# Spatiotemporal Trajectory Models for Metalevel Target Tracking

*Mustafa Fanaswala, Vikram Krishnamurthy*
**University of British Columbia**
**Vancouver, BC, Canada**

## INTRODUCTION

Physical sensor–based target tracking is a classical problem that has been studied in great detail [1], [2]. This article presents metalevel tracking middleware algorithms to help human radar operators interpret tracks in order to detect and visualize suspicious spatiotemporal target trajectories. While state space models are ideal for target tracking, the main idea in this article is that stochastic context-free grammar (SCFG) models are also useful for modeling and interpreting trajectories.

In our previous articles [3], [4], several SCFG models were presented for specific target trajectories signifying malicious intent. The current article generalizes these trajectory models and extends them toward novel multitarget anomalous patterns. More specifically, eight trajectory models are presented: (a) random walk, (b) reciprocal process, (c) linear, (d) arclike, (e) rectangular, (f) destination-aware, (g) palindromic, and (h) target rendezvous trajectories. Our modeling framework focuses on the human–sensor interface (middleware) tasked with high-level reasoning and visualization from lower-level sensor measurements. Bayesian signal processing algorithms are also developed to perform model classification and change detection using novel SCFG models.

## EXAMPLE

Consider the scenario in Figure 1, where a target executing a trapezoidal trajectory (dashed blue line) is observed in noise. This trajectory could signify a target avoiding an obstacle by deviating away, passing the obstacle, and then returning to its previous path. Given noisy point measurements, how can one devise algorithms to detect whether the target executed such a pattern? This is a nontrivial estimation problem (it cannot be solved efficiently with template matching techniques), because exponentially many–scaled versions of the shape need to be considered.[1] SCFG models offer an efficient framework to model common shapes (Figure 2a) in a scale-invariant manner. The trajectory classification and modeling approach proposed in this article can also be viewed as a visualization layer that is able to assist the human analyst by extracting suspicious spatiotemporal patterns embedded in noisy tracks, as depicted in Figure 1. Efficient polynomial-time algorithms exist to perform classification using SCFG models, which are used in the section on numerical examples to demonstrate the effectiveness of SCFG models over competing hidden Markov models (HMMs).

## MOTIVATION: METALEVEL TRACKING AND INTENT INFERENCE

Classical target tracking assumes a Markovian state space model for the target kinematics. Such models are useful on short timescales (on the order of several seconds), and many well-known target tracking algorithms based on such Markov models have been developed in the literature [1]. This article is motivated by metalevel target tracking applications on longer timescales (on the order of several minutes). In metalevel tracking, one is interested in devising automated procedures that assist a human analyst to interpret the tracks obtained from a conventional tracking algorithm. On such longer timescales, most real-world targets are driven by a premeditated intent. The intent of a target can manifest in the shape of the target trajectory or its final destination (among other attributes). In this article, trajectory shape is modeled using SCFG models that are ideally suited to model patterns due to scale invariance, which emerges from their self-embedding properties. Moreover, time-varying SCFGs are used in this article to reflect the intent of the target

This research was supported by a Natural Sciences and Engineering Research Council strategic research grant. A preliminary version of this paper was presented at the 2014 International Conference on Information Fusion.
Authors' current address: M. Fanaswala, V. Krishnamurthy, University of British Columbia, Electrical and Computer Engineering, 2332 Main Mall, Main Street, Vancouver, BCV6T 1Z4 Canada, E-mail: mustafaf@ece.ubc.ca.
Manuscript was received April 3, 2014, revised July 27, 2014, and ready for publication August 21, 2014.
DOI. No. 10.1109/MAES.2014.140062.
Review handled by P. Willett.
0885/8985/14/ $26.00 © 2015 IEEE

[1] As an equivalent example, consider a string *al bm cm dn*, where *l*, *m*, and *n* are unknown positive integers such that $l + 2m + n = k$. How can one extract the arc trajectory bm cm from a noisy version of the string? If *b* and *c* were composite alphabets that were the union of *r* other symbols, template matching would require listing an exponential number $\sum_{m=0}^{k/2}[k-(2m-1)]r^{m+1}$ of possibilities for all possible choices of *l*, *m*, and *n*, since the values of these integers are not known.

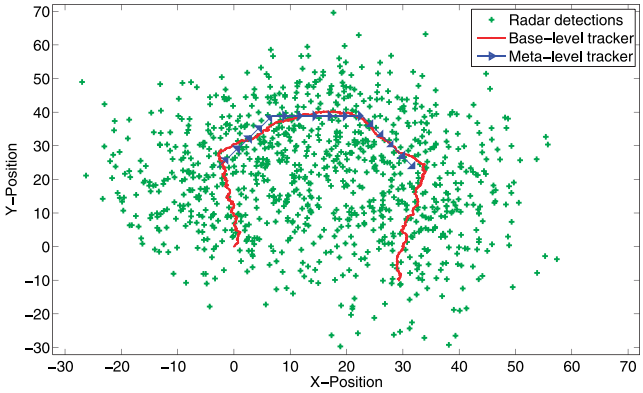


**Figure 1.**
A radar makes noisy observations (green dots) that can be tracked by a conventional tracker, as shown by the solid red line. However, useful information like shape and destination is often obscured by the estimation process. The main aim of this article is to detect and track higher-level characteristics of a trajectory like the movement patterns shown as blue arrows. In this example, a target executes an arc movement (the open trapezoidal shape) that can be associated with various anomalous behaviors. Our two-scale approach uses the base-level tracker output to track intent-driven trajectories with long-range dependencies.

to move toward its destination. Such a characterization emerges from the observation that local target dynamics at the metalevel may be anticipative (destination aware) or noncausal to terminate a trajectory in a known destination. Our previous article [3] utilized reciprocal stochastic models for such destination-aware trajectories, which are generalized in this article using time-varying SCFGs. The use of SCFGs in metalevel tasks is mainly motivated by their ability to capture arbitrary-range (as opposed to fixed-length) long-term dependencies in the target trajectory [5]. Consequently, metalevel analysis of target trajectories can serve as a visualization tool for suspicious trajectories and anomalous target behavior.

## LITERATURE SURVEY

The classification and tracking of anomalous spatiotemporal trajectories arise in many application areas, such as target tracking using radars [3], gesture recognition using optical [6] and time-of-flight [7] sensors, human action recognition in camera networks [8], gait analysis [9], network packet traces [10], and vehicular geoposition coordinates [11]. The maritime surveillance literature has also recently seen an interest in trajectory anomaly detection applications [12], [13]. In particular, [13] examined the target rendezvous problem. A common approach in such sequential pattern recognition problems is the use of HMMs to capture the temporal dependence of observations. Such an approach was taken in [14] by using flow vectors on objects being tracked in video sequences. Spatial nodes of interest are first isolated using a Gaussian mixture modeling technique. Routes are then created by clustering different trajectories, and a high-level HMM is learned for each route. However, such a model cannot incorporate destination-specific information. Moreover, long-term dependencies in the trajectory are lost due to the Markov assumption.

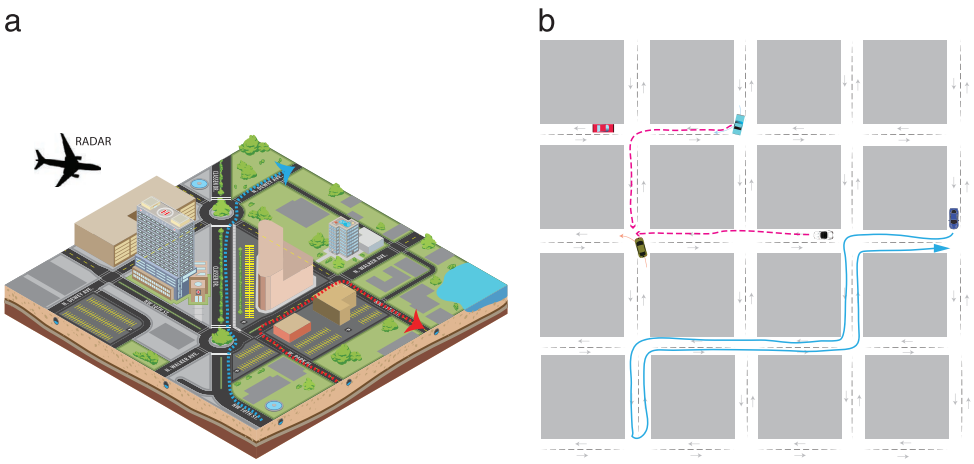


**Figure 2.**
(a) A conceptual sketch of a citywide radar surveillance application. Target trajectories evolve over roads and are constrained to certain shapes due to the geometry imposed by urban environments. The trajectory in red depicts an m-rectangle, while that in blue represents an arc. (b) The road network in a city can be treated as a directed graph with nodes representing intersections and edges representing connecting streets. The trajectory in blue shows a palindrome path representing a target retracing its path. Such paths can be considered anomalous in many settings. The red trajectory depicts rendezvousing targets, which is of great interest in many crime-related events. Such targets are called "destination aware" because the evolution of their trajectory is influenced by the destination endpoint.

Our article is related to the approach taken in [15] and [16]. A nonprobabilistic context-free grammar approach was used in [15] to identify two-person interactions like hugs, handshakes, kicks, and punches to enforce syntactic structure on detected events. In [16], SCFGs were used to recognize cheating actions in card games at casinos. Our article departs from them significantly as we consider trajectory modeling in a tracking situation and not an action recognition system. This article generalizes the treatment in [3] and presents novel models for multitarget anomalous trajectories.

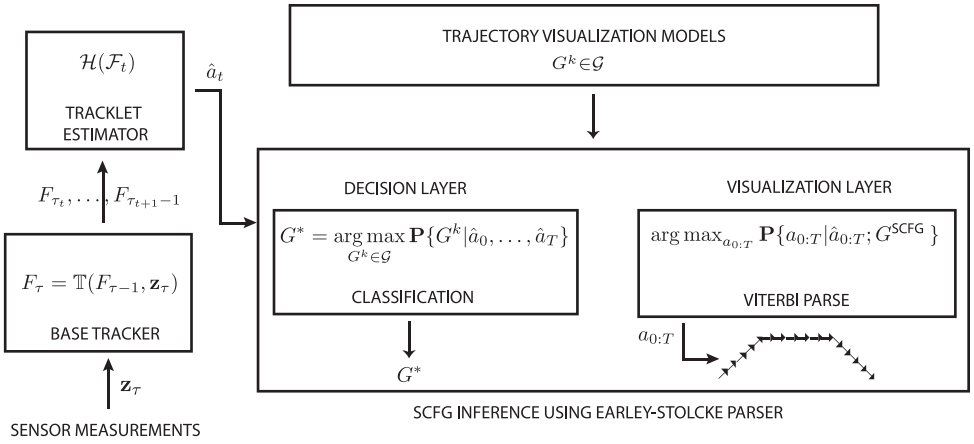


**Figure 3.**
The proposed hierarchical tracking framework to assist human radar operators. A base-level tracker $\mathbb{T}$ outputs filtered state estimates at a fast timescale $\tau$. The tracklet estimator aggregates state estimates from the base tracker and emits quantized tracklets $\hat{a}_t$ at a slower timescale $t$. A set of SCFG models for different threat scenarios is considered in the classification problem. The sequence of noisy tracklets $\hat{a}_{0:T}$ is fed into the Earley-Stolcke parser either to perform model classification to generate alarms or, as a visualization tool, to recover the suspicious trajectory.

Multitarget tracking (MTT) has a rich literature comprising various powerful tracking algorithms like the joint probabilistic data association [17], interacting multiple-model filter [1], multiple hypothesis tracking [18], and probability hypothesis density filter [19]. A taxonomy of MTT approaches is provided in [20]. The proposed work is not an alternative to a comprehensive tracker but a complementary tool to pick out anomalous trajectories at a coarser scale. Tracking errors manifesting as missed or spurious detections, incomplete or false tracks, and errors in data association are assumed to have originated and/or been corrected at the fine timescale. Such errors are not explicitly treated in this article but are regarded as noise parameters, which can be incorporated into the higher-level logic. Previous articles [21] and [22] dealt with some of these tracker limitations more explicitly.

## HIERARCHICAL TRACKING FRAMEWORK TO ASSIST HUMAN OPERATORS

In this section, a system-level description of the tracking framework is presented (Figure 3). A conventional (base-level) target tracker operates on the fast timescale, while the higher-level middleware layer operates on a slower timescale.

### BASE-LEVEL TRACKER

The base-level tracker is a Bayesian filter operating on the fast timescale (order of seconds) denoted by $\tau = 1, 2, \ldots$. The base-level tracker can be represented as an operator $\mathbb{T}$ that uses sensor measurements $\mathbf{z}_\tau$ to update a posterior filtering distribution $F_\tau$ over the position and velocity of the target by

$$F_\tau = \mathbb{T}(F_{\tau-1}, \mathbf{z}_\tau). \tag{1}$$

For example, consider a target represented by its kinematic state $\mathbf{s} = [s^x, s^y, s^{\dot{x}}, s^{\dot{y}}]^{\mathrm{T}}$, The state variables $(s^x, s^y)$ refer to the position of the target, while $(s^{\dot{x}}, s^{\dot{y}})$ refer to the velocity of the target in Cartesian coordinates. Classical target tracking uses a state space model

$$\mathbf{s}_\tau + 1 = f(\mathbf{s}_\tau, \mathbf{w}_\tau),$$
$$\mathbf{z}_\tau = h(\mathbf{s}_\tau) + \mathbf{v}_\tau,$$

where $\mathbf{w}_\tau$ and $\mathbf{v}_\tau$ represent the state and the measurement noise, respectively. The state transition and measurement functions are represented by $f(\cdot)$ and $h(\cdot)$, respectively. A base-level tracker estimates the target state trajectory from the radar measurements in a causal manner. This is a filtering problem involving computation of (and approximation to) the a posteriori filtering distribution $F_\tau = \mathbf{P}\{\mathbf{s}_\tau | \mathbf{z}_0, \mathbf{z}_1, \ldots, \mathbf{z}_\tau\}$, where $\mathbf{P}\{A\}$ refers to the probability of event $A$.

### TRACKLET ESTIMATION

Let $t = 1, 2, \ldots$ denote a slower timescale (on the order of minutes), which we call the epoch scale and at which the human analyst makes decisions. The human analyst reduces the posterior distributions $F_\tau$ to the mean position/velocity and quality of estimate (e.g., variance). More specifically, given the sequence of track distributions, $\mathcal{F}_\tau = \{F_{\tau t}, \ldots, F_{\tau t+1} - 1\}$, define a tracklet on the slow timescale as

$$\hat{a}_t = \mathcal{H}(\mathcal{F}_t) \sim \mathbf{P}\{\cdot | a_t\}. \tag{2}$$

Here, the tracklet $\hat{a}_t$ denotes, a quantization of an average state vector $\hat{s}_t$ of the target obtained from the tracklet estimator at epoch $t$ and can be viewed as a noisy version of the "true" underlying quantized position and/or unit velocity vector of the target denoted as $a_t$.

Two types of tracklets are considered: (a) the position tracklets $\hat{a}_t^{\text{pos}}$, which are output by the position tracklet estimator $\mathcal{H}$

pos, and (b) the velocity tracklets $\hat{a}_t^{\text{vel}}$, which are quantized through $\mathcal{H}_{\text{vel}}$. Tracklets are used as syntactic subunits of target trajectories. The SCFG shape models utilize velocity tracklets as subunits of the trajectory shape, while SCFG destination-specific pattern models utilize position tracklets as subunits of a goal-directed trajectory (following a specific pattern of visited sites).

The position tracklet estimator operates on a discretized surveillance space $\Lambda$ over which the target is observed. At each time instant, the position tracklet estimator quantizes the average $(\hat{s}_t^x, \hat{s}_t^y)$ state estimates to the closest element $(x_i, y_i) \in \Lambda$ on the discretized two-dimensional grid $\Lambda$, as shown in Figure 4a. The velocity tracklet estimator utilizes the average velocity estimate $(\hat{s}_t^{\dot{x}}, \hat{s}_t^{\dot{y}})$ to find the direction of motion of the target. The possible directions of motion of the target are quantized into eight radial cardinal directions from the set

$$\mathcal{V}^{\text{vel}} = \left\{ \vec{a} = -\pi, \vec{b} = -\frac{3\pi}{4}, \vec{c} = \frac{-\pi}{2}, \vec{d} = \frac{-\pi}{2}, \vec{e} = 0, \vec{f} = \frac{\pi}{4}, \vec{g} = \frac{\pi}{2}, \vec{h} = \frac{3\pi}{2} \right\}.$$

The cardinal directions represented by mode are shown in Figure 4b. Each mode is represented with a lowercase letter under an arrow to denote that it is a unit directional vector.

The framework presented in this section and the prior section on the base-level tracker does not consider the effects of missed and/or spurious detections. The SCFG framework, however, can also be used to deal with such scenarios, as shown in [22]. A simpler approach based on modifying the grammar was presented in [21].

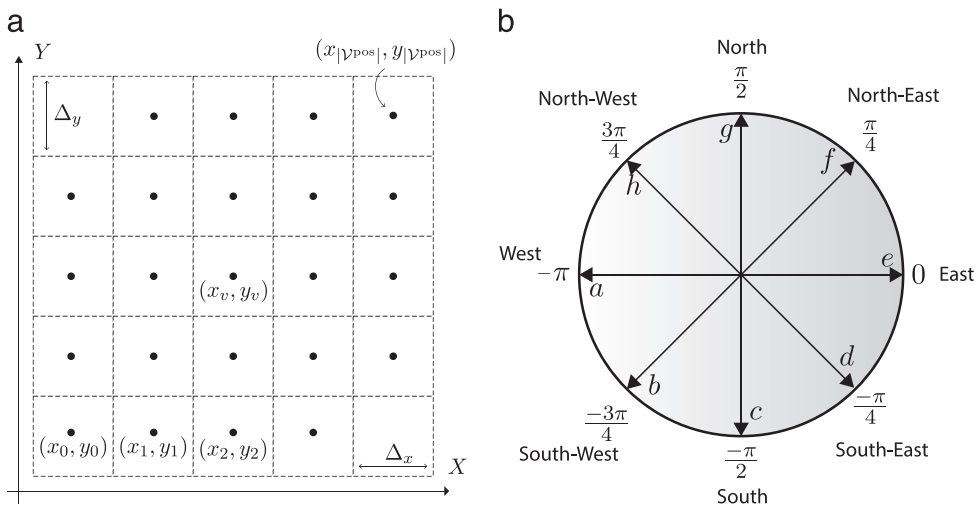


**Figure 4.**
(a) The discretization of a surveillance space $\Lambda$ into grids $(x_i, y_i)$, each of size $\Delta_x \times \Delta_y$. Each grid element is lexicographically ordered into a set $\mathcal{V}^{\text{pos}}$. There are $|\mathcal{V}^{\text{pos}}|$ bins in the surveillance space. (b) The velocity tracklets $v \in \mathcal{V}^{\text{vel}}$, representing cardinal radial directions.

## TRAJECTORY CLASSIFICATION OBJECTIVE

A target trajectory is associated with a specific spatiotemporal pattern depending on its shape and/or the pattern of sites visited by the target. Each trajectory is assumed to be generated by a model $G_k \in \mathcal{G}$, $k = \{1, \ldots, K\}$, where there are $k = |\mathcal{G}|$ different types of anomalous patterns under consideration. Development of these models is the main idea of this article and is described in the next section. As a target moves in a region of interest, it generates tracklets $\hat{a}_t, t = 0, 1, \ldots$. The anomalous trajectory classification task is then defined as finding the model $G^* \in \mathcal{G}$ that has the highest probability of explaining the observed tracklet sequence $a_0, \ldots, a_T$:

$$G^* = \arg\max_{G^k \in \mathcal{G}} \mathbf{P}\{G^k \mid \hat{a}_0, \ldots, \hat{a}_T\}. \tag{3}$$

## SCFG MODELS FOR ANOMALOUS PATTERNS

In this section, eight types of models for anomalous target trajectories are presented. Readers who are unfamiliar with the SCFG formalism should read the tutorial material in Appendix A. The motivation behind using SCFG models like arcs, rectangles, and palindromes is that such trajectories cannot be exclusively generated by Markov models.[2] A formal proof of this assertion can be constructed using the pumping lemma for regular languages (HMMs) [23], [24].

## RANDOM WALK MODELS

Random walks are the most elementary of Markov models for target trajectories evolving on a road network represented as a finite set of nodes $\mathcal{V}^{\text{pos}} = \{v_i i, = 1, \ldots, | \mathcal{V}^{\text{pos}} |\}$ (Figure 4a). The target position $a_t$ can be represented using the bin $v_i \in \mathcal{V}^{\text{pos}}$ in which it resides at instant $t$. A Markov chain model $G^{\text{MC}}$ of target dynamics over such a road network can be parameterized using a prior distribution $\mathbf{P}\{a_0 = v_i\}$ over the initial state $a_0$ at time $t = 0$ and a possibly time-varying transition matrix $A_t(i, j) = \mathbf{P}\{a_t = v_j | a_{t-1} = v_i\}$. A transition matrix for such random walk models can be calculated based on physical distance between nodes. The state transition diagram of such a model is shown in Figure 5.

## RECIPROCAL MODELS FOR DESTINATION-AWARE TRAJECTORIES

A human target following anomalous behavior rarely moves according to a random walk model. The sequence of actions taken by an intelligent agent is often premeditated on a global scale with random variation at the local scale. With this assumption in mind, the local dynamics of a target are more appropriately modeled using a reciprocal process; see also our previous article [3].

[2] This implies that a Markov chain can generate a sample path that is an instance of a shape with some finite probability. However, it cannot do so with a probability of 1 unless the state space is extended artificially to the length of the trajectory.

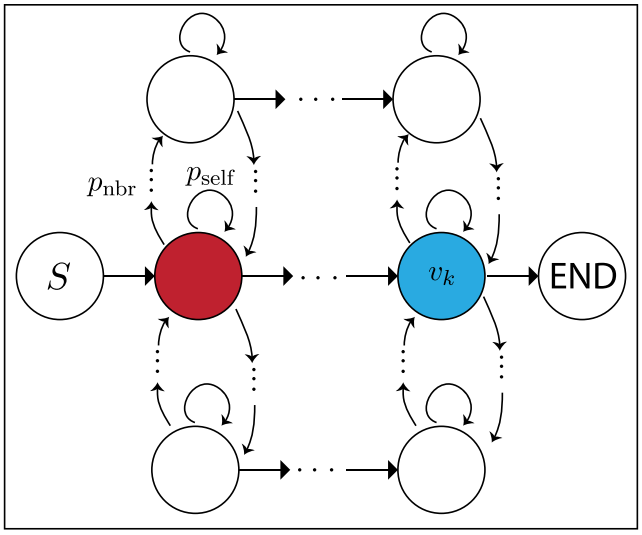


**Figure 5.**
The state transition diagram for target trajectories on a road network. For a random walk model, the transition probabilities $p_{\text{nbr}}$ are nonzero only for neighboring nodes that are connected by streets. The neighbor probabilities are uniformly distributed over all connected neighbors. For the Markov bridge model, the transition probabilities are time varying.

A discrete-time reciprocal process $a_t \in \mathcal{V}^{\text{pos}}$ is a one-dimensional Markov random field with the noncausal property

$$\mathbf{P} = \{a_t = v_j \mid a_{s \neq t}\} = \mathbf{P} = \{a_t = v_j \mid a_{t-1} = v_i, a_{t+1} = v_l\}, \tag{4}$$

parameterized by the homogeneous three-point transitions $Q(i, j, l) = \mathbf{P}\{a_t = v_j | a_{t-1} = v_i, a_{t+1} = v_l\}$ with $v_i$, $v_j$, and $v_i \in \mathcal{V}^{\text{pos}}$. In urban environments, traffic information can be used to estimate the three-point transitions $Q(i, j, l)$ between intersections in a road network. This is described in the section on single-target scenarios.

Using reciprocal dynamics, a destination-aware trajectory is defined by a priori fixing the final destination of the target $a_T = v_k$. The resulting Markov bridge is characterized by a probability transition law

$$\mathbf{P}\{a_t \mid a_{t-1}, a_T\} = \frac{\mathbf{P}\{a_t \mid a_{t-1}, a_{t+1}\}}{\mathbf{P}\{a_{t+1} \mid a_t, a_T\}} \mathbf{P}\{a_{t+1} \mid a_{t-1}, a_T\},$$

which induces a backward recursion for time-varying two-point Markov transitions

$$B_t^k(i,j) = \frac{Q(i,j,l)}{B_{t+1}^k(j,k)} \left( \sum_{j' \in \mathcal{V}^{\text{pos}}} \frac{Q(i,j',l)}{B_{t+1}^k(j',l)} \right)^{-1}. \tag{5}$$

The second term on the right-hand side of (5) is the normalization constant. The time-varying transitions $B_t^k(i,j) = \mathbf{P}\{a_t = v_j \mid a_{t-1} = v_i, a_T = v_k\}$ refer to two-point transitions of a target with final destination $a_T = v_k$. To ensure that the final destination of the target $a_T = v_k$, we initialize the recursion with $B_{T-1}^k(i,j) = 1.0$ for $j = k$ and 0.0 otherwise. In addition, at time $t = T - 2$, the target transitions according to $B_{T-2}^k(i,j) = Q(i,j,k)$. A more detailed treatment can be found in [3].

## DESTINATION-AWARE PATHS

A destination-aware path is a target trajectory that is heading toward a known destination $a_T = v_k$ while following local dynamics according to the three-point transitions $Q(i, j, l)$. The Markov bridge approach can also be viewed as a time-varying SCFG for destination-specific trajectories.

A destination-aware trajectory satisfies the constraint $\mathbf{P}\{a_T = v_k\} = 1.0$. A destination-aware SCFG model can be defined using a starting rule of the form $S \rightarrow v_i X_i v_k$ with rule probability $\mathbf{P}\{S \rightarrow v_i X_i v_k\} = \mathbf{P}\{a_0 = v_i | a_T = v_k\}$. The destination-constrained SCFG model is characterized by rules of the form $X_i \rightarrow v_j X_j$ with time-varying rule probabilities given by the Markov bridge transitions such that $\mathbf{P}\{X_i \rightarrow v_j X_j\} = B_t^k(i,j)$, where $B_t^k(i,j)$ represents the probability in (5). Such rules are only created for neighboring nodes $v_i$, $v_j$ that are connected by an edge. Suppose $v_k$ represented the position of a sensitive asset like an embassy or a checkpoint. Using the approach presented earlier, a grammatical model $G_k^{\text{SCFG}}$ represents all trajectories with the target destination $v_k$.

## TARGET RENDEZVOUS

Consider the trajectories followed by two targets[3] represented by position tracklets $a_t^1$ and $a_t^2$. A rendezvous is defined as an anomalous event where two targets meet at the same position $v_k$ at time $T$. Such a situation is depicted in Figure 7. The rendezvous of two targets at a given node $v_k$ can be considered a destination-aware trajectory and modeled using the grammar rules described in the previous section. Define the multiple target position tracklet sequence $\mathbf{a}_t = \left[a_t^1, a_t^2\right]^{\mathrm{T}}$ as a vector concatenation. The rendezvous of two targets is then modeled as a destination-aware trajectory $\mathbf{a}_0, \ldots, \mathbf{a}_{T-1}$, $\mathbf{a}_T = [v_k, v_k]^{\mathrm{T}}$, with the final state of both targets pinned to the intended meeting point $v_k$ at time $T$. The precomputed time-varying transition matrix $B_t^k(i,j)$ is used in an extended state space model to define a higher-order (two-target) transition matrix

$$B_t^{k,k}(m,n) = B_t^k(i,j) \otimes B_t^k(i,j), \tag{6}$$

where $i,j \in \mathcal{V}^{\text{pos}}$ and $m,n \in \mathcal{V}^{\text{pos}} \times \mathcal{V}^{\text{pos}}$. The $\otimes$ operator represents the Kronecker product between two matrices. The notation $B_t^{k,k}(m,n)$ represents the time-varying transition probability $\mathbf{P}\{\mathbf{a}_t = n \mid \mathbf{a}_{t-1} = m, \mathbf{a}_T = [v_k, v_k]^{\mathrm{T}}\}$. The grammar model for target rendezvous has the same structure as destination-aware trajectory models. However, the nonterminal and terminal sets are different due to the expansion in the state space. The time-varying rule probabilities are specified in (6).

## PALINDROME PATHS

A palindrome refers to a sequence $a_0 a_1 \ldots a_{T-1} a_T = a_T a_{T-1} \ldots a_1 a_0$ such that reversing the sequence results in the same pat-

[3] The restriction to two targets is for simplification of notation; the model can be extended to an arbitrary number of targets. However, the state space increases exponentially with the number of targets.

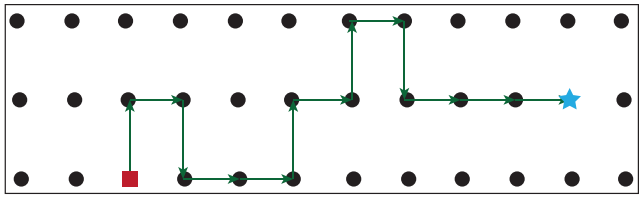

**Figure 6.**
The blue star node is the destination node and the red square node is the initial starting point. The target trajectory follows a destination-aware model to travel between the initial and the destination nodes.

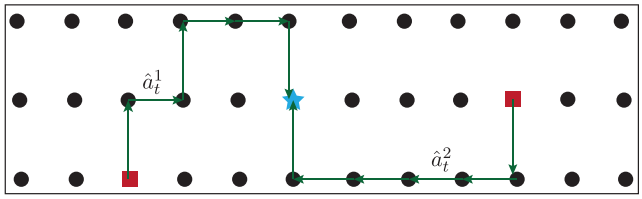


**Figure 7.**
The two targets, represented by their trajectories $\hat{a}_t^1$ and $\hat{a}_t^2$, follow a rendezvous model to meet at the blue star destination node.

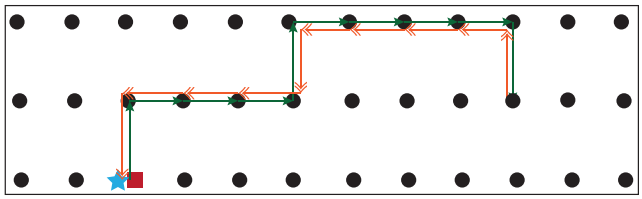

**Figure 8.**
The red square and blue star represent the same starting and ending node. The green lines are the first part of the trajectory, while the orange lines show the second half of the retraced trajectory. Such a pattern is called a palindrome.

tern. In the context of target trajectories using position tracklets, a palindromic path refers to a target retracing its previously traversed path. Such paths can be indicative of behaviors such as searching for a dropped or lost item. A simple grammar for a palindromic path is shown later in Figure 15a, which is characterized by rules of the form $X_i \rightarrow v_j X_j v_j$. Due to the inside–outside (branching) manner in which an SCFG generates a sequence, we can ensure the palindrome property by emitting a matching terminal on the left- and right-hand sides when transitioning to a new node $v_j$.

*Remark:* The next few models generate geometric shapes using velocity tracklets as geometric primitives. The clean terminals $a_t$ and noisy observations $\hat{a}_t$ take values from the cardinal direction set $\mathcal{V}^{\text{vel}}$ shown in Figure 4b. These generalize our past article [3] by incorporating system uncertainty in the movement patterns of the target. This generalization is important because a target attempting to travel in a specific geometric shape does not have a global view of the evolution of its trajectory. As a result, the local movement patterns of the target are subject to small perturbations. The internal process noise is incorporated by allowing each directional movement to have a finite probability of perturbation into neighboring radial directions.

## LINEAR TRAJECTORIES

Linear trajectories are straight paths that are generated by constant velocity (CV) target dynamics obeying local Markov dependency. Linear grammar models are represented using the compact form $G^{\text{line}} = \{\vec{a}^n\}$, implying that the model can generate all trajectories involving $n$ movements of a target in the direction represented by the unit vector $\vec{a}$. A simple regular grammar for lines is characterized by rules of the form $S \rightarrow \vec{a}S|\vec{a}$, with $\vec{a} \in \mathcal{V}^{\text{vel}}$ representing the target's direction of motion.

## ARCLIKE TRAJECTORIES

The compact form for arclike trajectories is $G^{\text{arc}} = \{\vec{a}^n\vec{b}^+\vec{c}^n\}$, which is characterized by equal movements in opposing directions represented by the unit vectors $\vec{a}$ and $\vec{c}$ (Figure 9a). The notation $\vec{b}^+$ denotes an arbitrary number of movements in the direction represented by $\vec{b}$. A simple grammar capable of generating arcs of all lengths is shown in Figure 9b. We use the notion of arcs to represent U-turn and open trapezoidal patterns.

## RECTANGLE TRAJECTORIES

We consider the modified-rectangle (m-rectangle) language (with associated grammar shown in Figure 10b) as $G^{\text{m-rectangle}} = \{\vec{a}^m\vec{b}^+\vec{c}^m\vec{d}^*\}$. The m-rectangle grammar 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. The notations $\vec{b}^+$ and $\vec{d}^*$ represent an arbitrary number of movements in the corresponding directions represented by that mode. A full rectangle with both opposite sides of equal length cannot be modeled by an SCFG [24].

## SUMMARY

In formal language theory, it is known (and provable using pumping lemmas [23]) that trajectories like arcs, rectangles, and palindromes are impossible to generate using Markov models exclusively [23]. In addition, the incorporation of reciprocal dynamics using time-varying rule probabilities allows destination-aware and rendezvous trajectories to be efficiently modeled. This section illustrated the use of such non-Markovian models in trajectory classification.

# ILLUSTRATIVE EXAMPLE: MULTITARGET PATTERN OF LIFE CHANGE DETECTION

In this section, multiple targets are incorporated into anomaly detection as a change detection problem. The aggregate behavior of all targets moving in the surveillance space $\Lambda$ is assumed to arise from reciprocal dynamics characterized by the three-point transitions $Q(i, j, l)$. An anomaly can be defined as a change in the underlying reciprocal dynamics such that a normal regime $Q^{(1)}(i, j, l)$ is in effect in the time interval $t = \{0, 1, \ldots, t_0 - 1\}$ and an abnormal regime $Q^{(2)}(i, j, l)$ is in effect in the time interval $t = \{t_0, t_0 + 1, \ldots, T\}$.

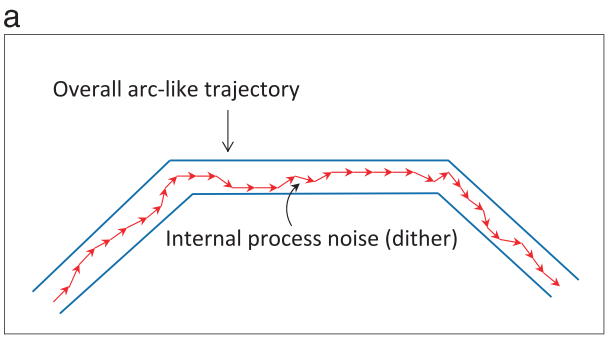


**Figure 9.**
(a) An arclike trajectory with system uncertainty. The incorporation of internal process noise serves the philosophical purpose of modeling overall geometry using self-embedding rules to generate shapes that have sides with equal lengths while also allowing internal perturbations. In the case of a human target, such a process noise models the inability of the target to globally monitor its trajectory. (b) A simple grammar model for arclike trajectories.

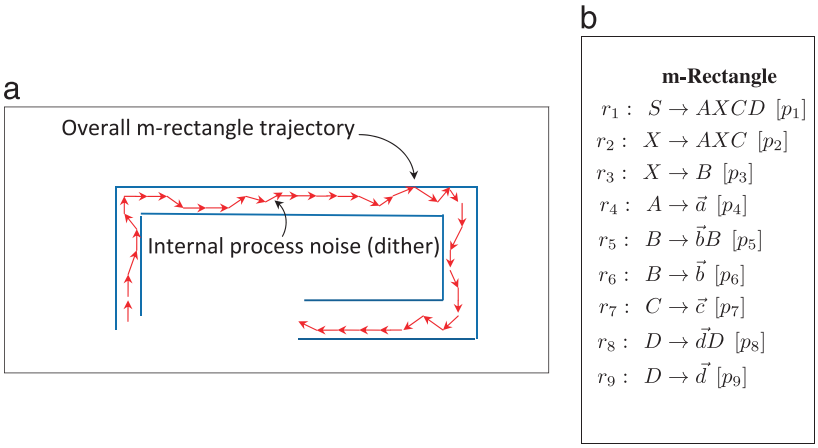


**Figure 10.**
(a) An m-rectangle trajectory with internal perturbations and global shape. (b) The grammar is able to generate all sizes of trajectories satisfying such a shape.

In [3], a surveillance application is described that seeks to detect changes in the pattern of life of a local population. Often, the local population is sympathetic toward insurgent and rebel groups. Information about anomalous events like the installation of an improvised explosive device (IED) can quickly propagate through the local population, which manifests itself in the change of traffic patterns around the area where the IED was placed. A pictorial depiction of such a scenario is shown in Figure 11. A sensitive asset is located at node $v_k$. The normal operation of targets follows the behavior shown using dashed green lines. However, a hypothetical anomaly in the vicinity of node $v_k$ causes target behavior to change significantly, as depicted by the solid red lines.

Change detection can be described as the problem of determining if or when, during the observation interval, the underlying reciprocal dynamics switches between two known models. We impose no prior knowledge on the switching time (we consider it to be an unknown deterministic quantity). Consider a stacked vector $\hat{\mathbf{a}}_t = \left[\hat{a}_t^1, \ldots, \hat{a}_t^n, \ldots, \hat{a}_t^N\right]^{\mathrm{T}}$ representing the noisy position tracklets $\hat{a}_t^n$ of each target $n = \{1, \ldots, N\}$ existing in the surveillance space $\Lambda$ at time $t$. The total number of targets is denoted by $N$. For notational convenience, the number of targets is assumed to be fixed and all targets begin moving at epoch $t = 0$. The likelihood computation of the multitarget tracklet sequence $\mathbf{P}\{\hat{a}_{0:T}\}$ factorizes over individual targets because the trajectory of each target is assumed to be independent of other targets and only depends on the underlying reciprocal dynamics. Recall that destination-aware targets have their final state pinned to some destination $v_k$ reached in a fixed number of movements from epoch 0 to epoch $T$. Consequently, they are expected to arrive at their destination after $T$ time steps. The vector tracklet sequence $\hat{\mathbf{a}}_{0:T}$ obeys one of the following hypotheses:

$H_1$: The tracklet sequence $\hat{a}_{0:T}^n$ of each target $n$ being tracked was generated by an SCFG $G^{(1)}$ with underlying reciprocal dynamics $Q^{(1)}$.

$H_{t_0}$: For some $t_0 \in \{1, \ldots, T-2\}$, the tracklets $\hat{a}_t^n$ were generated by an SCFG $G^{(1)}$ with normal underlying reciprocal dynamics $Q^{(1)}$ for $t = \{0, \ldots, t_0 - 1\}$ and by an SCFG $G^{(2)}$ with anomalous reciprocal dynamics $Q^{(2)}$ for $t = \{t_0, \ldots, T\}$.

$H_{T-1}$: The entire tracklet sequence $\hat{a}_{0:T}^n$ of each target $n$ was generated by an SCFG $G^{(2)}$ with underlying reciprocal dynamics $Q^{(2)}$.

Our aim is to provide a maximum likelihood technique to detect (a) whether a change occurred from the normal operating regime and, if so, (b) the time $t_0$ at which the change occurred. Toward this goal, consider the log-likelihood $L(T, t_0) = \mathbf{P}\{\hat{\mathbf{a}}_{0:T} \mid H_{t_0}\}$ of the noisy observed multitarget tracklet sequence $\hat{\mathbf{a}}_0, \ldots, \hat{\mathbf{a}}_T$ of all targets existing in the surveillance space:

$$\begin{aligned} L(T,t_0) &= \log \mathbf{P}\{\hat{\mathbf{a}}_0, \ldots, \hat{\mathbf{a}}_T \mid H_{t_0}\} \\ &= \sum_{n=1}^{N} \log \mathbf{P}\{\hat{a}_0^n, \ldots, \hat{a}_T^n \mid H_{t_0}\} \\ &= \sum_{n=1}^{N} \underbrace{\log \mathbf{P}\{\hat{a}_0^n, \ldots, \hat{a}_{t_0-1}^n \mid G^{(1)}\}}_{\text{from Earley-Stolcke parser in (13)}} + \underbrace{\log \mathbf{P}\{\hat{a}_{t_0}^n, \ldots, \hat{a}_T^n \mid G^{(2)}}_{\text{from Earley-Stolcke parser in (13)}} \end{aligned} \tag{7}$$

The multitarget anomaly detection problem is cast as a change detection problem seeking the hypothesis $H_{t_0}$ with the maximum likelihood of explaining the observed vector tracklet sequence

$$t_0^* = \arg\max L(T, t_0), \tag{8}$$

where it is understood that $t_0^* = 1$ implies that the normal regime was in operation during the observation period and $t_0^* = T - 1$ implies that the anomalous regime was in operation during the entire observation period $t = 0, \ldots, T$. The log-likelihood in (7)

can also be computed in a recursive manner for each target using the prefix probability in (11).

## NUMERICAL EXAMPLES

In the next section, the classification performance of single-target trajectories is considered. The section on the pattern of life analysis explores the use of grammatical models toward rendezvous detection for two targets and change detection in aggregate target behavior for pattern of life analysis.

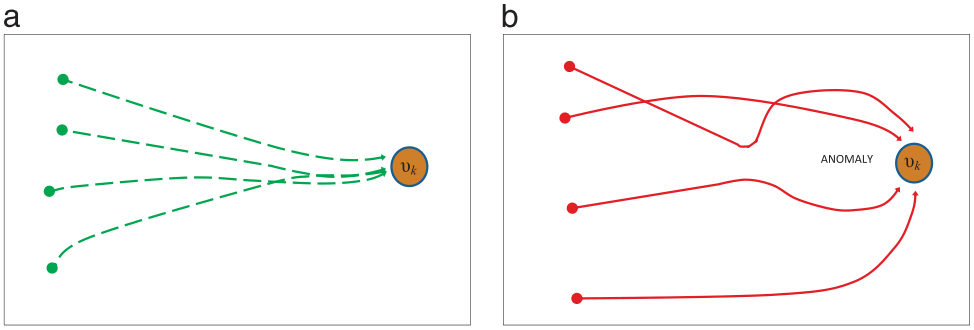


**Figure 11.**
(a) The node $v_k$ represents the location of a sensitive asset and forms the endpoint of a destination-aware trajectory model $G_k^{\text{SCFG}}$. In a normal pattern of life, local traffic flows toward $v_k$, with sample trajectories shown as green dashed lines. (b) When an anomalous event occurs near the node $v_k$, the pattern of life switches to an abnormal state, with sample trajectories (shown in solid red) arriving at $v_k$ using aberrant paths in an attempt to avoid the anomalous event.

## SINGLE-TARGET SCENARIOS

SCFG models are compared to equivalent HMMs whose parameters are learned by a Baum-Welch estimation procedure over synthetic tracklet sequences generated to obey its characteristic feature (such as its shape or destination). We demonstrate the effectiveness of our approach using receiver operating characteristic (ROC) curves for detection performance.

### *Velocity Tracklet–Based Simulations*

A continuous-valued target state trajectory $\mathbf{s}_{0:T}$ is generated for three kinds of shapes: lines, arcs, and m-rectangles. For example, an arc (like that shown in Figure 1) can be generated by concatenating the trajectories generated by three CV models. The first CV model is initialized using a normal distribution that has a mean with the position at origin and a unit velocity in the northeast direction such that $\mathbf{P}\{\mathbf{s}_0\} \sim \mathcal{N}\left([0,0,\frac{1}{\sqrt{2}},\frac{1}{\sqrt{2}}]^\intercal,\Sigma\right)$. The covariance matrix $\Sigma$ can be chosen to generate trajectories with different signal-to-noise ratios. This model is then used to generate the upward part of the arc for $\tau = 0, \ldots, \tau_1$ time points. The second CV model is initialized using the final position of the target in CV model 1 and with velocity magnitude $v_{\tau_1}^{\text{mag}} = \sqrt{(s_{\tau_1}^{\dot{x}})^2 + (s_{\tau_1}^{\dot{y}})^2}$ equal to the final velocity of the target in CV model 1. However, the velocity magnitude is concentrated in the x-axis such that $\mathbf{P}\{s_{\tau_1+1}\} \sim \mathcal{N}\left([s_{\tau_1}^x, s_{\tau_1}^y, v_{\tau_1}^{\text{mag}}, 0]^\intercal, \Sigma\right)$. Finally, the last part of the arc trajectory is created by running a third CV model for time points $\tau = \tau_2 + 1, \ldots, T$ such that the initial state

$$\mathbf{P}\{\mathbf{s}\tau_{2+1} \sim \mathcal{N}\left([s_{\tau_2}^x, s_{\tau_2}^y, \frac{v_{\tau_2}^{\text{mag}}}{\sqrt{2}}, -\frac{v_{\tau_2}^{\text{mag}}}{\sqrt{2}}]^\intercal, \Sigma\right).$$

In all simulations, we generate trajectories of a length of 1,000 time points at the fine timescale $\tau$ with randomly chosen segment lengths obeying shape characteristics. Monte-Carlo runs of more than 1,000 trajectories are used for the results shown in Figure 12 and all subsequent simulations. A noisy radar observation process is used, and the measurements $\mathbf{z}_\tau$ are fed into an extended Kalman filter. Ideal detection is assumed, and the data association problem associated with multiple targets is not incorporated in the simulations. The filter estimates are then aggregated every 20 time points, and velocity tracklets $\hat{a}_t$ are generated at the slower timescale $t$. For the shape-based trajectories, we create four types of models: $\mathcal{G}^{\text{SCFG}} = \{G_{\text{line}}^{\text{SCFG}}, G_{\text{arc}}^{\text{SCFG}}, G_{\text{m-rectangle}}^{\text{SCFG}}, G_{\text{arbitrary}}^{\text{SCFG}}\}$ and $\mathcal{G}^{\text{HMM}} = \{G_{\text{line}}^{\text{HMM}}, G_{\text{arc}}^{\text{HMM}}, G_{\text{m-rectangle}}^{\text{HMM}}, G_{\text{arbitrary}}^{\text{HMM}}\}$ using both SCFG and HMM frameworks. The $G_{\text{arbitrary}}$ models for arbitrary trajectories are also added to each model set. The transition matrix of the arbitrary HMM is set up such that the transition probability from tracklet $v_i$ to $v_j$ is inversely proportional to the angular separation of the directions represented by $v_i$ and $v_j$. It can represent jumps from any movement direction to all other directions. Such a model is the archetype of a random walk. A similar model is also used in grammatical form for the arbitrary SCFG model. Equivalent HMMs for lines, arcs, and rectangles are created using a left-to-right transition structure, and the transition probabilities are learned from synthetically generated example sequences for each shape.

### *Position Tracklet–Based Simulations*

Position-tracklet trajectories depend on the incorporation of the three-point transitions $Q(i, j, l)$ described in the section on random walk models. Traffic authorities in various cities collect and store the turns ratio and traffic flow information over road networks using traffic cameras and induction loop sensors. Traffic flow $\omega(i,j)$ is a count of vehicles traveling between two neighboring nodes $i$ and $j$. The traffic flow is undefined for nodes that are not connected by a road. The turns ratio $\kappa(i, j, l)$ at node $v_j$ represents the proportion of cars turning toward node $v_l$ when arriving at node $v_j$ from node $v_i$. The three-point transitions $\hat{Q}(i,j,l)$ can be empirically estimated using these quantities as

$$\hat{Q}(i,j,l) = \frac{\omega(i,j) \odot \omega(j,l)}{\sum_{j'} \omega(i,jp') \odot \omega(j',l)} \approx \frac{\omega(i,j)\kappa(i,j,l)}{\sum_{j'} \omega(i,jp')\kappa(i,j',l)}, \tag{9}$$

where $\omega(i,j) \odot \omega(j,l)$ denotes the number of vehicles that travel first from node $v_i$ to node $v_j$ and finally on to node $v_l$. However, this quantity cannot be computed without specifically identifying each vehicle. Consequently, the approximation in (9) is used. In the absence of traffic data, auxiliary information from cellular localization or vehicular global positioning system traces can also be used to estimate $Q(i, j, l)$. In such cases,

the estimation does not require the approximation in (9), because each cell phone or vehicle can be uniquely identified. However, the traces must be quantized to grid positions in the surveillance space. For the purposes of simulations, a synthetic three-point transition matrix is used such that the three-point transitions $Q(i, j, l)$ at each node $v_j$ are inversely proportional to the distance between $v_l$ and destination node $v_k$. Such a choice biases the target to make transitions to reach the destination $v_k$ in the shortest number of hops.

### Destination-Aware Trajectories

Using the three-point transitions described earlier and the time-varying transitions in (5), a destination-aware trajectory model $G_k^{SCFG}$ is created for an arbitrary node $v_k$, which was chosen as an interesting node hypothetically containing a sensitive asset. We then simulate multiple trajectories of length $T$ with node $v_k$ as the destination. These trajectories are observed using a noisy process such that

$$\mathbf{P}\{\hat{a}_i = v_i | a_i = v_j\} = \begin{cases} 0.8, \text{ for } v_i = v_j \\ \dfrac{0.2}{|\mathcal{N}_j|}, \text{ for } v_i \in \mathcal{N}_j \end{cases}, \tag{10}$$

where $\mathcal{N}_j$ refers to the set of intersections connected to node $v_j$ by a street and $|\mathcal{N}_j|$ refers to the number of connected nodes. This noise distribution was chosen to incorporate averaging and quantization effects from the tracklet estimator in (2). As before, models for both SCFG and HMM frameworks are created such that $\mathcal{G}_{\text{SCFG}} = \{G_k^{\text{SCFG}}, G_{i_1}^{\text{SCFG}}, \ldots G_{i_N}^{\text{SCFG}}\}$ and $\mathcal{G}_{\text{HMM}} = \{G_k^{\text{HMM}}, G_{i_1}^{\text{HMM}}, \ldots G_{i_N}^{\text{HMM}}\}$. The $G_{i_N}$ models are destination constrained but have a final destination at some node $i_n \neq k$. We chose several such nodes within 2–3 hops of the intended destination $k$. The detection performance can be seen in Figure 13a.

### Palindrome Trajectories

Examples are also provided for the detection of palindrome paths over the road network Λ A palindrome trajectory is artificially simulated by first generating an arbitrary trajectory for $[T/2]$ time points and then appending the reversed sequence to create a retraced path. The same noisy observation process as in (10) is used to obtain tracklet measurements $\hat{a}_{0:T}$. We use a destination-aware model with the same start $a_0 = v_k$ and end $a_T = v_k$ points represented as $G_{k,k}$, as well as a more general SCFG model allowing random transitions like a random walk within the model set $\mathcal{G}_{\text{SCFG}} = \{G_{\text{palindrome}}^{\text{SCFG}}, G_{k,k}^{\text{SCFG}}, \ldots, G_{\text{general}}^{\text{SCFG}}\}$. The equivalent HMM has a fully connected state transition diagram, and the rule probabilities are learned using Baum-Welch reestimation [25] from example palindrome trajectories generated from its generative SCFG model. The performance results are shown in Figure 13b.

The target rendezvous event can be considered an expanded state space version of the destination-aware trajectory model. Simulations show similar results to the single-target destination-aware case. As a result, simulation results are not shown for such events.

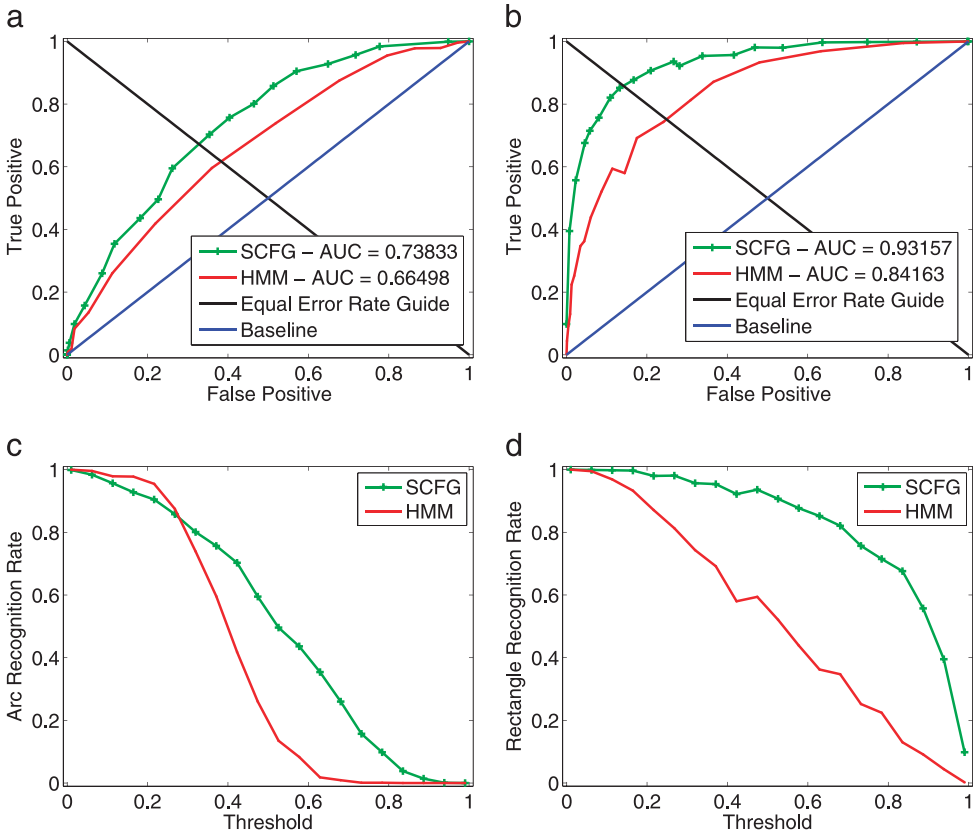


**Figure 12.**
(a) The ROC curve for SCFG arclike trajectories with an area under the curve (AUC) equal to 0.738 shows better performance than HMMs that have AUC = 0.664. The SCFG models also have a better absolute recognition rate for higher thresholds, as seen in (c). In (b), the ROC curve for SCFG m-rectangle trajectories with AUC = 0.928 shows better performance than HMMs with AUC = 0.841. The SCFG rectangle models also have better absolute recognition rates, as shown in (d).

## PATTERN OF LIFE ANALYSIS

The multitarget change detection problem presented in the section with the illustrative example is examined in this section. The normal operating regime consists of a regular 50-node network, as shown in Figure 2b. The normal regime is simulated by making the three-point transitions $Q(i, j, l)$ inversely proportional to the distance between the node $v_l$ and the destination node $v_k$ to bias the target to reach the destination as quickly as possible. An anomalous regime is then created by remov-

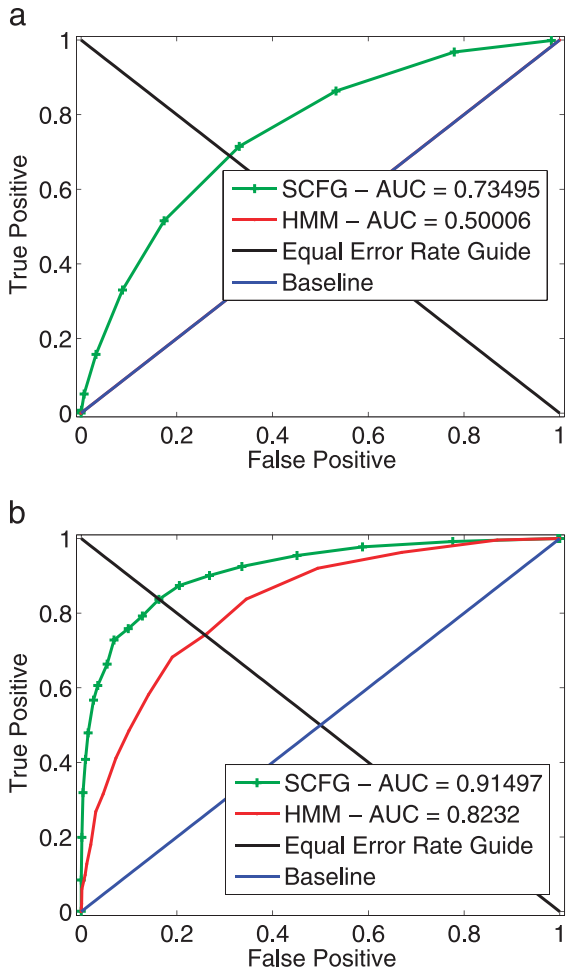


**Figure 13.**
(a) The ROC curve for an SCFG destination-aware trajectory shows a marked improvement over HMMs. The AUC for the SCFG models is 0.734, while the AUC for HMMs is equivalent to a random classifier with AUC = 0.50. In (b), the ROC curve for SCFG palindrome trajectories with AUC = 0.91 also shows a marked improvement over an HMM learned from example palindromes generated by an SCFG.

ing edges around a chosen interesting node $\upsilon_k$. In addition, the three-point transitions at each node $\upsilon_j$ are chosen proportional to the distance between the node $\upsilon_l$ and the destination node $\upsilon_k$. Such a choice is opposite to the normal regime, and it biases the target to avoid transitions, taking it closer to the destination node $\upsilon_k$.

Various single-target destination-aware trajectories are simulated heading toward the destination node $\upsilon_k$ using the normal regime. After a specific amount of time $t_0$, the trajectory models are switched to the abnormal regime. Using the procedure outlined in the section with the illustrative example, the different hypotheses $H_{t_0}$ are tested. The change point detection performance results are shown in Figure 14.

## CONCLUSION

In this article, a hierarchical tracking framework is proposed to assist human radar operators in the detection and forensic analysis of anomalous trajectory patterns. Geometric primitives like movement patterns and quantized positions are used as syntactic subunits within an SCFG framework to perform anomaly detection. The expressive power of SCFG models is able to capture long-term dependencies in intent-driven trajectories while utilizing efficient polynomial time algorithms for their inference. The Earley-Stolcke parser is modified using a scaling trick to allow processing of long tracklet sequences. A novel interpretation of reciprocal processes using time-varying SCFG rules is also presented. We devised novel models for anomalous events like target rendezvous and palindrome paths. Furthermore, the pattern-of-like-analysis problem was formulated as a change detection problem using multiple target trajectories. Finally, a comprehensive numerical evaluation of our proposed models was carried out to show an increase in the detection performance over conventional Markov models.

## APPENDIX A: TUTORIAL OVERVIEW OF SCFGS

### STRUCTURAL DESCRIPTION OF SCFG

A grammar $G = (\mathcal{X}, \mathcal{V}, \mathcal{S}, \mathcal{R})$ is a quadruple consisting of a set $\mathcal{X}$ of latent variables $X_i$, $i = 1, \ldots, |\mathcal{X}|$ called nonterminals; a set $\mathcal{V}$ of discrete observations $\upsilon_i$, $i = 1, \ldots, |\mathcal{V}|$ called terminals; a special start symbol $S$; and a set $\mathcal{R}$ of production rules $r_m$, $m = 1, \ldots, |\mathcal{R}|$. In the context of trajectory modeling, nonterminals represent hierarchical structural segments of a trajectory while terminals represent actual subunits of a trajectory. The production rules $r_m$ describe the manner in which the trajectory can evolve by combining the hidden structural parts of the trajectory. Each nonterminal $X_i$ can have $n_{X_i}$ alternative rules that can be chosen when nonterminal $X_i$ is being considering for rewriting.

A context-free grammar $G^{\text{CFG}}$ additionally has the property that its production rules can only have the form $X_i \rightarrow \alpha$, such that a nonterminal $X_i$ can be rewritten as an arbitrary string $\alpha \in (\mathcal{X} \cup \mathcal{V})^*$. The notation $(\mathcal{X} \cup \mathcal{V})^*$ denotes an arbitrary (possible empty) combination of nonterminal and terminal symbols. An SCFG $G^{\text{SCFG}}(G^{\text{CFG}}, \mathcal{P})$ is a pair consisting of a context-free grammar and a rule probability set $\mathcal{P} = \{p_m\}_{m=1}^{|\mathcal{R}|}$, assigning a probability $p_m \overset{\text{def}}{=} \mathbf{P}\{r_m\}$ to each rule $r_m \in \mathcal{R}$. The rule probabilities create a conditional probability distribution over all alternative expansions of a nonterminal $X_i$ such that $\sum_{m=1}^{n_{X_i}} p_m = 1.0$ for each $X_i$.

In the following discussion, an SCFG is treated as a triple stochastic process with two layers of latent variables. The rule choices used in the derivation of a sentence is the first latent layer. A graphical representation of the sequence of rewriting rules used until epoch $t$ is called a partial parse tree $\Psi_t$. The sequence of rule choices is also called a derivation that generates a sequence of "clean" terminal symbols $a_{0:T}$ on termination. The clean terminal symbols represent the second hidden layer because they are only observed in noise. The final observation process generates a sequence $\hat{a}_{0:T}$ of "noisy" terminals. The SCFG measurements are represented by an observation distribution $C(i,j) = \mathbf{P}\{\hat{a}_t = \upsilon_i \mid a_t = \upsilon_j\}$. The noisy observation layer can be subsumed into the grammar rules by creating a special "emitting" nonterminal for each terminal. However, this

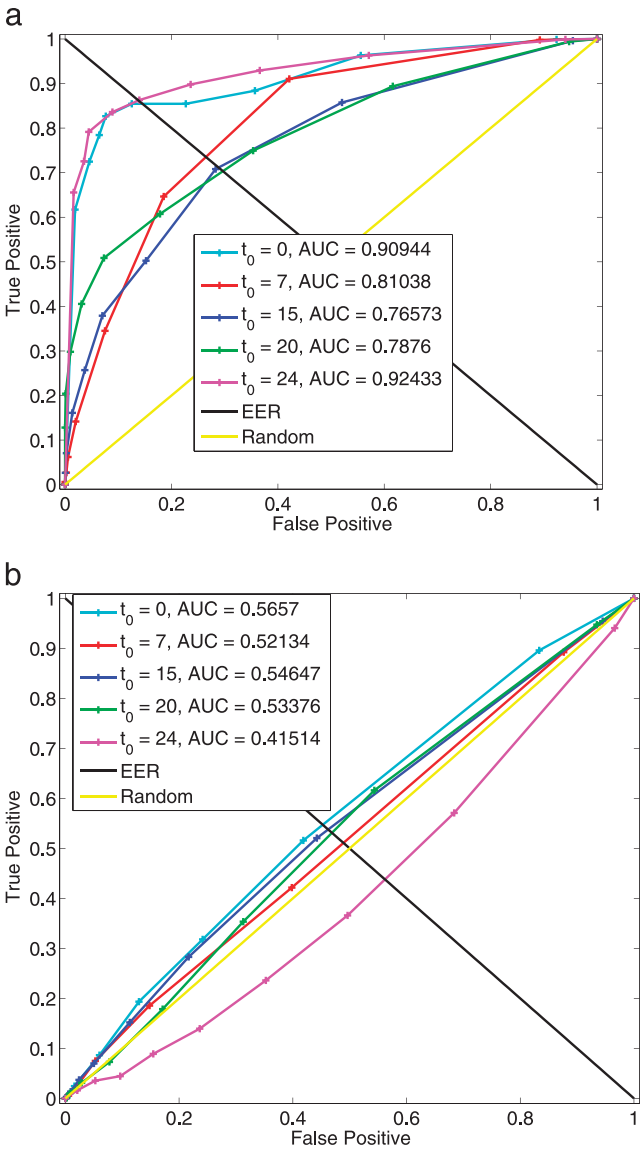


**Figure 14.**
(a) The SCFG model ROC curves for various switching times in trajectories with a length of 25 time points. (b) In comparison, we observe that HMMs cannot capture the target dynamics and hence perform poorly for pattern of life analysis. This result is in agreement with the result shown in Fig. 13a, because HMMs cannot represent destination-aware trajectories.

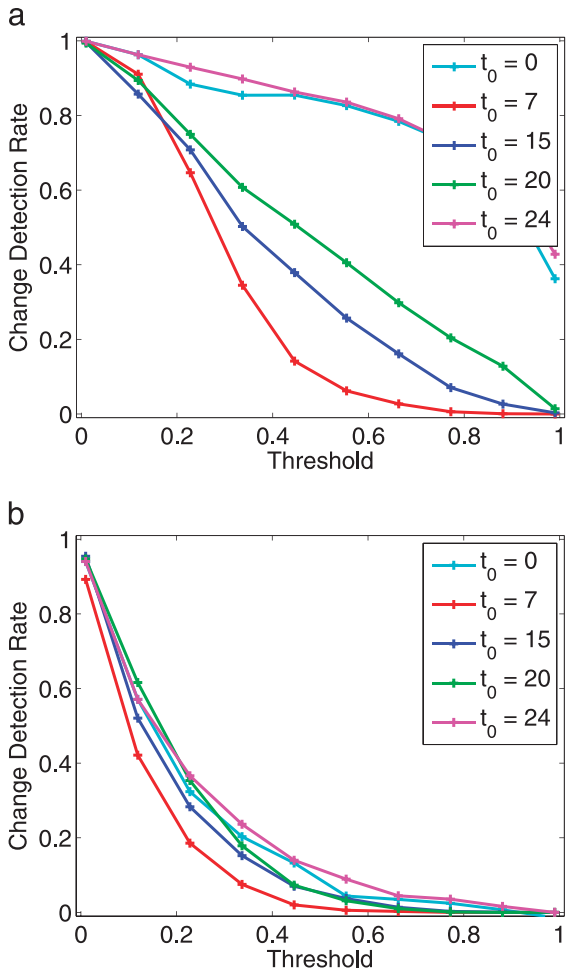


**Figure 15.**
The absolute recognition rates for both SCFG and HMMs are shown in (a) and (b), respectively. The best performance of the SCFG change point detection algorithm is when all target trajectories are entirely simulated under either the normal regime or the abnormal regime.

increases the size of the grammar and makes the rules less intuitive [26].

## EXAMPLE OF SCFG PATTERN GENERATION

To provide more intuition, we compare and contrast SCFGs to HMMs. Consider the grammar shown in Figure 15a. This is a simple grammar generating palindromes over a binary-valued alphabet $\mathcal{V} = \{a,b\}$. The SCFG-generation process starts with a special start symbol $S$. Initially, only nonterminal $S$ is present in the parse tree shown in Figure 15a. We then sample one of the rules from the conditional distribution over all alternative rewrite rules of the start symbol $S$. The next generation of the tree is created by replacing the nonterminal under expansion (the start symbol $S$ in this case) with the right-hand side of the sampled rewrite rule. In the example, rule $r_1$ was chosen with probability $p_1$ and the start symbol $S$ is replaced by the right-hand side "$a\ S\ a$" of rule $r_1$. The procedure is repeated in each generation by leftmost derivation in which only the leftmost nonterminal in the current symbol stack is rewritten with the right side of a chosen production rule. The resultant parse tree $\Psi$ is depicted on the right in Figure 15a. At the end of the derivation process, no more nonterminal symbols exist and the symbol stack only contains clean terminal symbols. The noisy observation process is represented by associating a discrete probability mass function $C(i,j) = \mathbf{P}\{v_j|v_i\}$ to each clean terminal $v_i$. The right bottom part of Figure 15a depicts the noisy observation process. The hat accent is used to differentiate noisy terminals $\hat{a}$ from clean terminals $a$.

An HMM attempting to generate palindromes is also shown in Figure 15b. Markov models cannot exclusively generate palindromes [23]; hence, the model in Figure 15b forms a poor generative model for palindromes. Nevertheless, it is instruc-

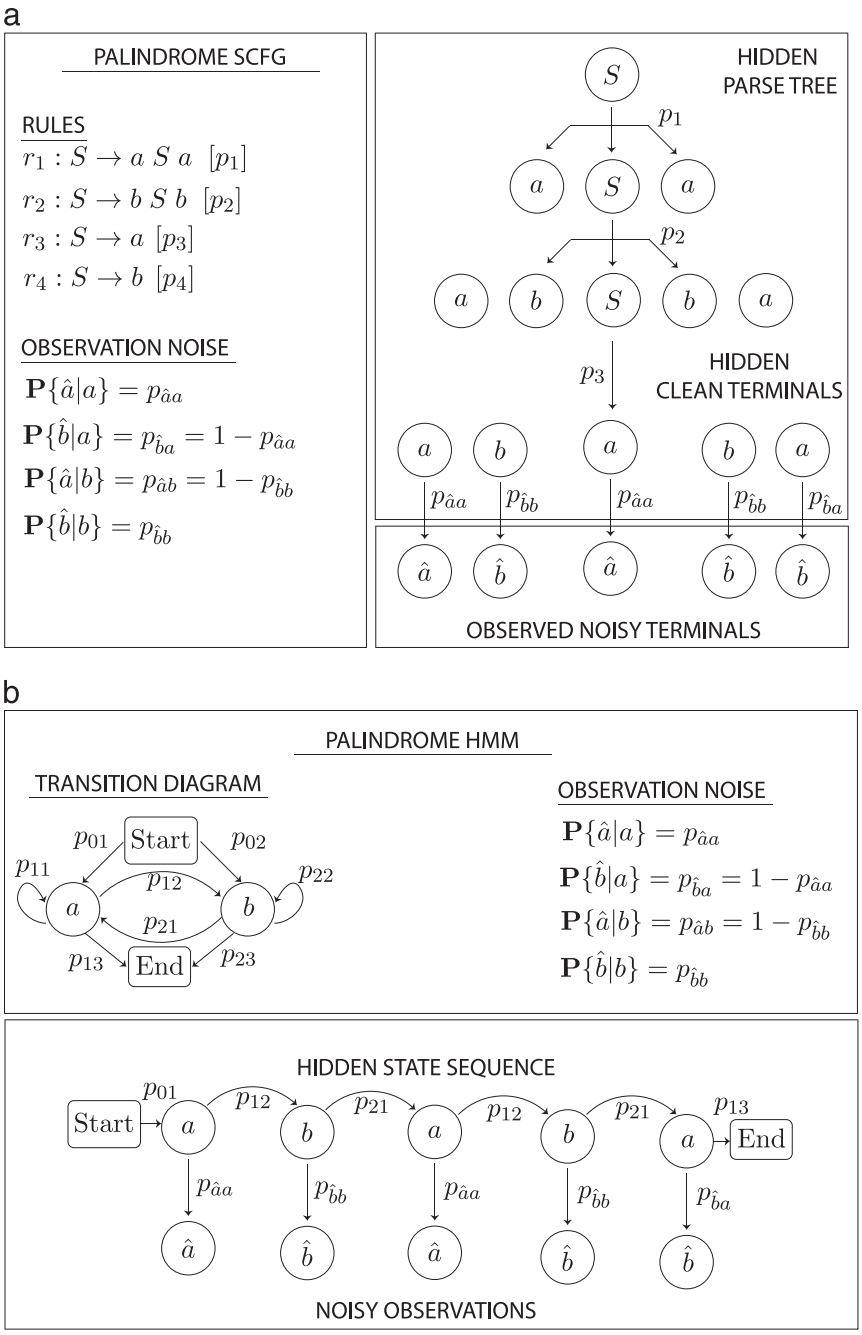


**Figure 16.**
A depiction of the sequence-generation process of a four-state HMM with observations coming from a binary alphabet. (a) The state transitions form a linear directed graph with no inherent branching. (b) The analogous SCFG model of a palindrome. We observe that the number of parameters in the grammar rules is fewer than the number in the HMM case. Moreover, the sequence generation is a branching process with a characteristic inside–outside structure represented using self-embedding rules. The noisy observations are differentiated from the clean terminals by a hat accent.

tive to compare the sequence-generation process of an HMM and an SCFG. We observe that while the HMM state sequence forms a linear directed graph, the SCFG parse tree exhibits a branching structure representing self-embedding rules of the form $S \rightarrow a\ S\ a$. These rules are the basis for the ability of SCFGs to model long-term dependencies in a symbol sequence. SCFGs have been studied in probability theory under the umbrella of random branching processes. In particular, they form a special class of branching processes called multitype Galton-Watson branching processes [27]. From the comparison in Figure 15, it can also be observed that the number of parameters in the HMM is larger than the number in the analogous SCFG model. The predictive entropy of an SCFG has been shown in [28] to be greater than a comparative HMM with a similar number of parameters.

## INFERENCE USING SCFGS

In this article, we are mainly concerned with (a) computing trajectory likelihoods $\mathbf{P}\{\hat{a}_{0:T}; G^{\text{SCFG}}\}$ and (b) recovering the most likely clean terminal sequence $\arg\max_{a_{0:T}} \mathbf{P}\{a_{0:T} \mid \hat{a}_{0:T}; G^{\text{SCFG}}\}$ from an observed sentence $\hat{a}_{0:T}$ given an SCFG model $G^{\text{SCFG}}$. The model likelihoods are used to classify anomalous patterns as formulated in (3) and in the change detection problem in the section with the illustrative example. The most likely parse tree estimation is used to recover the clean terminal sequence originally generated by the SCFG. As described in the introduction, the Earley-Stolcke parser is used in this article due to its ability to deal with unrestricted context-free rules. Moreover, it is capable of generating likelihoods for partial sentences $\hat{a}_{0:t}$ due to its incremental left-to-right operation.

The Earley-Stolcke parser builds leftmost derivations of a string by keeping track of all possible derivations that are consistent with the observations $\hat{a}_{0:T}$ up to a certain epoch $t$. As more observations are revealed, the set of possible derivations (each of which corresponds to a partial parse tree) can either expand as new choices are introduced or shrink as a result of resolved ambiguities. As each observation $\hat{a}_t$ is received by the parser, a set of states $u_t = \{u_t^n, n = 1, 2, \ldots\}$ is created that represents all derivations that can explain the observations $\hat{a}_{0:t}$.

An Earley state is the basic control structure used in the operations of the Earley-Stolcke parser. Each state $u_t^n \in u_t$ represents a rewrite rule $r_m \in \mathcal{R}$ such that the observation $\hat{a}_t$ is derived from its right-hand side. An Earley state is denoted by ${}_{t'}^{t}X \rightarrow \lambda \cdot Y\mu[\alpha, \gamma]$. The uppercase letters $X$ and $Y$ are nonterminals, and $\lambda$ and $\mu$ are substrings of nonterminals and terminals. The index $t$ represents the current epoch. The dot "·" demarcates the portion of the right-hand side that has already been recognized by the parser. The index $t'$ is a backpointer to

the epoch at which the parser rewrote nonterminal $X$ using an instance of production rule $r_m$. Each state is a link in an incomplete chain of rule choices that could have generated the observed sequence $\hat{a}_{0:t}$.

Each state is also associated with a forward probability $\alpha$ and an inner probability $\gamma$. The forward probability $\alpha(u_t^n)$ of a state $u_t^n$ represents the probability of the grammar $G^{\text{SCFG}}$ deriving the observations $\hat{a}_{0:t}$ while passing through the state $u_t^n = {}_{t'}^{t} X \to \lambda \cdot Y\mu$ at epoch $t$. This is analogous to the definition of forward probability for the case of an HMM $G^{\text{HMM}}$ [25]. The inner probability $\gamma(u_t^n)$ of a state $u_t^n$ is defined as the probability of a particular rule generating an infix of the sentence $\hat{a}_{t':t}$ represented by the accumulated probabilities of all derivations starting with a state ${}_{t'}^{t} X \to \lambda \cdot Y\mu$ and ending at the state ${}_{t}^{t} X \to \lambda Y \cdot \mu$.

For the purposes of dealing with the start symbol, the Earley-Stolcke parser is initialized with a dummy state ${}_{0}^{0} \to \cdot S[\alpha = 1, \gamma = 1]$. At each epoch, the states in an Earley set $u_t$ are processed in order by performing one of three operations on each state. These operations may add more states to $u_t$ and may 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. Such a sequence of linked states represents different rules choices that could have generated the input string:

1) The *prediction* operation is applied to states when there is a nonterminal to the right of the dot. It causes the addition of one new state to $u_t$ for each alternative production rule of that nonterminal. The dot is placed at the beginning of the production rule in each new state. The backpointer is set to $t$, since the state was created in $u_t$. Thus, the predictor adds to $u_t$ all productions that might generate substrings beginning with $a_{t+1}$. More formally, for a state ${}_{t'}^{t} X \to \lambda \cdot Y\mu$ in the state set $u_t$, the predictor adds a new predicted state ${}_{t}^{t} Y \to \cdot v$ for each of the alternative production rules $(Y \to v) \in \mathcal{R}$. A link is thus created between these states.

2) The *scanning* operation is applicable only when there is a terminal $a$ to the right of the dot. The scanner compares that symbol $a$ with the observation $\hat{a}_{t+1}$, 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 the terminal symbol has been scanned. If ${}_{t'}^{t} X \to \lambda \cdot a\mu$ exists and $\mathbf{P}\{\hat{a}_{t+1} \mid a\} > 0$, the scanning operation adds a new scanned state ${}_{t}^{t+1} X \to \lambda a \cdot \mu$ to state set $u_{t+1}$. The symbol likelihood $\mathbf{P}\{\hat{a}_{t+1} \mid a\} > 0$ allows us to incorporate a noisy observation process into SCFG generation.

3) The *completion* operation is applicable to a state if its dot is at the end (${}_{t'}^{t} Y \to v.$) of its production rule. Such a state is called a complete state. For every complete state, the parser backtracks to the state set $u_{t'}$ indicated by the pointer in the complete state and adds all states from $u_{t'}$ to $u_t$ that have $Y$ (the nonterminal corresponding to that production) to the right of the dot. It moves the dot over $Y$ in such states. Intuitively, $u_{t'}$ is the state set where $Y$ was first expanded. The parser has seen evidence $\hat{a}_{t':t}$ that $Y \twoheadrightarrow v$ was used, so we move the dot over the $Y$ in these states to show that its terminal yield been successfully scanned. A completion operation adds a new state ${}_{t''}^{t} X \to \lambda Y \cdot \mu$ (called a completed state) using ${}_{t''}^{t'} X \to \lambda \cdot Y\mu$ and ${}_{t'}^{t} Y \to v.$, where $t'' \le t' \le t$.

The Earley-Stolcke parser continues in this manner until all observation $\hat{a}_{0:T}$ symbols have been scanned. If the final state set $u_T$ contains the state ${}_{0}^{0} S' \to S\cdot$, then the algorithm terminates successfully. It represents a successful parse of the sentence $a_1, \ldots, a_t, \ldots, a_T$. The recursive updates of the forward probability $\gamma$ and inner probability $\gamma$ for each of the state operations are summarized in Algorithm 1.

To deal with underflow issues, state-independent scaling must be applied at every epoch. Define the prefix probability $\varsigma_t = \mathbf{P}\{\hat{a}_{0:t} \mid G^{\text{SCFG}}\}$ as the probability of the grammar $G^{\text{SCFG}}$ generating the observations $\hat{a}_0, \ldots, \hat{a}_t$ as the prefix of a complete sentence $\hat{a}_0, \ldots, \hat{a}_T$. Philosophically, this involves a summation over all possible suffixes $\beta \in (\mathcal{X} \cup \mathcal{V})^*$. However, in [29], it is shown that this is equivalent to the sum of the forward probabilities $\alpha$ of all scanned states in the state set $u_t$ such that

$$\varsigma_t = \sum_{\beta \in (\mathcal{X} \cup \mathcal{V})^*} \mathbf{P}\{\hat{a}_0, \ldots, \hat{a}_t, \beta\} = \sum_{n \in u_t} \alpha({}_{t'}^{t} X \to \lambda a \cdot \mu), \tag{11}$$

where $n$ is a scanned state. An appropriate choice for the scaling factor $c_t$ in each state set is to use the inverse of the one-step prediction probability defined as

$$\varsigma_{t|t-1} = \mathbf{P}\{\hat{a}_t \mid \hat{a}_{0:t-1}\} = \frac{\mathbf{P}\{\hat{a}_0, \ldots, \hat{a}_t\}}{\mathbf{P}\{\hat{a}_0, \ldots, \hat{a}_{t-1}\}} = \frac{\varsigma_t}{\varsigma_{t-1}}. \tag{12}$$

The scaling factor $c_0 = 1/\varsigma_0$ is initialized using the prefix probability of the first observation $\hat{a}_0$. The scaling factor at all other epochs $t = 1, \ldots, T$ is computed as $c_t = 1/\varsigma_{t|t-1}$. The forward $\alpha$ and $\gamma$ probabilities of all scanned states are multiplied by the scaling factor to prevent underflow issues. The likelihood $L_t = \mathbf{P}\{\hat{a}_{0:t} \mid G^{\text{SCFG}}\}$ of the observation sequence $\hat{a}_{0:t}$ can be computed from the sequence of scaling factors $c_0, \ldots, c_t$ as

$$L_t = \frac{1}{\prod_{t'=0}^{t} c_{t'}}. \tag{13}$$

An algorithmic description for one cycle of the Earley-Stolcke parser operation is given in Algorithm 1. The notation + = on lines 13, 14, 17, and 18 of Algorithm 1 denotes an accumulated sum over all repeated states in the state set being processed. $R_L(Z, Y\}$ and $R_U(Z, Y)$ are precomputed $|\mathcal{X}| \times |\mathcal{X}|$ matrices representing collapsed factors to account for looping production rules into a single step with modified probability updates. Further details can be found in [29].

Algorithm 1.
**Earley-Stolcke Parser**

1: **function** EARLEYSTOLCKEPARSER $(u_{0:t-1}, \hat{a}_t)$
Scanning
2: **for** ${}_{t'}^{t-1} X \to \lambda \cdot a\mu[\alpha, \gamma] \in u_{t-1}$ **do**
3: Add ${}_{t'}^{t} X \to \lambda a \cdot \mu[\alpha, \gamma]$ if $\mathbf{P}\{\hat{a}_t \mid a\} > 0$.
4: $\alpha' = \alpha \mathbf{P}\{\hat{a}_t \mid a\}$

5: $\gamma' = \gamma \mathbf{P}\{\hat{a}_t \mid a\}$

6: $\varsigma_t = \sum_{n \in u_t} \alpha({}^{t}_{t'}X \rightarrow \lambda a \cdot \mu)$

7: Compute $c_t = \frac{1}{\varsigma_{t|t-1}}$ by computing $\varsigma_{t|t-1}$ from (12)

8: Scale $\alpha$, $\gamma$ for all scanned states using $c_t$

Completion

9: **for** ${}^{t}_{t'}Y \rightarrow v \cdot [\alpha'', \gamma''] \in u_t$ **do**

10: **for** ${}^{t'}_{t''}X \rightarrow \lambda \cdot Z\mu[\alpha, \gamma] \in u_{t'}$ **do**

11: **if** $R_U(Z, Y) \neq 0$ **then**

12: Add ${}^{t}_{t''}X \rightarrow \lambda Z \cdot \mu[\alpha', \gamma']$

13: $\alpha' \mathrel{+}= \alpha\gamma'' \, R_U(Z, Y)$

14: $\gamma' \mathrel{+}= \gamma\gamma'' \, R_U(Z, Y)$

Prediction

15: **for** ${}^{t}_{t'}X \rightarrow \lambda \cdot Z\mu[\alpha, \gamma] \in u_t$ **do**

16: Add ${}^{t}_{t}Y \rightarrow \cdot v[\alpha', \gamma']$ if $R_L(Z, Y) \neq 0$.

17: $\alpha' \mathrel{+}= \alpha R_L(Z, Y)\mathbf{P}\{Y \rightarrow v\}$

18: $\gamma' \mathrel{+}= \mathbf{P}\{Y \rightarrow v\}$

19: **return** $u_t$, $c_t$

In addition to sequence likelihoods $L_t$, we are interested in recovering the sequence of clean tracklets $a_{0:T}$. For an SCFG model, this involves computing the Viterbi parse of the observations $\hat{a}_{0:T}$. A Viterbi parse recovers the sequence of rule choices (or the parse tree $\Psi^*$) that assigns maximum probability to $\hat{a}_{0:T}$ among all possible parses. The same sequence of operations as described in Algorithm 1 can be used to recover the Viterbi parse. However, each state $u_t^n$ also keeps tracks of its Viterbi path probability $\phi$. The Viterbi probability $\phi$ is propagated in the same manner as the inner probability $\gamma$. During completion, the accumulated sum + = is replaced by a maximum over all products that contribute to the completed state. The complete state ${}^{t}_{t'}Y \rightarrow v.$ at the current epoch $t$ associated with the maximum is recorded as the predecessor of ${}^{t}_{t''}X \rightarrow \lambda Z \cdot \mu$. The Viterbi probability update also does not use the collapsing loop factor $R_U(Z, Y)$, because loops can only lower the probability of a derivation. Once the final state ${}^{0}_{0}S' \rightarrow S.$ is accepted by the parser, a recursive procedure can backtrack over the Earley state sets to recover the Viterbi parse $\Psi^*$ [29]. The leaf nodes of the parse tree can then be traversed to recover the clean terminals $a_{0:T}$.

## EXAMPLE OF PARSER OPERATION

The parser operations are illustrated with the use of a simple yet nontrivial example utilizing the palindrome grammar in Figure 15a. The grammar has four rules with the rule probabilities $p_1 = 0.4$, $p_2 = 0.2$, $p_3 = 0.1$, and $p_4 = 0.3$. Suppose that we are presented with the query pattern *a b a*. The classifica-

**Table 1.**

**Earley-Stolcke Operations on the Pattern {*a b a*} Using the Grammar Model in Figure 15**

| Operations | State | Attributes |
|---|---|---|
| State set $t = 0$ | | |
| *Predicted* | ${}^{0}_{0} \rightarrow \cdot S$ | $\alpha = 1.0$, $\gamma = 1.0$ |
| | ${}^{0}_{0}S \rightarrow \cdot aSa$ | $\alpha = 0.4$, $\gamma = 0.4$ |
| | ${}^{0}_{0}S \rightarrow \cdot bSb$ | $\alpha = 0.2$, $\gamma = 0.2$ |
| | ${}^{0}_{0}S \rightarrow \cdot a$ | $\alpha = 0.1$, $\gamma = 0.1$ |
| | ${}^{0}_{0}S \rightarrow \cdot b$ | $\alpha = 0.3$, $\gamma = 0.3$ |
| State set $t = 1$ | | |
| *Scanned* | ${}^{1}_{0}S \rightarrow a \cdot Sa$ | $\alpha = 0.36$, $\gamma = 0.36$ |
| | ${}^{1}_{0}S \rightarrow b \cdot Sb$ | $\alpha = 0.02$, $\gamma = 0.02$ |
| | ${}^{1}_{0}S \rightarrow a \cdot$ | $\alpha = 0.09$, $\gamma = 0.09$ |
| | ${}^{1}_{0}S \rightarrow b \cdot$ | $\alpha = 0.03$, $\gamma = 0.03$ |
| *Completed* | ${}^{1}_{0} \rightarrow \cdot S$ | $\alpha = 0.12$, $\gamma = 0.12$ |
| *Predicted* | ${}^{1}_{1}S \rightarrow a \cdot Sa$ | $\alpha = 0.152$, $\gamma = 0.4$ |
| | ${}^{1}_{1}S \rightarrow \cdot bSb$ | $\alpha = 0.076$, $\gamma = 0.2$ |
| | ${}^{1}_{1}S \rightarrow \cdot a$ | $\alpha = 0.038$, $\gamma = 0.1$ |
| | ${}^{1}_{1}S \rightarrow \cdot b$ | $\alpha = 0.114$, $\gamma = 0.3$ |
| State set $t = 2$ | | |
| *Scanned* | ${}^{2}_{1}S \rightarrow a \cdot Sa$ | $\alpha = 0.0152$, $\gamma = 0.04$ |
| | ${}^{2}_{1}S \rightarrow a \cdot Sb$ | $\alpha = 0.0684$, $\gamma = 0.18$ |
| | ${}^{2}_{1}S \rightarrow a \cdot$ | $\alpha = 0.0038$, $\gamma = 0.01$ |
| | ${}^{2}_{1}S \rightarrow b \cdot$ | $\alpha = 0.1026$, $\gamma = 0.27$ |
| *Completed* | ${}^{2}_{0} \rightarrow \cdot S$ | $\alpha = 0.1008$, $\gamma = 0.1008$ |
| | ${}^{2}_{0}S \rightarrow b \cdot Sb$ | $\alpha = 0.0056$, $\gamma = 0.0056$ |
| *Predicted* | ${}^{2}_{2}S \rightarrow \cdot aSa$ | $\alpha = 0.03344$, $\gamma = 0.4$ |
| | ${}^{2}_{2}S \rightarrow \cdot bSb$ | $\alpha = 0.01672$, $\gamma = 0.2$ |
| | ${}^{2}_{2}S \rightarrow \cdot a$ | $\alpha = 0.00836$, $\gamma = 0.1$ |
| | ${}^{2}_{2}S \rightarrow \cdot b$ | $\alpha = 0.02508$, $\gamma = 0.3$ |
| State set $t = 3$ | | |
| *Scanned* | ${}^{3}_{0}S \rightarrow aSa \cdot$ | $\alpha = 0.0.09072$, $\gamma = 0.09072$ |
| | ${}^{3}_{0}S \rightarrow bSb \cdot$ | $\alpha = 0.00056$, $\gamma = 0.00056$ |
| | ${}^{3}_{0}S \rightarrow a \cdot Sa$ | $\alpha = 0.030096$, $\gamma = 0.36$ |
| | ${}^{3}_{2}S \rightarrow b \cdot Sb$ | $\alpha = 0.001672$, $\gamma = 0.02$ |
| | ${}^{3}_{2}S \rightarrow a \cdot$ | $\alpha = 0.007524$, $\gamma = 0.09$ |
| | ${}^{3}_{2}S \rightarrow b \cdot$ | $\alpha = 0.002508$, $\gamma = 0.03$ |
| *Completed* | ${}^{3}_{1} \rightarrow aS \cdot a$ | $\alpha = 0.001824$, $\gamma = 0.0048$ |
| | ${}^{3}_{1}S \rightarrow bS \cdot b$ | $\alpha = 0.008208$, $\gamma = 0.0216$ |
| | ${}^{3}_{1} \rightarrow S$ | $\alpha = 0.09128$, $\gamma = 0.09128$ |

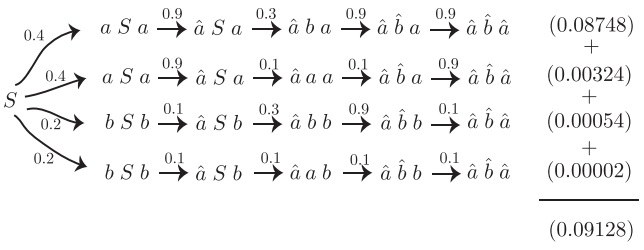


**Figure 17.**
The palindrome grammar of Figure 15a can generate the string *a b a* using four derivation paths. Each path results from a different sequence of applied probability rules or because of nonideal detection probabilities of the terminal symbols. The sentence probability of the input pattern *a b a* is the sum of all possible derivation paths. Due to ambiguity in the grammar-generation process, the sentence probability must account for all possible ways in which a particular terminal sequence can be generated. As can be seen from the parser operations in Table 1, the completed dummy state ${}^{0}_{0}\!\rightarrow S.$ has the correct forward probability of 0.09128 obtained independently by summing the individual probabilities of each alternative derivation path. The numerical superscript over the arrows in the figure represents the conditional probability of choosing that particular rule expansion when a nonterminal is being expanded.

tion problem seeks to evaluate the class conditional probability $\mathbf{P}\{aba|G_{\text{palindrome}}\}$ of the input pattern *aba* given the grammar model $G_{\text{palindrome}}$. In addition, the symbols *a* and *b* are not always reliably detected and the detection probability is parameterized by $\mathbf{P}\{\hat{a}\,|\,a\} = 0.9$ and $\mathbf{P}\{\hat{b}\,|\,b\} = 0.9$, where $\hat{a},\hat{b}$ are used to denote the noisy observations as opposed to the clean terminal symbols *a*, *b*. The Earley-Stolcke parser state sets in each epoch are shown in Table 1. The accompanying code[4] can also be run to see the details of the parsing operations and reproduce the Table 1. In essence, the parser resolves four possible derivations of the grammar that could give rise to the observation sequence *aba*. These paths are shown in Figure 16. The most direct derivation of the input pattern is the top path with the probability of 0.08748. The accumulated sum of all possible derived paths is the class conditional probability of the input pattern. The forward or inner probability of the completed dummy state ${}^{0}_{0}\!\rightarrow S\cdot$ is equivalent to the sentence probability. In this example, the ambiguity arises from the uncertainty in detecting the symbols. However, ambiguity can also arise directly from the structural rules of the grammar. Many examples of such ambiguity can be found in [30]. ◈

## REFERENCES


[1] Bar-Shalom, Y., Willett, P., and Tian, X. *Tracking and Data Fusion: A Handbook of Algorithms.* YBS Publishing, 2011.

[2] Mallick, M., Krishnamurthy, V., and Vo, B. *Integrated Tracking, Classification, and Sensor Management: Theory and Applications.* Wiley, 2012.

[3] Fanaswala, M., and Krishnamurthy, V. Detection of anomalous trajectory patterns in target tracking via stochastic context-free grammars and reciprocal process models. *IEEE Journal of Selected Topics in Signal Processing,* Vol. 7, 1 (2013), 76–90.

[4] Wang, A., Krishnamurthy, V., and Balaji, B. Intent inference and syntactic tracking with GMTI measurements. *IEEE Transactions on Aerospace and Electronic Systems,* Vol. 47, 4 (2011), 2824–2843.

[5] Grenander, U. *Elements of Pattern Theory.* Johns Hopkins University Press, 1996.

[6] Mohandoss, J. A novel approach to detect anomalous behaviour using gesture recognition. In *Proceedings of the 4th International Conference on Signal and Image Processing,* India, 2013, Vol. 222, 113–126.

[7] Takahashi, M., Fujii, M., Naemura, M., and Satoh, S. Human gesture recognition system for TV viewing using time-of-flight camera. *Multimedia Tools and Applications,* Vol. 62, 3 (2013), 761–783.

[8] Derpanis, K. G., Sizintsev, M., Cannons, K. J., and Wildes, R. P. Action spotting and recognition based on a spatiotemporal orientation analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence,* Vol. 35, 3 (2013), 527–540.

[9] Palmer, J. W., Bing, K. F., Sharma, A. C., and Perkins, J. B. Exploitation of radar Doppler signatures for gait analysis. In *Proceedings of the 46th IEEE Conference on Signals, Systems and Computers (ASILOMAR),* 2012, 629–632.

[10] Androulidakis, G., Chatzigiannakis, V., and Papavassiliou, S. Network anomaly detection and classification via opportunistic sampling. *IEEE Network,* Vol. 23, 1 (2009), 6–12.

[11] Liu, S., Ni, L., and Krishnan, R. Fraud detection from taxis' driving behaviors. *IEEE Transactions on Vehicular Technology,* Vol. 63, 1 (Jan. 2014), 464–472.

[12] Pallotta, G., Vespe, M., and Bryan, K. Vessel pattern knowledge discovery from ais data: A framework for anomaly detection and route prediction. *Entropy,* Vol. 15, 6 (2013), 2218–2245.

[13] Katsilieris, F., Braca, P., and Coraluppi, S. Detection of malicious ais position spoofing by exploiting radar information. In *Proceedings of the 16th International Conference on Information Fusion,* July 2013, 1196–1203.

[14] Morris, B. T., and Trivedi, M. M. Trajectory learning for activity understanding: Unsupervised, multilevel, and long-term adaptive approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence,* Vol. 33, 11 (2011), 2287–2301.

[15] Ryoo, M. S., and Aggarwal, J. K. Recognition of composite human activities through context-free grammar based representation. In *Proceedings of the Conference on Computer Vision and Pattern Recognition,* 2006, 1709–1718.

[16] Bobick, A., and Ivanov, Y. Action recognition using probabilistic parsing. In *Proceedings of the Conference on Computer Vision and Pattern Recognition,* June 1998, 196–202.

[17] Bar-Shalom, Y., Daum, F., and Huang, J. The probabilistic data association filter. *IEEE Control Systems Magazine,* Vol. 29, 6 (2009), 82–100.

[18] Blackman, S., and Popoli, R. *Design and Analysis of Modern Tracking Systems,* Artech House Radar Library series. Artech House, 1999.

[19] Mahler, R. P. S. *Statistical Multisource–Multitarget Information Fusion.* Artech House, 2007.

[20] Pulford, G. Taxonomy of multiple target tracking methods. *IEE Proceedings: Radar, Sonar and Navigation,* Vol. 152, 5 (Oct. 2005), 291–304.


[4] https://www.ece.ubc.ca/~vikramk/Site_2/tracking.html.

[21] Krishnamurthy, V., and Fanaswala, M. H. Intent inference via syntactic tracking. *Digital Signal Processing,* Vol. 21, 5 (Sept. 2011), 648–659.

[22] Ivanov, Y., and Bobick, A. Recognition of visual activities and interactions by stochastic parsing. *IEEE Transactions on Pattern Analysis and Machine Intelligence,* Vol. 22, 8 (Aug. 2000), 852–872.

[23] Aho, A. V., and Ullman, J. D. *The Theory of Parsing, Translation, and Compiling.* Upper Saddle River, NJ: Prentice Hall, 1972.

[24] Hopcroft, J. E., and Ullman, J. D. *Introduction to Automata Theory, Languages, and Computation.* Addison-Wesley, 1979.

[25] Rabiner, L. R. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 1989, 257–286.

[26] Manning, C. D., and Schütze, H. *Foundations of Statistical Natural Language Processing.* MIT Press, 1999, Vol. 999.

[27] Athreya, K., and Ney, P. *Branching Processes.* Dover Publications, 2004.

[28] Lari, K., and Young, S. J. The estimation of stochastic context-free grammars using the inside–outside algorithm. *Computer Speech and Language,* Vol. 4, 1 (1990), 35–56.

[29] Stolcke, A. An efficient probabilistic context-free parsing algorithm that computes prefix probabilities. *Computational Linguistics,* Vol. 21, 2 (1995), 165–201.

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