathbf{P}\{Y \rightarrow \nu\}.$$

(a)

Scanning

$${}_s^tX \rightarrow \lambda \cdot a\mu \ [\alpha,\gamma] \Rightarrow {}_s^{t+1}X \rightarrow \lambda a \cdot \mu \ [\alpha',\gamma']$$
$$\alpha' = \alpha\mathbf{P}\{a\} \ ,$$
$$\gamma' = \alpha\mathbf{P}\{a\}.$$

(b)

Completion

$$\left.\begin{array}{l} {}_r^tY \rightarrow \nu\cdot \ [\alpha'',\gamma''] \\ {}_s^rX \rightarrow \lambda \cdot Y\mu \ [\alpha,\gamma] \end{array}\right\} \Rightarrow {}_s^tX \rightarrow \lambda Y \cdot \mu \ [\alpha',\gamma']$$
$$\alpha' += \alpha\gamma'' \text{ (a sum over all duplicate states)},$$
$$\gamma' += \gamma\gamma'' \text{ (a sum over all duplicate states)}.$$

(c)

Fig. 7. The updates for the forward probability α and the inside probability γ for each of the Earley-Stolcke operations.

IV. Trajectory Modeling and Inference Using Reciprocal Stochastic Processes

In this section, we describe a 1 dimensional Markov random field called a reciprocal stochastic process. These processes are used in this paper to model target trajectories with an intended destination. On the time scales used in meta-level tracking, most real world targets are "destination-aware"—they have a well defined destination, and they rarely move according to a "drunkard's" random walk (Markov chain). From a probabilistic modeling point of view, being destination-aware means that the initial and final target states (in terms of position) need to be chosen from a joint distribution before specifying the transition law of the target dynamics. Naturally, finite state Markov chains cannot capture this long range dependency.

The modeling of trajectories using RPs is significant because it offers an alternative to the SCFG based modeling described in Section III. While an SCFG is able to constrain the expected destination of a target, the use of RP models and position tracklets allows us to exactly constrain the final destination of the target. Such a capability is of obvious importance in accurate destination prediction.

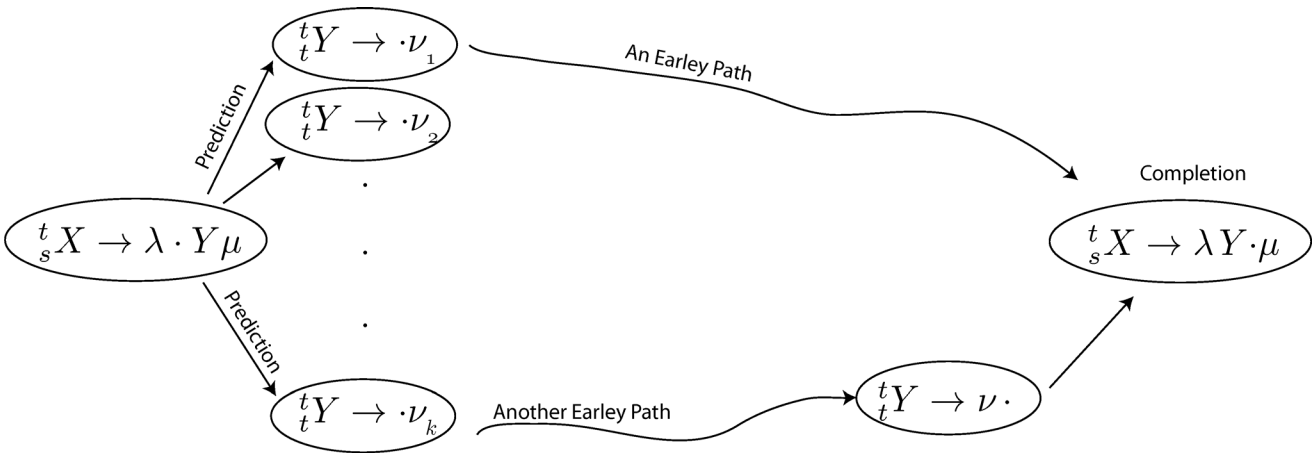


Fig. 8. This figure shows the links created due to the prediction, scanning and completion operations between states in an Earley-Stolcke parser. The prediction creates visible splits in the path while scanning and completion merge different paths. Some paths created by the prediction operation do not lead to a viable string and hence remain incomplete.

A. Reciprocal Process Models for Destination-Constrained Trajectories

Review of RP's: Consider a random process X_t indexed on the set $t \in \{0, \ldots, T\}$ for some integer $T \geq 2$. We assume that the process takes values in some set $\mathcal{N}$ with finite cardinality $N \geq 1$. The state space $\mathcal{N}$ is the set of integers $\mathcal{N} = \{1, \ldots, N\}$ (a discrete-state process). A stochastic process is defined to be reciprocal in nature if

$$\mathbf{P}\{X_t \,|\, X_s, s \neq t\} = \mathbf{P}\{X_t \,|\, X_{t-1}, X_{t+1}\}, \qquad (14)$$

for each $t = 2, \ldots, T-1$. As a result, X_t is conditionally independent of its value at all time points $X_0, \ldots, X_{t-2}$, $X_{t+2}, \ldots, X_T$ given its neighbors X_{t-1} and X_{t+1}. The RP model is specified by its 3-point transition functions in (14) along with a given joint distribution $\Pi = \mathbf{P}\{X_0, X_T\}$ on the end-points of the process. A reciprocal process is a one-dimensional version of a Markov random field. Our interest in reciprocal stochastic processes stems from the fact that a trajectory with a known destination can be represented as a pinned reciprocal process in which the end point X_T is fixed. This is shown in [17] to generate a Markov bridge.

A reciprocal process can be considered to be composed of N Markov bridges, each Markov bridge corresponding to the end-point X_T being fixed at one of the $n \in \mathcal{N}$ states. The model for each such Markov bridge is encapsulated by [14]

$$\mathbf{P}\{X_t \,|\, X_{t-1}, X_T\} = \frac{\mathbf{P}\{X_t \,|\, X_{t-1}, X_{t+1}\}}{\mathbf{P}\{X_{t+1} \,|\, X_t, X_T\}} \times \mathbf{P}\{X_{t+1} \,|\, X_{t-1}, X_T\}. \qquad (15)$$

The 3-pt probability transitions $\mathbf{P}\{X_t = j \,|\, X_{t-1} = i, X_{t+1} = l\}$ can be denoted as $Q_{i,j,l}(t)$. We assume that the 3-pt transitions are time invariant and hence $Q_{i,j,l}(t) = Q_{i,j,l}$. We can then fully specify the set of N Markov bridges using (15) by the backward recursion for $t = T-2, T-3, \ldots, 1, 0$,

$$B^n_{i,j}(t) = \mathbf{P}\{X_{t+1} = j \,|\, X_t = i, X_T = n\} = \frac{Q_{i,j,l}}{B^n_{j,l}(t+1)} \left(\sum_{r=1}^{N} \frac{Q_{i,r,l}}{B^n_{r,l}(t+1)} \right)^{-1}. \qquad (16)$$

The last term on the right-hand side of (16) is the normalization constant. We can initialize $B^n_{i,j}(T-1) = 1$ if $j = n$ and 0 otherwise.

Construction of Markov Bridges: A random process X^n_t can then be constructed by conventional Markov process construction if the initial state X^n_0 is drawn from $\mathbf{P}\{X_0\}$ and propagated forward using the transition probabilities $B^n(t)$. Such a process is the required Markov bridge pinned at the end-point with $X^n_T = n$ with probability 1. An RP X_t can be constructed by drawing $(X_0 = i, X_T = n)$ from the specified RP end-point distribution and constructing the Markov bridge starting at $X_0 = i$ and using the transition probabilities $B^n(t)$ defined in (16).

Hidden Reciprocal Models (HRM): The notion of a hidden reciprocal model (HRM) is introduced by defining an observation process Z_t with the conditional independence property that $\mathbf{P}\{Z_0, \ldots, Z_T \,|\, X_0, \ldots, X_T\} = \prod_{t=1}^{T} \mathbf{P}\{Z_t \,|\, X_t\}$. The process Z_t is called a hidden reciprocal model (HRM) with $C_i(t) = \mathbf{P}\{Z_t \,|\, X_t = i\}$, denoting the observation probabilities. Each Markov bridge is defined to be a model parameterized by $\theta^n = (\Pi, B^n_{i,j}(t), C_i(t))$, where $\theta^n \in \Theta, n = 1, \ldots, N$.

The representation of reciprocal processes using the concept of Markov bridges allows a convenient characterization for fixing the end-point of a target's trajectory. This permits consideration of "destination-constrained" trajectories which can be imparted with the intent of the target. The starting and ending point of the trajectory is thus constrained by a Markov bridge model $\theta_n \in \Theta$. Each such trajectory with different ending points can be represented using a Markov bridge model. If we consider the intent of the target as deciding which of the n regions the target will end up in, the intent inference task amounts to classifying the observed trajectory of the target to one of the N models in Θ.

B. Signal Processing of RP Models

In this section, we present equations for evaluating the log likelihood $L_t(\theta) = \log(\mathbf{P}\{Z_0, \ldots, Z_t\})$ of an observed sequence $Z_0, \ldots, Z_t$ from an RP model $\theta \in \Theta$. Detailed derivations for the associated filtering and maximum likelihood state

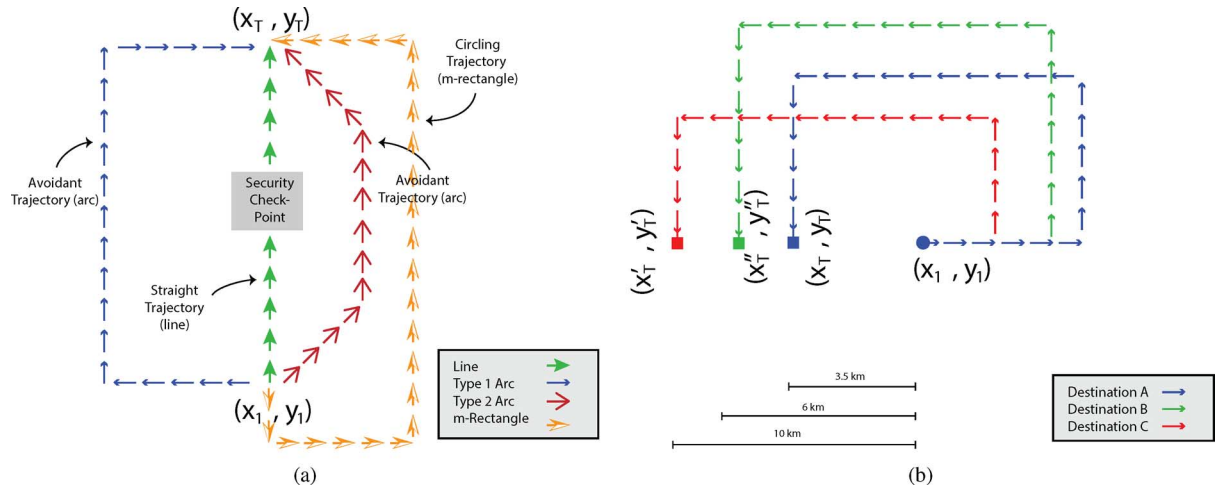


Fig. 9. (a) Depiction of different anamolous trajectory shapes and associated intent. (b) Targets traveling to different destination using the same trajectory shape.

estimation of RP models can be found in [14]. We define the quantity

$$\alpha_i^n(t) = \mathbf{P}\{X_t = i, Z_0, \ldots, Z_t \,|\, X_T = n\} \tag{17}$$

with initialization

$$\alpha_i^n(0) = \mathbf{P}\{X_0 = i, Z_0 \,|\, X_T = n\} = \frac{\pi_i^n C_i(0)}{\gamma^n(0)}. \tag{18}$$

The first term in (18) comes from the RP end-points distribution such that

$$\pi_i^n = \mathbf{P}\{X_0 = i \,|\, X_T = n\} = \frac{\Pi_{i,n}}{\sum_{j=1}^N \Pi_{j,n}}, \tag{19}$$

and

$$\gamma^n(0) = \sum_{i=1}^N \pi_i^n C_i(0) = \mathbf{P}\{Z_0 \,|\, X_T = n\}. \tag{20}$$

We also define $L_0(\theta) = \log(\mathbf{P}\{Z_0\}) = \sum_{n=1}^N \gamma^n(0)\mathbf{P}\{X_T = n\}$. The filtering equation can be computed recursively

$$\begin{aligned}\alpha_i^n(t) &= \sum_{j=1}^N \mathbf{P}\{X_t = i, X_{t-1} = j, Z_0, \ldots, Z_t \,|\, X_T = n\} \\ &= \frac{C_i(t)\sum_{j=1}^N B_{j,i}^n(t-1)\alpha_j^n(t-1)}{\gamma^n(t)}\end{aligned} \tag{21}$$

for $t = 0, \ldots, T-2$. We have applied the property that the RP pinned at $X_T = n$ is a Markov bridge with initial state distribution π_i^n and transition probabilities $B_{i,j}^n(t)$. The normalization factor $\gamma^n(t)$ is given by

$$\gamma^n(t) = \sum_{i=1}^N C_i(t) \sum_{j=1}^N B_{j,i}^n(t-1)\alpha_j^n(t-1). \tag{22}$$

The log-likelihood L_t can then be recursively computed to be

$$L_t(\theta) = L_{t-1}(\theta) + \log\left(\sum_{i=1}^N \gamma^n(t)\,\mathbf{P}\{X_T = n\}\right).$$

For maximum likelihood classification with partial observations $Z_0, \ldots, Z_t$, the hidden Markov bridge model with the maximum likelihood is classified as the destination of the target using

$$\theta^\star = \arg\max_{\theta\in\Theta} L_t(\theta). \tag{23}$$

V. Numerical Examples

In this section, numerical studies are presented to illustrate SCFG and RP models in anomalous trajectory detection and classification. The first set of experiments in Section V-A demonstrates individual examples for the classification of anomalous trajectories based on SCFG shape models. The scenario being considered is depicted in Fig. 9(a).

The second set of experiments in Section V.B deals with the use of the SCFG and RP models in both shape and destination-constrained trajectories for pattern of life analysis. We contrast the SCFG representation with that of hidden reciprocal models. In each simulation, a base tracker is used to extract tracklets which are then input into the appropriate classifier. The SCFG filter refers to the Bayesian signal processing algorithm described in Section III-D. An extended Kalman filter (EKF) is employed for tracking the target using a constant acceleration model for the target dynamics in all the subsequent simulations.

A. Classification of Trajectory Shape

In this section, the results of detecting different anomalous trajectories based on shape are presented. Consider the scenario depicted below in Fig. 9(a), where the target moves in a specific trajectory around a sensitive asset (like an embassy or a security check-point). The shape of the trajectory correlates with the intent of the target and is of great importance if it can be reliably detected. The two shapes considered here are an m-rectangle trajectory which depicts circling behavior and an arc trajectory which can be used to signify avoidant behavior. A clean and estimated version of the m-rectangle trajectory traversed by a target is shown in Fig. 10(a) while that of an arc trajectory is shown in Fig. 10(b).

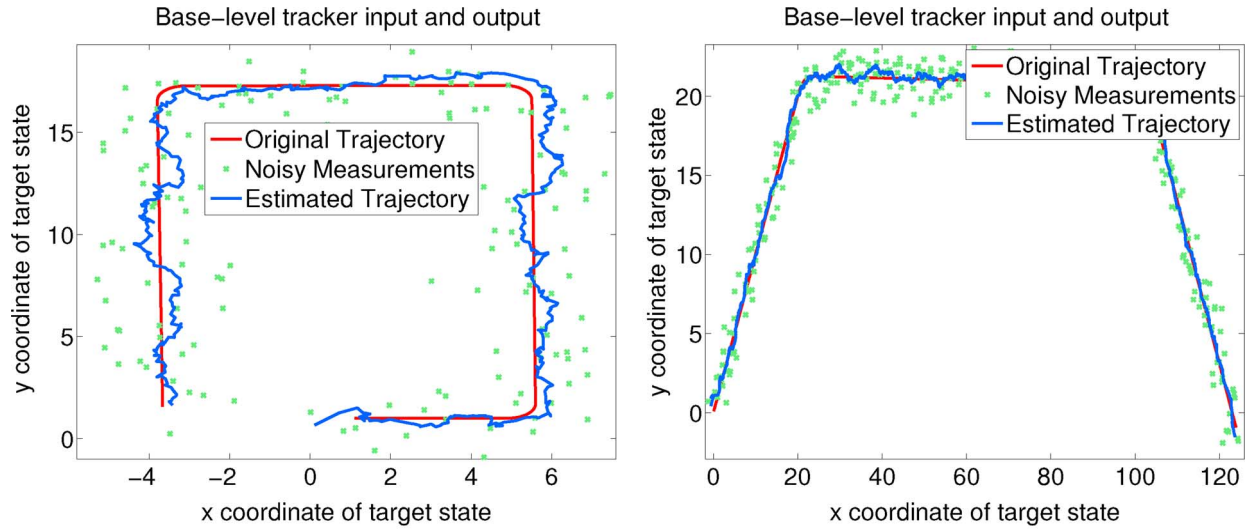


Fig. 10. (a) An m-rectangle trajectory, its noisy measurements and estimated output from the base level tracker. (b) An arc trajectory, its noisy measurements and estimated output from the base level tracker.

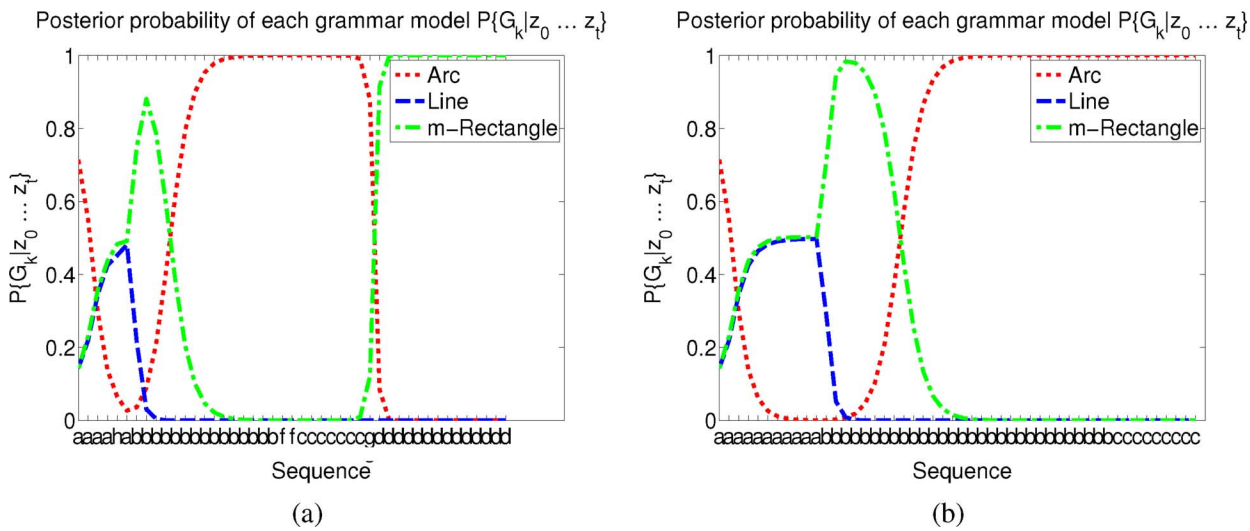


Fig. 11. The posterior probability of different shapes when tracking a target following a (a) m-rectangle trajectory and an (b) arc trajectory.

The estimated state of the target output by the base-level tracker is used to estimate tracklets that form the input to the SCFG filter. The quantization of the velocity tracklets has a de-noising effect on the state estimates. As a result, even bad base-level trackers can be used to obtain relatively clean strings of the target trajectory. The quantization noise can be captured using the probability mass function

$$\mathbf{P}\{z_t = q_j \,|\, z_t = q_i\} = \begin{cases} 0.7 & \text{if } i = j, \\ 0.1 & \text{if } |i-j| = \frac{\pi}{4}, \\ 0.05 & \text{if } |i-j| = \frac{\pi}{2}, \\ 0 & \text{otherwise} \end{cases} \tag{24}$$

For the m-rectangle trajectory in Fig. 10, the grammar model probabilities are computed using Bayes' rule and the assumption of a uniform prior over all grammar models. The posterior probability distribution is shown in Fig. 11(a). We observe that the line grammar initially has the highest posterior probability because the target has not yet made a turn. As soon as the first turn is made, the posterior probability of the line vanishes. After the third turn is made, the m-rectangle trajectory becomes the more likely grammar model.

B. *Destination-Constrained Trajectories*

In this section, the usefulness of SCFG grammars with constrained destinations is examined by a proof-of-concept example. Consider the target trajectories shown in Fig. 9(b) which are of a m-rectangle trajectory to different destinations. Using the constraints described in Section III.C, a trajectory can be forced to end at a certain point (expected destination). The rule probabilities of a template m-rectangle grammar are solved for using these constraints and three different grammar models are produced. The trajectory in Fig. 10(a) is the same as the one having destination A in Fig. 9(b). The posterior probability of the grammar models is computed using the trajectory with destination A as the input and the result is shown in Fig. 12(a). For this example, we notice that after the 3rd turn, the correct destination is chosen as the model with

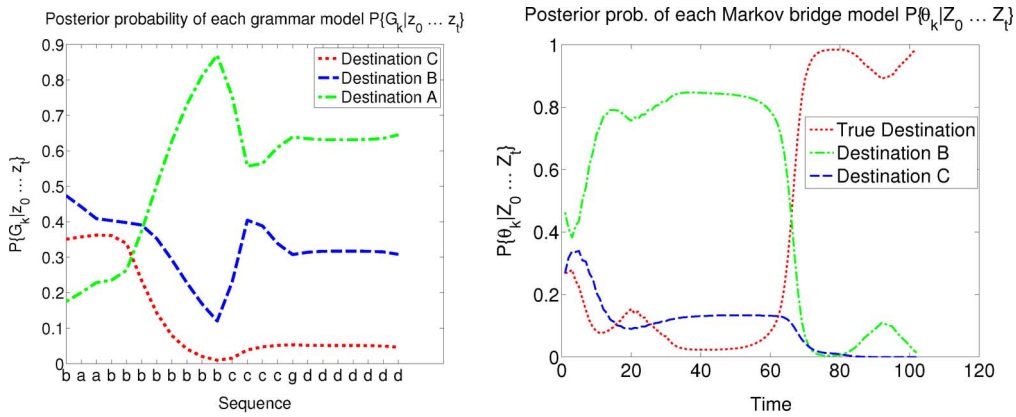


Fig. 12. (a) The posterior probability of different destinations using an m-rectangle trajectory as depicted in Fig. 9(b). (b) Markov bridge model classification for destination prediction.

highest posterior probability. This is because for an m-rectangle trajectory, displacement only occurs in one dimension. Since two opposite sides of an m-rectangle should be of the same size, only one of the remaining sides needs to be observed, before the model can predict the destination.

Using the same set of 3 destinations depicted in Fig. 9(b), a set of Markov bridge models can be constructed using the development in Section IV-A. The trajectory with destination A is then used as input to classify the destination based on the signal processing of RP models. The results from (17)–(23) are used to obtain the posterior probabilities (where each Markov bridge represents one of the destinations). The posterior probabilities can be seen in Fig. 12(b). We immediately notice that the RP formulation is unable to distinguish between the models until nearly the end of the trajectory. However, at a certain point in the trajectory, the posterior probability of the correct destination increases until it reaches its destination with probability 1.

C. Pattern of Life Analysis

In this section, we compare the performance of a destination-constrained SCFG (as presented in Section III-C with the fusion of an SCFG shape model and an RP destination model. SCFGs can be used to effectively determine complex spatial patterns of a trajectory. However, the destination can only be predicted in an expected sense. Alternatively, reciprocal processes cannot incorporate shape constraints on trajectories but are well suited towards predicting the destination of a target. If the intent of target is modeled as a function of its shape and destination, then the conditional joint probability of shape and destination achieves a form of inter-model fusion.

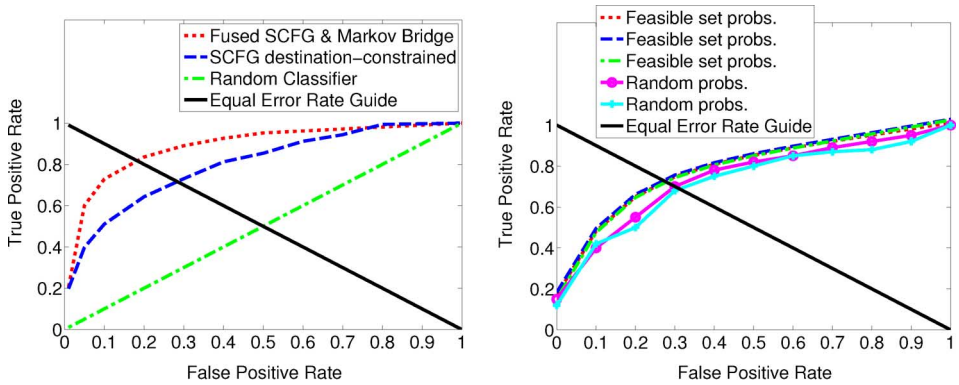
Suppose that there are K syntactic trajectory patterns of interest (like arcs, rectangles, closed loops, move-stop-move etc.) and N destinations of interest in an intent inference task. Each trajectory shape of interest is modeled through the use of a SCFG $G_k, k = \{1, \ldots, K\}$ and each destination of interest is modeled by a Markov bridge $\theta_n, n = \{1, \ldots, N\}$. The set of all target intents $\mathcal{I} = \{I_m \,|\, I_m \in G_k \times \theta_n, m = \{1, \ldots, KN\}\}$ under consideration is given by the Cartesian product of the SCFG and Markov bridge models. Under the independence assumption of target shape and target destination, the posterior probability of the target intent is given by

$$\mathbf{P}\{I_m \,|\, z_{1:t}\} = \mathbf{P}\{G_k \,|\, z_{1:t}\}\mathbf{P}\{\theta_n \,|\, z_{1:t}\} \tag{25}$$

where $z_{1:t} = \{z_1, \ldots, z_t\}$ are understood to be velocity tracklets when dealing with SCFG models and position tracklets when dealing with Markov bridge models. The individual posterior probabilities of the SCFG model G_k and Markov bridge model θ_n can be computed using the filter equations presented in Sections III-D and IV-B respectively.

Using the joint shape and destination classifier of (25), our aim is to detect a significant departure of a target or targets from a pre-established routine trajectory. This involves a trajectory either deviating in shape between known starting point and destination point or a target deviating in both shape and destination point. The example demonstrated in Fig. 1 is used for the numerical study.

The flavor of the experiment is as follows: Consider that targets traveling between point A and point B are tracked and its trajectory on the X-Y plane is recorded. A normal trajectory has a linear shape between point A and point B. A sensitive asset is located between points A and B. A suspicious trajectory, on the other hand, involves any kind of circling or avoidant behavior between points A and B. For instance, we would like to classify m-rectangle shaped trajectories with point B as its destination. In the experiment, we create 100 differently shaped trajectories between point A and point B. Of these, 25 trajectories are arc-shaped, 50 are m-rectangles and 25 are arbitrarily shaped. The m-rectangle trajectories form positive examples while the other shapes represent negative examples. We compare two classifiers, viz., the destination constrained SCFG and a fusion of the RP model (for destination prediction) and the SCFG model (for shape classification). The receiver operating characteristics for these classifiers are shown in Fig. 13(a). The fused SCFG and RP classifier outperforms the destination constrained SCFG classifier. On the equal error rate guide, the classification accuracy is 83% for the fused SCFG and RP model while it is 73% for the constrained SCFG model. The area under



(a) (b)

Fig. 13. (a) The receiver operating characteristics of the fused SCFG and RP model classifier versus a destination constrained SCFG. (b) The ROC curve for only SCFG based shape classification with and without destination constraints.

the curve (AUC) for the fused SCFG and RP model classifier is 0.8879 while the AUC for the destination constrained SCFG is 0.8012. We feel that the expected destination constraint makes the constrained SCFG a weak classifier for destination prediction. It would be prudent to point out that constraining higher order moments of the sentence length may lead to better classification accuracy. For example, the variance of the sentence length could place a more stringent constraint on the destination through creating a narrow distribution around the expected destination.

As demonstrated in Fig. 13(a), the expected destination constraint does the not make the constrained SCFG very amenable towards destination prediction. However, it does make the SCFG more well-behaved in terms of its performance when only considering shape classification. In Fig. 13(b), an SCFG model is only used for shape classification of the models in Section III-B. We see one distinct cluster of similar ROC curves in Fig. 13(b). These curves are obtained for SCFG models recovered by using feasible solutions generated from (11). The other curves are obtained by randomly choosing probabilities for the m-rectangle grammar model that do not satisfy any destination constraint. The classification accuracy is an average of 73% in the former case while it is lowered to 69% in the latter.

VI. Conclusion

This paper has presented stochastic context free grammars (SCFG) and reciprocal Markov process (RP) models for the trajectories of targets. Our premise is that the target's trajectory can be modeled as words (modes) spoken by a SCFG language or RP model. Then, Bayesian signal processing algorithms are proposed to detect anomalous trajectories. The methods developed in this paper can be viewed as *middleware forming the human-sensor interface* since they interpret information from a tracker to assist a human operator. The parsing of the motion trajectories is implemented with Earley-Stolcke parsing algorithm. In [9], we have fed back this syntactic information to improve the performance of the tracker and demonstrate the performance on GMTI data collected with DRDC Ottawa's XWEAR radar. In related work, [10] presents data fusion algorithms for SCFGs in a camera-network for surveillance. The main thrust in this paper has been to incorporate the destination of a target as part of its intent (in addition to the spatial pattern exhibited by the trajectory).

The simulations present proof-of-concepts for the various applications presented. The use of velocity tracklets allows a parsimonious representation of both shape and destination of the target trajectory. RP models can only be used to represent destination-constrained trajectories. They also require discretization of the surveillance space into a grid which leads to a large state-space. RP models are also disadvantageous because they require knowledge of the time T at which the destination is reached. However, SCFGs can only constrain the trajectory destination weakly. A fusion of the the SCFG and RP models provides a better classification of joint shape and destination-based target intent.

Acknowledgment

The authors would like to thank Dr. Langford White, School of Electrical and Electronics Engineering at The University of Adelaide for his valuable discussion and development of discrete-state hidden reciprocal models for destination-constrained trajectories.

References

[1] Y. Bar-Shalom and X. Li, *Estimation and Tracking: Principles, Techniques and Software*. Boston, MA: Artech House, 1993.

[2] K. S. Fu, *Syntactic Pattern Recognition and Applications*. Englewood Cliffs, NJ, USA: Prentice-Hall, 1982.

[3] K. Lari and S. J. Young, "The estimation of stochastic context-free grammars using the inside-outside algorithm," *Comput. Speech Lang.*, vol. 4, no. 1, pp. 35–56, 1990.

[4] M. I. Miller and J. A. O. Sullivan, "Entropies and combinatorics of random branching processes and context-free languages," *IEEE Trans. Inf. Theory*, vol. 38, no. 4, pp. 1292–1310, Jul. 1992.

[5] C. Piciarelli, C. Micheloni, and G. Foresti, "Trajectory-based anomalous event detection," *IEEE Trans. Circuits Syst. for Video Technol.*, vol. 18, no. 11, pp. 1544–1554, Nov. 2008.

[6] J. Muncaster and Y. Ma, "Activity recognition using dynamic Bayesian networks with automatic state selection," in *Proc. IEEE Workshop Motion and Video Comput., WMVC'07*, Feb. 2007, pp. 30–34.

[7] M. S. Ryoo and J. K. Aggarwal, "Recognition of composite human activities through context-free grammar based representation," in *Proc. CVPR'06: IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recogn.*, Washington, DC, 2006, pp. 1709–1718, IEEE Computer Society.

[8] A. Bobick and Y. Ivanov, "Action recognition using probabilistic parsing," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recogn.*, Jun. 1998, pp. 196–202.

[9] A. Wang, V. Krishnamurthy, and B. Balaji, "Intent inference and syntactic tracking with GMTI measurements," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 47, no. 4, pp. 2824–2843, Oct. 2011.
[10] V. Krishnamurthy and M. Fanaswala, "Intent inference via syntactic tracking," *Digital Signal Process.*, vol. 21, no. 5, pp. 648–659, 2011.
[11] S.-W. Joo and R. Chellappa, "Attribute grammar-based event recognition and anomaly detection," in *Proc. 2006 Conf. Comput. Vis. Pattern Recogn. Workshop*, Washington, DC, 2006, pp. 107–115, ser. CVPRW'06, IEEE Computer Society.
[12] A. Stolcke, "An efficient probabilistic context-free parsing algorithm that computes prefix probabilities," *Comput. Linguist.*, vol. 21, no. 2, pp. 165–201, 1995.
[13] D. A. Castanon, B. C. Levy, and A. S. Willsky, "Algorithms for the incorporation of predictive information in surveillance theory," *Int. J. Syst. Sci.*, vol. 16, pp. 367–382, 1985.
[14] L. B. White and F. Carravetta, "Optimal smoothing for finite state hidden reciprocal processes," *IEEE Trans. Autom. Control*, vol. 56, no. 9, pp. 2156–2161, Sep. 2011.
[15] A. V. Aho and J. D. Ullman*, The Theory of Parsing, Translation, and Compiling*. Upper Saddle River, NJ: Prentice-Hall, 1972.
[16] C. S. Wetherell, "Probabilistic languages: A review and some open questions," *ACM Comput. Surv.*, vol. 12, pp. 361–379, Dec. 1980.
[17] B. Jamison, "Reciprocal processes: The stationary Gaussian case," *Ann. Math. Statist.*, vol. 41, pp. 1624–1630, 1970.

Mustafa Fanaswala (S'12) received his bachelor's degree in electrical and electronics engineering from the American University of Sharjah, UAE in 2007, and M.A.Sc. in electrical engineering from Carleton University, Ottawa, in 2009. He is currently pursuing a Ph.D. at the Department of Electrical Engineering, University of British Columbia, Vancouver, Canada. His current research is focused on modeling long-range interactions in time-series with applications in target tracking and intent inference.

Vikram Krishnamurthy (F'05) is currently a professor and holds the Canada Research Chair at the Department of Electrical Engineering, University of British Columbia, Vancouver, Canada. His current research interests include computational game theory, stochastic dynamical systems for modeling of biological ion channels and stochastic optimization and scheduling. Dr. Krishnamurthy has served as Distinguished lecturer for the IEEE signal processing society and on the editorial board of several IEEE journals.