Syntactic Models for Trajectory Constrained Track-Before-Detect

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

***Abstract*—In this paper, a track before detect approach utilizing trajectory shape constraints is proposed to track dimly lit targets. The shape of the target trajectory is modeled syntactically using stochastic context-free grammar models (SCFG) that arise in natural language processing. The directional vector of the target acceleration modes are used as geometric primitives called tracklets. The tracklets are syntactic sub-units of complex spatial trajectory shapes. Stochastic context-free grammars are a generalization of Markov chains (regular grammars) and can model such complex spatial patterns with long range dependencies. Knowledge about the evolution of the trajectory is used in enhancing the track before detect algorithm. A novel multiple model SCFG particle filter is proposed and numerical results are presented to show significant improvement over conventional jump Markov models in track before detect.**

***Index Terms*—Dimly lit targets, multiple model particle filter, natural language processing, stochastic context-free grammars, track-before-detect, trajectory models.**

I. Introduction

CONVENTIONAL target tracking algorithms employ a detect-then-track approach that performs filtering based on target detection. A target is detected by applying a hard threshold on the sensor measurement utilizing a suitable metric, for example, a constant false alarm ratio. Such a method of first detecting targets and subsequently forming tracks is more efficient in terms of computational complexity. However, in dimly lit (low signal-to-noise ratio) conditions, the background clutter is often at a comparable strength to the target returns and hard thresholding leads to overwhelming spurious detections. A track-before-detect (TBD) approach uses multiple frames of the raw sensor measurements with the objective of avoiding a hard thresholding decision. Consequently, TBD algorithms jointly estimate the existence of the target (detection) as well as track its kinematic state (filtering).

In this paper, we enhance the TBD approach by exploiting target trajectory patterns. Conventionally, maneuvering targets are modeled using jump Markov state space models [1], [2]. However, modeling maneuvers via Markov chains does not facilitate modeling complex spatial trajectories like u-turns, closed trajectories and circling patterns as shown in Fig. 1(b). The main idea of this paper is to model maneuvering targets as a stochastic context-free grammar (SCFG) modulated state space model. The proposed track-before-detect approach exploits higher level information regarding the intent and/or trajectory pattern of the target. We show that the resulting syntactic TBD algorithms can successfully detect and track targets at significantly lower SNR than conventional TBD algorithms.

A. Why Use Stochastic Context-Free Grammars for Trajectory Modeling?

An SCFG is defined formally in Section III-A and is presented in analogy to a hidden Markov model (HMM) to aid signal processing readers who are unfamiliar with this formalism. SCFGs have been studied extensively in natural language processing and are a generalization of Markov chains (as shown in Fig. 1(a)). The expressive power of SCFGs enables scale-invariant modeling of complex spatial trajectories with variable-order long-range dependencies that naturally arise when humans (or human operated objects) move in an environment [3]. Such patterns cannot be generated by Markov chains (this is proved in computer science using "pumping lemmas" [4]). In Fig. 1(a), we show the Chomsky hierarchy of grammatical models which depicts that SCFGs are a more expressive generalization of Markov models.

Inference using SCFGs can also be performed in polynomial time using efficient algorithms like the inside-outside algorithm [5] and the Earley-Stolcke parser [6]. This makes SCFGs practically relevant unlike more general context-sensitive grammars (see Fig. 1(a)) where inference is known to be NP-complete [7]. Additionally, SCFGs have a compact formal representation in terms of production rules that allow human intuition to be easily codified into high-level rules. This, in turn, permits the design of high-level Bayesian signal processing algorithms to detect trajectories of interest. The ability for the designer to encode domain expertise into a knowledge base is important because the lack of sufficient field data is a limiting factor in training anomaly recognition systems. From an information-theoretic perspective, it is shown in [5] that the predictive power of SCFGs, as measured by its predictive entropy, is greater than that of an analogous hidden Markov model with the same number of free parameters. These characteristics make SCFGs an ideal trajectory modeling tool.

Manuscript received February 20, 2014; revised June 25, 2014; accepted September 10, 2014. Date of publication September 23, 2014; date of current version November 04, 2014. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Amir Asif. A portion of this paper was presented at CAMSAP 2013.

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/TSP.2014.2360142

B. Why Consider Trajectory-Constrained TBD?

Track-before-detect plays an important role for situational awareness in low SNR conditions. The main idea of our approach is for the human operator to utilize meta-level information about suspicious trajectories to make the TBD algorithm more sensitive in dimly lit environments. Increased sensitivity, in this scenario, is defined as an increase in the tracking and detection performance at low signal-to-noise ratios when the target exhibits a suspicious trajectory. Consider the scenario depicted in Fig. 2(a) that can occur in dismount-moving target indicator (DMTI) radar applications for highly mobile targets. It depicts a hypothetical situation in which a human guard (or vehicle) is patrolling the perimeter of a compound in a circling pattern. Humans are capable of *turning on the spot* and instantaneously reversing their motion which represents a significant deviation from a constant velocity motion model. Such trajectories are also not well modeled by constant turn models. A standard track-before-detect particle filter with two modes of operation (a constant velocity and a constant turn mode) diverges and is unable to reliably detect and track a simulated rectangular trajectory at an SNR of 2 dB as shown in Fig. 2(b). However, an SCFG-switching multiple model approach can accurately detect and track the target as shown in Section V. As the target moves in the surveillance environment, it generates a sequence of modes $q_1, \ldots, q_k$ which constitute a trajectory $\lambda \in \mathcal{L}$. The set $\mathcal{L}$ denotes a class of scale-invariant, rotation-invariant trajectories with recursive embedding that can model shapes such as lines, arcs, m-rectangles and closed-loops as shown in Fig. 1(b). In this paper, we will choose $\mathcal{L}$ to be a set of trajectories that can be generated by a stochastic context-free grammar. In the sequel, we show that modeling the trajectory followed by the target using directional modes allows our syntactic TBD particle filter to operate better than conventional TBD algorithms (without trajectory constraints) in lower SNR conditions. Even a small SNR difference of 3 dB can amount to a target being detected with high fidelity ($\mathbf{P}_d = 0.99$) at $\sim$16 dB) or it being marginally detected ($\mathbf{P}_d = 0.5$ at 13 dB) [8].

We approach the multi-frame TBD problem as a tracking problem involving a high-dimensional sensor measurement that is also a highly non-linear function of the state contaminated with non-Gaussian noise. As depicted in Fig. 4, our syntactic TBD approach utilizes a hybrid particle filter to propagate a mixed continuous-discrete state given the entire image sequence of radar measurements without performing any hard thresholding at each measurement instant. The output of the TBD algorithm is a posterior filtering density from which the target can be simultaneously detected and tracked. In addition, the mode estimates can be used as a trajectory visualization tool. The mode estimates can also be used in a feedback loop to aid a higher level decision-layer in situational awareness type applications.

C. Literature Survey

A brief survey of the literature on the two major components of this paper a) non-linear filtering and b) trajectory pattern recognition are presented in this section.

On Non-Linear Filtering in TBD: The main difficulty in the TBD problem is the highly non-linear relationship between the sensor measurement image and the target state. A rich history of non-linear filtering exists in the TBD literature starting from the use of an extended Kalman filter in [9] and point mass (HMM) filter approximations in [10], [11]. An efficient alternative to state-space discretization is to use particle filters to solve the non-linear estimation problem, see [12], [13]. The histogram probabilistic multi-hypothesis tracking (H-PMHT) algorithm [14] is an efficient multi-target alternative to TBD as it does not threshold the sensor observations and also does not use likelihood ratios. A random finite set approach is taken in [15] for multiple targets which uses the probability hypothesis density of a multi-Bernoulli random finite set. This approach has been shown to be equivalent to a particle filter TBD approach in [16]. Finally, the recent work [17] addresses the computational complexity of TBD algorithms by using two detection thresholds to first produce a small set of detections and then exploiting space-time correlations. A textbook treatment of hybrid (mixed continuous and discrete state variables) state estimation techniques can be found in [1]. The target dynamical model used in this paper is similar to that used in road-constrained target tracking [18], [19] which is philosophically similar to the notion of trajectory constraints.

On Trajectory Pattern Recognition: The modeling of complex spatial trajectories considered in this paper stems from research in the syntactic pattern recognition community [20]. Trajectory modeling has been widely studied in the action recognition community. The study in [21] uses a two-tier approach towards goal recognition in a wireless LAN scenario. A hidden Markov model (HMM) is used as a feature-detector in the lower tier while a higher-order HMM is used to enforce syntactic structure in the upper tier. However, a standard HMM suffers from an exponential self-transition probability when the same state is visited for a long duration. As a result, repeatedly visiting the same state exponentially decreases the model likelihood. Consequently, a non-stationary hidden semi-Markov model is proposed in [22] to account for self-transitions. SCFGs can also effectively model such self-transitions using its self-embedding property (explained in Section III-A).

The grammar modeling approach in this paper is also related to the approach taken in [23] 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 the detection of certain anomalous activities like carjacking in parking lots. In this paper, constraints are also enforced albeit on system-theoretic quantities resulting in a convenient initialization for the rule probabilities of an SCFG.

To the best of our knowledge, constraints on the trajectory patterns have not been examined in the context of track-before-detect algorithms in the literature. In contrast to our past work [24], this paper considers a highly non-linear measurement function for which linear approximations like the extended Kalman filter and extensions like the variable-state interacting multiple model (VS-IMM) used in [24] cannot be suitably applied. Moreover, the particle filtering solution applied in this paper is different from the approximations used in [24]. The

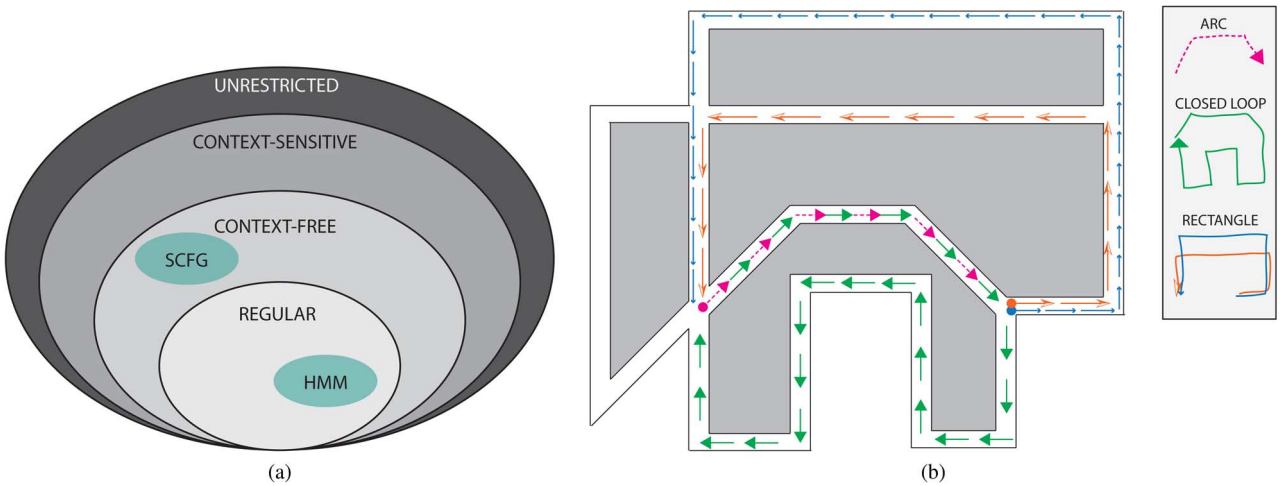

Fig. 1. (a) shows the Chomsky hierarchy of grammars. Polynomial time algorithms for inference are only known for the class of context-free and regular grammars. SCFGs belong to the class of context-free grammars which are more general and expressive than regular grammars (that contain HMMs). (b) shows examples of target trajectories that are scale-invariant and display recursive embedding. This is a conceptual sketch of a road network on which target trajectories are evolving. An urban landscape constrains targets to travel predominantly along certain directions. The rectangular trajectories shown have the same destination but are of different sizes. They can be captured by the same stochastic context free grammar model without a corresponding increase in computational complexity even though they exhibit memory of different orders.

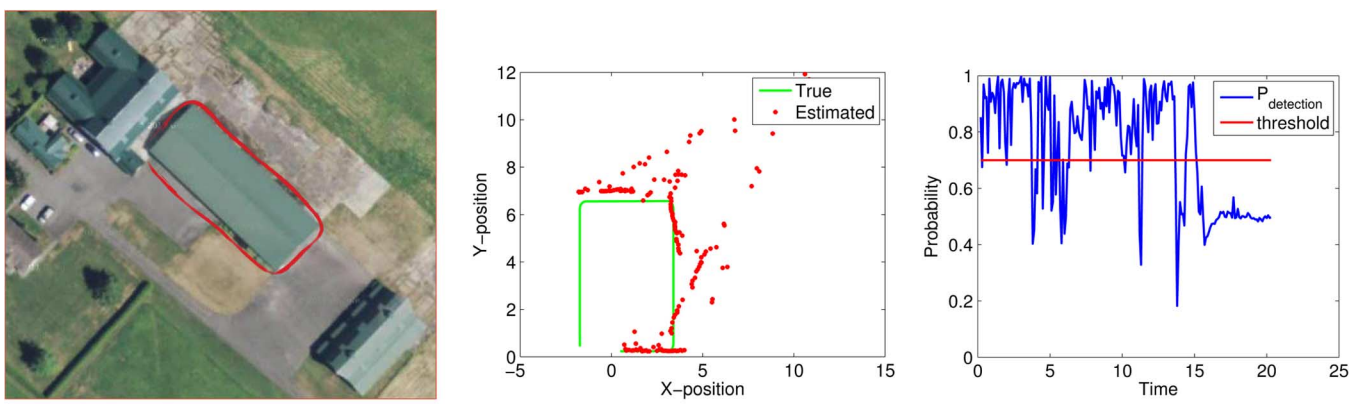

Fig. 2. (a) shows the perimeter surveillance proof-of-concept application. A dismounted target walks around the compound wall in a rectangular trajectory. In (b), a TBD particle filter algorithm with a standard constant velocity model together with a constant turn model is used in a 2 dB SNR scenario. The filter diverges midway through the evolution of the target trajectory. In (c), the probability of detection is shown with the selected threshold. We observe that almost half the time, the target is not even detected reliably. In Section V, we demonstrate the performance increase obtained by using SCFG trajectory models as shown in Fig. 8(a).

more recent work [25] uses similar trajectory models but is significantly different because it completely bypasses the tracker and builds a meta-level inference layer on top of the base-level tracker. This approach was taken to ensure legacy-compatibility with older tracking systems yet providing the prediction and detection capabilities of SCFGs to pick out anomalous trajectories.

The paper is organized as follows. The track-before-detect problem is formulated as an SCFG-driven multiple-model tracking problem. Details about the switching state space model used and the sensor characteristics are provided in Section II. In Section III, a brief review of SCFGs is provided together with grammar models for common trajectories which can be used as building blocks of more complex trajectories. In Section IV, a particle filtering solution for the SCFG-driven TBD problem is presented together with a Rao-Blackwellised version. This solution is then used to present numerical experiments in comparison with a Markov-modulated multiple-model TBD particle filter in Section V. Finally, we present concluding remarks in Section VI.

II. SCFG-Driven Multiple Model Track-Before-Detect

In this section, the track-before-detect problem is formulated as a constrained search within trajectories satisfying syntactic patterns. The trajectory dynamics are modeled using an SCFG-mode sequence, the target state dynamics follow a switching constant velocity model with directional process noise and the sensor characteristics are described assuming a line-of-sight radar sensor measurement.

A. Trajectory-Constrained Model

Consider a target moving in the x-y plane according to a discrete-time dynamic model of the form

$$\mathbf{x}_{k+1} = F\mathbf{x}_k + G\mathbf{v}_k(q_k), \tag{1}$$

where k is the discrete-time index, $q_k \in \mathcal{Q}$ is a mode, $\mathbf{v}_k$ is the state process noise and $\mathbf{x}_k = [x_k, \dot{x}_k, y_k, \dot{y}_k]^\mathsf{T}$ is the state vector comprising of the position and velocity components in the x and y coordinate axes. The transition matrix F and noise gain G are respectively,

$$F = \begin{bmatrix} 1 & 0 & \delta T & 0 \\ 0 & 1 & 0 & \delta T \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \; G = \begin{bmatrix} \frac{\delta T^2}{2} & 0 \\ 0 & \frac{\delta T^2}{2} \\ \delta T & 0 \\ 0 & \delta T \end{bmatrix},$$

where δT is sampling interval. The mode-dependent state process noise $\mathbf{v}_k$ is a white Gaussian process with co-variance matrix

$$Q = \rho_{q_k} \begin{bmatrix} \sigma_a^2 & 0 \\ 0 & \sigma_o^2 \end{bmatrix} \rho_{q_k}^\mathsf{T},$$
$$\text{with } \rho_{q_k} = \begin{bmatrix} \sin q_k & \cos q_k \\ -\cos q_k & \sin q_k \end{bmatrix}, \tag{2}$$

where the superscript $^\mathsf{T}$ denotes the transpose operation, σ_a^2 is the uncertainty along the direction indicated by q_k and σ_o^2 is the uncertainty along the orthogonal direction to q_k. The modes q_k serve to modulate the state process noise $\mathbf{v}_k(q_k)$ and cause it to switch between different variance values. The noise can be uncoupled across each state variable using the equivalent representation $\mathbf{x}_{k+1} = F\mathbf{x}_k + \tilde{\mathbf{v}}_k(q_k)$. The process noise $\tilde{\mathbf{v}}_k(q_k)$ is a zero-mean Gaussian [1], [26] with co-variance matrix

$$\tilde{Q}_k(q_k) = \begin{bmatrix} \frac{\delta T^3}{3} & 0 & \frac{\delta T^2}{2} & 0 \\ 0 & \frac{\delta T^3}{3} & 0 & \frac{\delta T^2}{2} \\ \frac{\delta T^2}{2} & 0 & \delta T & 0 \\ 0 & \frac{\delta T^2}{2} & 0 & \delta T \end{bmatrix} \times diag(Q, Q), \tag{3}$$

where $diag(Q, Q)$ refers to a 4×4 matrix obtained by diagonally concatenating Q from (2) and setting the extra entries to 0. Such a representation results in a non-singular co-variance matrix more amenable towards use in the particle filters presented in Section IV.

The observations $\mathbf{z}_k$ from the radar sensor provide data over a discretized two-dimensional domain consisting of an $M \times N$ grid of resolution bins of side length Δ. In particular, the measurement $\mathbf{z}_k = \{z_k^{m,n} : m = 1, \ldots, M, n = 1, \ldots, N\}$ is an image of measured intensities given by

$$z_k^{m,n} = \begin{cases} h^{m,n}(\mathbf{x}_k) + w_k^{m,n} & e_k = 1 \\ w_k^{m,n} & e_k = 0, \end{cases}$$

where $h^{m,n}(\mathbf{x}_k)$ is the target spread function that represents the contribution of the target intensity to the (m, n)th bin. The target existence e_k is a binary variable representing the absence $(e_k = 0)$ or the presence $(e_k = 1)$ of a target. The target spread function $h^{m,n}(\cdot)$ is modeled by a Gaussian spread

$$h^{m,n}(\mathbf{x}_k) = \frac{\Delta^2 I}{2\pi\sigma_h^2} \exp\left[\frac{-(m\Delta - x_k)^2 - (n\Delta - y_k)^2}{2\sigma_h^2}\right] \tag{4}$$

where I is the constant amplitude of the target and σ_h^2 is the variance of the Gaussian spread function. The measurement noise $w_k^{m,n}$ is assumed to be dominated by Rayleigh distributed clutter. When no target is present, the likelihood of the observation $z_k^{m,n}$ at bin (m, n), follows a Rayleigh distribution

$$p\left(z_k^{m,n} | \mathbf{x}_k, e_k = 0\right) = \frac{2z_k^{m,n}}{P} \exp \frac{-\left(z_k^{m,n}\right)^2}{P},$$

where P is the average Rayleigh clutter power determined by calculating the mean power across the measurement frame. When a target is present, the likelihood of the observation follows a Ricean distribution

$$p\left(z_k^{m,n} | \mathbf{x}_k, e_k = 1\right) = \frac{2z_k^{m,n}}{P} \times \exp\left(\frac{h^{m,n}(\mathbf{x}_k)^2 - \left(z_k^{m,n}\right)^2}{P}\right) I_0\left(\frac{2z_k^{m,n} h^{m,n}(\mathbf{x}_k)}{P}\right),$$

where $I_0(\cdot)$ is the modified Bessel function of zero order. The filtering solution in the sequel requires us to introduce the measurement likelihood ratio in a bin (m, n) as

$$l\left(z_k^{m,n} | \mathbf{x}_k\right) = \exp\left(-\frac{h^{m,n}(\mathbf{x}_k)^2}{2P}\right) I_0\left(\frac{z_k^{m,n} h^{m,n}(\mathbf{x}_k)}{P}\right). \tag{5}$$

The target existence e_k is modeled as a two state Markov chain with transition matrix

$$\Pi_e = \begin{bmatrix} 1 - P_{\text{birth}} & P_{\text{birth}} \\ P_{\text{death}} & 1 - P_{\text{death}} \end{bmatrix} \tag{6}$$

When the mode sequence is considered to arise from a Markov chain, the transition matrix of the modes is given by

$$\Pi_q = [\pi_q(k, l)] = \frac{e^{-(\pi - ||k-l| - \pi|)^2}}{\sum_m e^{-(\pi - ||k-m| - \pi|)^2}}, \; k, l, m \in \mathcal{Q}. \tag{7}$$

This transition probability function assigns maximum probability to the same mode such that $k = l$ and assigns an exponentially decaying probability to neighboring modes. We use $\Pi_q = \mathbf{P}\{q_k | q_{k-1}\}$ to parametrize the Markov chain model generating trajectory sequences $q_{1:k}$ (as an alternative to SCFG models).

B. Estimation Objective

We seek to obtain filtered state, mode and target existence estimates $\mathbf{E}\{\mathbf{x}_k, q_k, e_k | \mathbf{z}_{1:k}\}$ given the measurement sequence of sensor images $\mathbf{z}_{1:k}$. For each instant k, the presence (or absence) of the target is indicated by the random variable e_k which is modeled as a two-state Markov chain. The target is assumed to follow a trajectory which switches between constant velocity models in certain acceleration directions $q_k \in \mathcal{Q} = \{0, \frac{\pi}{4}, \frac{\pi}{2}, \frac{3\pi}{4}, \pi, \frac{5\pi}{4}, \frac{3\pi}{2}, \frac{7\pi}{4}\}$. These acceleration directions are shown in Fig. 3. A sequence of target modes (or maneuvers) is defined as a trajectory $q_{1:k}$ which is modeled using SCFGs. The precise formulation of the SCFG trajectory models is presented in Section III. As shown in Fig. 4, a particle filter is combined together with the Earley-Stolcke parser to perform multi-model state estimation for a jump SCFG non-linear state space model.

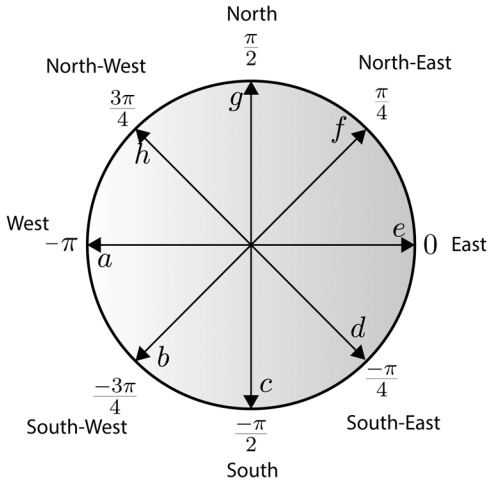

Fig. 3. The modes of operation of a target which are represented by directional motion models in the 8 quantized radial directions. We refer to 'North-East', 'South' etc. as cardinal directions in the paper.

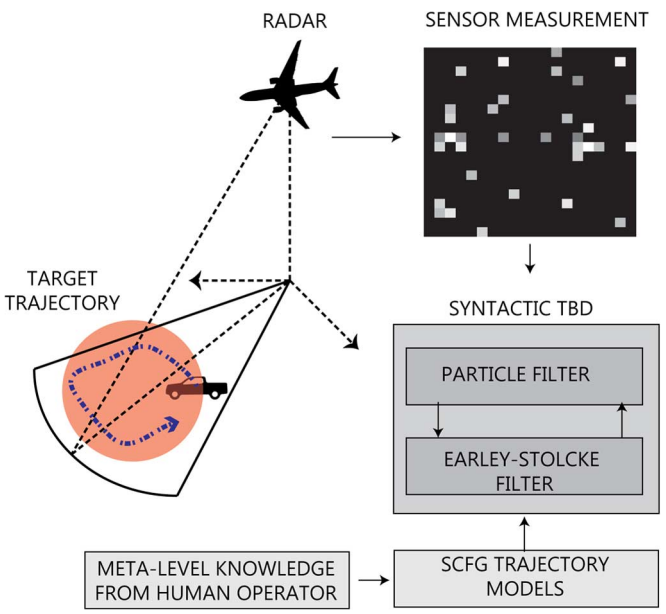

Fig. 4. The syntactic TBD system architecture. An image-based sensor measurement from a radar or FLIR (forward-looking infra-red) can form the input to our system. A knowledge base of radar operator intuition can be used to build SCFG models which help the particle filter to operate in low SNR conditions when certain trajectories are exhibited by a target.

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

Suppose we are interested in tracking dimly lit targets following a specific trajectory pattern such as a circling behavior shown in Fig. 2(a). How can this information be encoded into a tractable model and be used with a TBD algorithm? In this section, we define SCFGs and present SCFG models for geometric trajectory patterns like lines, arcs, m-rectangles and closed-loops. We also provide further insight into SCFGs by contrasting them with hidden Markov models. Finally, we describe how inference using SCFG models can be used within the trajectory-constrained framework. The development and notation in this section follows our previous work in [25].

A. Review of Stochastic Context-Free Grammars

In this section, we provide a structural description of stochastic context-free grammar (SCFG) models. A pedagogical treatment relating to signal processing applications of SCFGs can be found in [20]. A context-free grammar $\mathcal{L}_{\mathrm{CFG}}$ is a 4-tuple $(\mathcal{N}, \mathcal{V}, S, \mathcal{R})$, 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 $\mathcal{R}$ is a finite set of production rules r_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 $\rightarrow$ 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 $(\mathcal{L}_{\mathrm{CFG}}, p)$, where $p : \mathcal{R} \rightarrow [0, 1]$ is a probability function over the production rules $(A \rightarrow \alpha) \in \mathcal{R}$ 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.

B. SCFG Models for Anomalous Trajectories

In this section, various trajectories of interest are modeled using stochastic context-free grammars. Each trajectory pattern considered in this paper has an associated grammar model $\mathcal{L}$ with a common set of terminals $\mathcal{V} = \mathcal{Q}$ that represent the possible modes of operation. They may have different rule spaces $\mathcal{R}$ 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 rule probabilities are chosen so that certain system-theoretic conditions [25] are satisfied. The models described below have previously been considered in [25]. Only the grammatical descriptions are included to provide a unified description in this paper.

Linear Trajectories: Straight paths are denoted as linear trajectories that are generated by target dynamics obeying local Markov dependency. Linear grammar models are represented using the compact form $\mathcal{L}_{\mathrm{line}} = \{\vec{a}^n\}$ 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 $\vec{a} \in \mathcal{Q}$ representing the target's direction of motion.

Arc-Like Trajectories: Arc-like trajectories have the compact form $\mathcal{L}_{\mathrm{arc}} = \{\vec{a}^n \vec{b}^+ \vec{c}^n\}$ which is characterized by an equal number of movements in opposing directions represented by the unit vectors $\vec{a}$ and $\vec{c}$. 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 Fig. 5(a). The notion of arc-like patterns is used to represent u-turn and open trapezoidal patterns.

Rectangle Trajectories: Rectangular trajectories are of the sentential form $\mathcal{L}_{\mathrm{rectangle}} = \{\vec{a}^n \vec{b}^m \vec{c}^n \vec{d}^m\}$, where the target moves an equal number of times in opposing directions $\vec{a}, \vec{c}$ and $\vec{b}, \vec{d}$. However, it can be shown using a pumping lemma that a complete rectangle cannot be modeled by a context-free grammar [4]. A more expressive formalism called context-sensitive grammars (see Fig. 1(a)) are required. However, there are no known polynomial time algorithms to perform inference

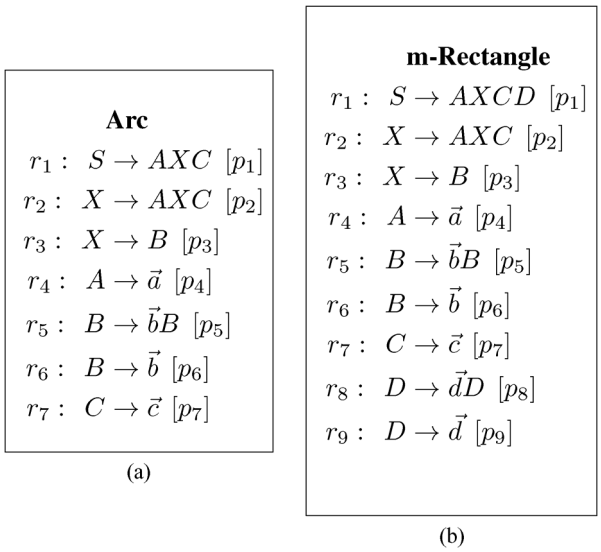

Fig. 5. An arc grammar in (a) and an m-rectangle grammar in (b). Only the production rules are shown. Capital case refers to abstract non-terminals and lower-case refers to terminal modes. A rule re-writing is analogous to a state transition in Markov models. A non-terminal N currently being considered for expansion is replaced by the string on the right hand side of the rule r_m chosen (by sampling the conditional distribution induced by all alternative rules) for expansion. Each rule in the conditional distribution is associated with a rule probability p_m.

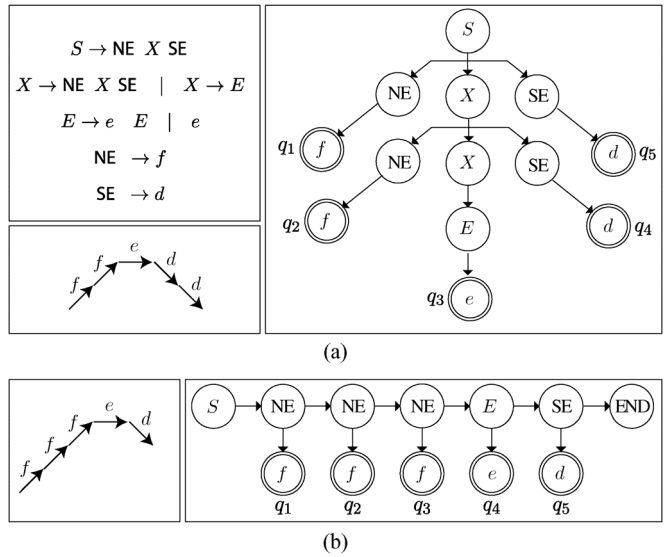

Fig. 6. (a) depicts the generation process of an SCFG as a branching process. The tree on the right is also called a parse tree or a derivation because the sequence is derived by repeated application of the production rules shown in the top left. The | in the production rules denotes an alternate re-writing rule. Circles with double boundaries represent observations while regular circles represent latent states. The derived mode sequence $q_{1:5}$ is an arc with a trapezoidal shape as shown in the bottom left. In (b), the generation process of an HMM is depicted as a linear directed graph. While the SCFG can always ensure equal opposing movements in every sample path due to self-embedding, the HMM cannot ensure this as observed in the sample HMM sequence to the left.

with context-sensitive models. Instead, we consider the modified-rectangle language (with associated grammar shown in Fig. 5(b)) as $\mathcal{L}_{\mathrm{m-rectangle}} = \{\vec{a}^m \vec{b}^+ \vec{c}^m \vec{d}^*\}$. The modified-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 notation $\vec{b}^+$ and $\vec{d}^*$ represent an arbitrary number of movements in the corresponding directions represented by that mode.

SCFG Model Estimation: The models presented in Section III-B happen to have compact forms representing simple geometric shapes. For more complicated trajectory patterns, we advocate the construction of the rule structure by domain experts incorporating intuitive knowledge into the rule-based framework of SCFGs. For example, a radar operator typically sees many anomalous trajectories and can subconsciously codify the evolution of the trajectory into high-level rules. A discussion of other grammar construction techniques is provided in [4]. A learning based approach can also be undertaken to estimate the rules through training data and a typical approach based on Bayesian model merging is presented in [6].

In addition to the syntactic rules comprising a grammar model, we are also required to choose the rule probabilities governing the conditional application of a particular rule in the generation of the trajectory sequence. Traditionally, a maximum likelihood approach can be taken [6] by estimating the rule probabilities that maximize the likelihood of some pre-obtained training data given a candidate grammar model. However, since our models have relatively simple structure, we use the consistency and expected word length constraints from [25] to estimate rule probabilities. Moreover, given that expectation-maximization type algorithms for estimating the rule probabilities of SCFGs are highly prone to getting stuck in local minima, such constraints can be used to find appropriate initial values for the rule probabilities.

C. Comparison Between SCFGs and HMMs

In this section, for the reader's convenience, we provide insight into SCFGs by an analogy with hidden Markov models (HMMs). In order to facilitate our comparison, an example scenario is depicted in Fig. 6 in which we seek to generate the mode sequence $q_{1:5} = \vec{f}\vec{f}\vec{e}\vec{d}\vec{d}$. This mode pattern represents an arc-like trajectory ($\vec{f}^2\vec{e}\vec{d}^2$) with two movements in the 'North-East' direction, one movement in the 'East' direction and two matching movements in the 'South-East' direction. The generative SCFG model for such a trajectory is shown in Fig. 6(a). It has a non-terminal set $\mathcal{N}^{\mathrm{SCFG}}_{\mathrm{example}} = \{S, \mathrm{NE}, X, E, \mathrm{SE}\}$ where NE represents the 'North-East' direction as depicted in Fig. 3. The special symbol X is a self-embedding non-terminal because it can repeatedly call itself while producing an equal number of opposing NE and SE non-terminals. The SCFG terminal set $\mathcal{V}^{\mathrm{SCFG}}_{\mathrm{example}} = \{d, e, f\}$ is comprised of the unit vectors corresponding to the cardinal directions. For example, as depicted in Fig. 3, the terminal corresponding to the 'North-East' cardinal direction is represented by the unit vector $\vec{f}$. An equivalent discrete-observation hidden Markov model is specified by a set of unobserved states $\mathcal{N}^{\mathrm{HMM}}_{\mathrm{example}} = \{S, \mathrm{NE}, E, \mathrm{SE}, \mathrm{END}\}$ that is similar to the non-terminal set of the SCFG. The HMM state set does not have the self-embedding symbol X and has been augmented with a special END state to guarantee finite-length termination. The HMM observation space is equivalent to the terminal set of the SCFG.

In direct analogy to an HMM, the non-terminals of an SCFG are abstract unobserved states while the terminals are observed

symbols. In the context of trajectory modeling, the non-terminals represent structural parts of a shape. For example, in Fig. 6(a), an arc is structurally decomposed as NE, E and SE segments. The terminal symbols correspond to unit movements in a cardinal direction. Fig. 6 shows that while the HMM state sequence evolves as a linear directed graph, the SCFG behaves like a branching process. The transitions of an HMM from one state to the next can be written as regular grammar rules of the form $N_k \rightarrow q_{k+1}N_{k+1}$ which represents a transition from the current state N_k to the next state N_{k+1} while emitting an observation q_{k+1}. This corresponds to the depiction of the HMM evolution as a linear directed graph in Fig. 6(b). However, the SCFG has more complex rules that manifest in a branching process as shown in Fig. 6(a). SCFGs are known to be a special class of multi-type Galton-Watson stochastic branching processes [3], [27]. The defining characteristic of an SCFG is the presence of self-embedding rules like $X \rightarrow \mathrm{NE}\ X\ \mathrm{SE}$ in Fig. 6(a). Such a rule, repeatedly calls itself to generate equal opposing movements that manifests as an unbounded dependency and imparts scale-invariance to SCFG trajectory models.

Analogous to the (hidden) Markov model case, there are several probabilistic queries that can be computed for a symbol sequence generated from an SCFG. (a) We can compute the likelihood of a sequence $\mathbf{P}\{q_1,\ldots,q_K|\mathcal{L}^{\mathrm{SCFG}}\}$ given a certain SCFG model. While the forward algorithm [28] is commonly used for HMMs, the inside algorithm [5] is a generalization that can be used for SCFGs. (b) We can estimate the hidden sequence of rules used in the derivation of a sequence analogous to hidden state sequence estimation in HMMs. The Viterbi algorithm can be used in conjunction with the inside algorithm for this purpose. (c) Finally, we can learn the rule probabilities from a dataset using a maximum-likelihood approach called the inside-outside algorithm [5] which is similar to the forward-backward algorithm used in Baum-Welch [28] re-estimation for HMMs.

D. Inference Using SCFGs

The main quantity of interest for the hybrid state estimation problem in Section II-B is the one-step ahead prediction $\mathbf{P}\{q_k|q_{1:k-1};\mathcal{L}^{\mathrm{SCFG}}\}$ that can be computed from a left-right pass over an observed terminal sequence. Calculation of the one-step ahead prediction requires the notion of the "prefix" probability $\mathbb{P}_k$ of a symbol sequence given an SCFG model $\mathcal{L}^{\mathrm{SCFG}}$. The computation of the prefix probability is represented by

$$\mathbb{P}_k = \sum_{\nu\in(\mathcal{N}\cup\mathcal{V})^*} \mathbf{P}\{q_1,\ldots,q_k,\nu\}, \tag{8}$$

where $(\mathcal{N}\cup\mathcal{V})^*$ denotes all possible arbitrary combinations of non-terminal and terminal symbols. The expression in (8) represents the probability that the sequence $q_{1:k}$ is the prefix of a sentence generated from a SCFG model $\mathcal{L}^{\mathrm{SCFG}}$ and it conceptually requires a summation over all possible suffixes. The one-step prediction utilizes the probabilistic rules of the SCFG model to predict the next mode in the sequence. This quantity is important in describing a suitable transitional density for the particle filter solution in Section IV. The one-step prediction probability

$$\mathbf{P}\{q_k = v|q_{1:k-1}\} = \frac{\mathbf{P}\{q_{1:k-1}, q_k = v\}}{\mathbf{P}\{q_{1:k-1}\}} = \frac{\mathbb{P}_k(\nu)}{\mathbb{P}_{k-1}}, \tag{9}$$

where $v \in \mathcal{Q}$ is an element of the finite mode set $\mathcal{Q}$ and we have implicitly assumed computation with respect to a particular SCFG model $\mathcal{L}^{\mathrm{SCFG}}$ and excluded it from the expressions for the sake of brevity. The Earley-Stolcke parser [6] provides an efficient algorithm to compute the quantity in (8) and the related one-step prediction probability in (9). A brief algorithmic description is provided in Appendix A.

IV. Bayesian Filtering of Syntactic TBD

In this section, a particle-filter based solution is derived for the multiple model track before detect problem outlined in Section II. Finite-dimensional filters are not known for non-linear and non-Gaussian measurement processes. The switching state-space model also introduces a posterior density with an exponentially increasing number of components. As a result, a particle filtering approach is employed with efficient use of the Earley-Stolcke parser as a proposal density generator. Finally, in Section IV-B, a Rao-Blackwellised scheme is outlined for variance reduction in the estimates.

A. Multiple-Model SCFG Particle Filter

Consider the extended target density $\mathbf{P}\{q_{1:k-1}, e_{1:k-1}, \mathbf{x}_{1:k-1}|\mathbf{z}_{1:k-1}\}$ that we are interested in for the multi-frame TBD problem. We define an extended target state $\mathbf{s}_{k-1} = \{q_{k-1}, e_{k-1}, \mathbf{x}_{k-1}\}$ and approximate the prior extended state density with an empirical random measure such that

$$\mathbf{P}\{q_{1:k-1}, e_{1:k-1}, \mathbf{x}_{1:k-1}|\mathbf{z}_{1:k-1}\} \approx \left\{w_{k-1}^{(i)}, \mathbf{s}_{1:k-1}^{(i)}\right\}_{i=1}^{N_p}, \tag{10}$$

where N_p is the number of particles used. A sequential importance sampling approach is used to obtain the posterior density in the usual manner using a prediction step and an update step. The prediction step can be decomposed as

$$\begin{aligned}
&\mathbf{P}\left\{q_{1:k}^{(i)}, e_k^{(i)}, \mathbf{x}_{1:k}^{(i)}|\mathbf{z}_{1:k-1}\right\} \\
&\quad= \mathbf{P}\left\{\mathbf{s}_{k-1}^{(i)}|\mathbf{z}_{1:k-1}\right\} \\
&\qquad\times \mathbf{P}\left\{\mathbf{x}_k^{(i)}|q_{1:k}^{(i)}, e_{1:k}^{(i)}, \mathbf{x}_{1:k-1}^{(i)}, \mathbf{z}_{1:k-1}\right\} \\
&\qquad\times \mathbf{P}\left\{e_k^{(i)}|q_{1:k}^{(i)}, e_{1:k-1}^{(i)}, \mathbf{x}_{1:k-1}^{(i)}, \mathbf{z}_{1:k-1}\right\} \\
&\qquad\times \mathbf{P}\left\{q_k^{(i)}|q_{1:k-1}^{(i)}, e_{1:k-1}^{(i)}, \mathbf{x}_{1:k-1}^{(i)}, \mathbf{z}_{1:k-1}\right\}.
\end{aligned} \tag{11}$$

In (11), the first term on the right hand side represents the prior density from (10). To generate new samples for the hybrid state $\mathbf{s}_k^{(i)}$, we use the respective transitional densities as proposal functions for the particle filter propagation. The continuous-valued state is propagated by sampling from the state

transition function in (1). The target existence is propagated using the birth-death Markov transition matrix in (6). Finally, the mode is propagated using the one-step prediction probability in (9). Mathematically, we sample N_p particles from the bootstrap proposal such that $\mathbf{s}_k^{(i)} \sim \pi\{\mathbf{s}_k|\mathbf{s}_{1:k-1}^{(i)}\}$, where

$$\pi\left\{\mathbf{s}_k^{(i)}|\mathbf{s}_{1:k-1}^{(i)}\right\} = f_{q_k^{(i)}}\left(\mathbf{x}_{k-1}^{(i)}, e_{k-1:k}^{(i)}, \mathbf{v}_k^{(i)}\right) \times \mathbf{P}\left\{e_k^{(i)} = n|e_{k-1}^{(i)} = m\right\}\mathbf{P}\left\{q_k^{(i)} = v|q_{1:k-1}^{(i)}\right\} \quad (12)$$

If the predicted target existence $e_k^{(i)} = 1$, then the following possibilities can occur:

Target Birth: When $e_{k-1}^{(i)} = 0$ and $e_k^{(i)} = 1$, the target state is drawn as a sample from a birth proposal density $\pi_{\text{birth}}\{\mathbf{x}_k^{(i)}|\mathbf{z}_k\}$ that is obtained in the following manner. The target position components are sampled from a uniform density over positions in the surveillance region where $\mathbf{z}_k > \gamma$, where γ is an intensity threshold. The newborn particles are thus positioned in regions of the surveillance area where the most recent sensor measurement has a large value. The target velocity component is sampled from a uniform proposal density $\sim \mathcal{U}[-\mathbf{v}_{\max}, \mathbf{v}_{\max}]$, where $\mathbf{v}_{\max} \in \mathbb{R}^2$ is the maximum assumed velocity in the x and y directions.

Target Continuance: When $e_{k-1}^{(i)} = 1$ and $e_k^{(i)} = 1$, the particle stays alive and the target state is drawn from the continuance proposal which is taken to be the transitional prior defined in (1).

If the predicted target existence $e_k^n = 0$, the target state is undefined. This is represented by $\mathbf{x}_k = \phi$. A new sampled particle $\mathbf{s}_k^{(i)} = \{\mathbf{x}_k^{(i)}, e_k^{(i)}, q_k^{(i)}\}$ is then appended to the particle representation such that $\mathbf{s}_{1:k}^{(i)} = \{\mathbf{s}_{1:k-1}^{(i)}, \mathbf{s}_k^{(i)}\}$. The measurement update for the particle filter is given by

$$\mathbf{P}\{\mathbf{x}_{1:k}, e_{1:k}, q_{1:k}|\mathbf{z}_{1:k}\} \propto \mathbf{P}\{\mathbf{z}_k|\mathbf{x}_{1:k}, e_{1:k}, q_{1:k}\}\mathbf{P}\{\mathbf{x}_{1:k}, e_{1:k}, q_{1:k}|\mathbf{z}_{1:k-1}\}, \quad (13)$$

where $\mathbf{P}\{\mathbf{z}_k|\mathbf{x}_{1:k}, e_{1:k}, q_{1:k}\} = \mathbf{P}\{\mathbf{z}_k|\mathbf{x}_k, e_k\}$ is the likelihood of the sensor measurement given by (5) and $\mathbf{P}\{\mathbf{x}_{1:k}, e_{1:k}, q_{1:k}|\mathbf{z}_{1:k-1}\}$ is the predicted density given by (12). The measurement likelihood described in (5) is a product of i.i.d random variables at each bin location (m, n). However, it is common in TBD literature [12] to limit the influence of a target in a spatial bin (m, n) to a small neighborhood centered around that bin. We denote this neighborhood by $\mathcal{C}_m(\mathbf{x}_k)$ indicating the affected bins in the X-dimension and $\mathcal{C}_n(\mathbf{x}_k)$ indicating the affected bins in the Y-dimension. The incremental un-normalized importance weights for the bootstrap particle filter are then

$$\tilde{w}_k^{(i)} = \begin{cases} \prod_{m\in\mathcal{C}_m\left(\mathbf{x}_k^{(i)}\right)}\prod_{n\in\mathcal{C}_n\left(\mathbf{x}_k^{(i)}\right)} l\left(z_k^{m,n}|\mathbf{x}_k^{(i)}\right) & \text{if } e_k^{(i)} = 1 \\ 1, & \text{if } e_k^{(i)} = 0. \end{cases} \quad (14)$$

Finally, the weights are normalized and an appropriate resampling step is carried out. At each stage of the particle filtering algorithm, we can compute estimates from the particle representation of the extended state $\{\mathbf{s}_k^{(i)}, w_k^{(i)}\}_{i=1}^{N_p}$ Target existence $\hat{e}_k$ can be computed as

$$\hat{e}_k = \frac{\sum_{i=1}^{N_p} e_k^{(i)}}{N_p}. \quad (15)$$

Target presence is then declared if $\hat{e}_k$ is above a threshold value. This can be used to initiate a track based on the estimated target state given by

$$\hat{\mathbf{x}}_k = \frac{\sum_{i=1}^{N_p} \mathbf{x}_k^{(i)} e_k^{(i)}}{\sum_{i=1}^{N_p} e_k^{(i)}} \quad (16)$$

In a similar manner, we can also compute mode estimates at each time instant k by computing the proportion of particles in each mode and selecting the most likely mode

$$\hat{q}_k = \arg\max_{v\in\mathcal{V}} \frac{\sum_{i=1}^{N_p} \mathbb{I}\left(q_k^{(i)} = v\right) e_k^{(i)}}{\sum_{i=1}^{N_p} e_k^{(i)}}. \quad (17)$$

An algorithmic description of the SCFG multiple model particle filter for syntactic track-before detect is presented in Algorithm 1.

Algorithm 1: SCFG-driven Multiple Model Particle Filter

function SCFG-MMPF $(\{\mathbf{s}_{1:k-1}^{(i)}, w_{k-1}^{(i)}\}, \mathbf{z}_k)$
  Sample $q_k^{(i)} \sim \mathbf{P}\{q_k|q_{1:k-1}^{(i)}; \mathcal{L}^{\text{SCFG}}\}$ using (9)
  Sample $e_k^{(i)} \sim \mathbf{P}\{e_k|e_{k-1}^{(i)}\}$ using (6)
  for $i \leftarrow 1$ to N_p **do**
    if $e_k^{(i)} = 1, e_{k-1}^{(i)} = 0$ (target birth) **then**
      Sample $\mathbf{x}_k^{(i)} \sim \pi_{\text{birth}}\{\mathbf{z}_k\}$
    else if $e_k^{(i)} = 1, e_{k-1}^{(i)} = 1$ (continuation) **then**
      Sample $\mathbf{x}_k^{(i)} \sim F\mathbf{x}_{k-1}^{(i)} + G\mathbf{v}_k(q_k^{(i)})$
    else if $e_k^{(i)} = 0$ (target death) **then**
      $\mathbf{x}_k^{(i)} = \emptyset$
    Evaluate $\tilde{w}_k^{(i)}$ using (14)
  $W_k = \sum_{i=1}^{N_p} \tilde{w}_k^{(i)}$
  Normalize $w_k^{(i)} = \frac{\tilde{w}_k^{(i)}}{W_k}, \forall i = \{1, \ldots, N_p\}$
  if $N_{\text{effective}} < \text{threshold}$ **then**
    RESAMPLE $\{\mathbf{s}_k^{(i)}, w_k^{(i)}\}$
    for $i \leftarrow 1$ to N_p **do**
      $w_k^{(i)} \leftarrow \frac{1}{N_p}$
  return $\{\mathbf{s}_{1:k}^{(i)}, w_k^{(i)}\}$

B. Rao-Blackwellised Multiple-Model SCFG Particle Filter

In this section, analytical sub-structure present in the problem is used to reduce the variance of our estimates by using a Rao-

Algorithm 2: Rao-Blackwellised Particle Filter for Syntactic TBD

function RBPF $(\{\mathbf{s}_{1:k-1}^{(i)}, w_{k-1}^{(i)}\}, \zeta_{k-1}^{(i)}, \mathbf{z}_k)$
  for $i \leftarrow 1$ to N_p **do**
    Predict $\zeta_{k|k-1}^{(i)}(p)$ using (19)
    Sample $\mathbf{s}_k^{(i)} \sim \pi_{\mathrm{RBPF}}\{\mathbf{s}_k^{(i)}|\mathbf{s}_{1:k-1}^{(i)}\}$ using (20)
    Update importance weights $\tilde{w}_k^{(i)}$ using (22)
  $W_k = \sum_{i=1}^{N_p} \tilde{w}_k^{(i)}$
  for $i \leftarrow 1$ to N_p **do**
    Normalize $w_k^{(i)} = \frac{\tilde{w}_k^{(i)}}{W_k}, \forall i = \{1, \ldots, N_p\}$
    Update $\zeta_{k|k}^{(i)}$ using (21)
  if $N_{\mathrm{effective}} < \mathrm{threshold}$ **then**
    RESAMPLE $\{\mathbf{s}_k^{(i)}, w_k^{(i)}\}$
    for $i \leftarrow 1$ to N_p **do**
      $w_k^{(i)} \leftarrow \frac{1}{N_p}$
    return $\{\mathbf{x}_{1:k}^{(i)}, q_{1:k}^{(i)}, w_k^{(i)}\}, \zeta_k^{(i)}$

Blackwellised [29] version of the particle filter in Section IV-A. Consider the filtering density $\mathbf{P}\{e_{k-1}, q_{1:k-1}, \mathbf{x}_{1:k-1}|\mathbf{z}_{1:k-1}\}$ decomposed as

$$\mathbf{P}\{e_{k-1}, q_{1:k-1}, \mathbf{x}_{1:k-1}|\mathbf{z}_{1:k-1}\} = \mathbf{P}\{e_{k-1}|q_{1:k-1}, \mathbf{x}_{1:k-1}, \mathbf{z}_{1:k-1}\}\mathbf{P}\{q_{1:k-1}, \mathbf{x}_{1:k-1}|\mathbf{z}_{1:k-1}\}. \quad (18)$$

It can be observed that conditioned on the state sequence $\mathbf{x}_{1:k-1}$ and the mode sequence $q_{1:k-1}$, the target existence is generated by a Markov chain. Consequently, the HMM filter can be used to exploit analytical sub-structure for the conditional target existence density $\zeta_{k-1}(l) = \mathbf{P}\{e_{k-1} = l|q_{1:k-1}, \mathbf{x}_{1:k-1}, \mathbf{z}_{1:k-1}\}$. The continuous-valued target state $\mathbf{x}_{k-1}$ and the discrete-valued mode q_{k-1} are represented by an extended state $\mathbf{s}_{k-1} = \{q_{k-1}, \mathbf{x}_{k-1}\}$ and their conditional density is approximated by a set of N_p weighted random particles as the empirical random measure $\{\mathbf{s}_{1:k-1}^{(i)}, w_{k-1}^{(i)}\}_{i=1}^{N_p}$.

The prediction of the conditional target existence density is given by

$$\zeta_{k|k-1}^{(i)}(p) = \sum_l \mathbf{P}\{e_k = p|e_{k-1} = l\}\zeta_{k-1}^{(i)}(l), \quad (19)$$

for $l \in \{0, 1\}$ and $i = 1, \ldots, N_p$. The prediction for the random measure $\{\mathbf{s}_{1:k-1}^{(i)}, w_{k-1}^{(i)}\}_{i=1}^{N_p}$ is performed using a suitable proposal density $\pi_{\mathrm{RBPF}}\{\mathbf{s}_k^{(i)}|\mathbf{s}_{1:k-1}^{(i)}, \mathbf{z}_{1:k-1}\}$. We choose to use the popular bootstrap proposal such that prediction of the extended target state is given by

$$\pi_{\mathrm{RBPF}}\left\{\mathbf{s}_k^{(i)}|\mathbf{s}_{1:k-1}^{(i)}\right\} = f_{q_k^{(i)}}\left(\mathbf{x}_{k-1}^{(i)}, \mathbf{v}_k^{(i)}\right)\mathbf{P}\left\{q_k^{(i)} = v|q_{1:k-1}^{(i)}\right\}, \quad (20)$$

where $f_{q_k^{(i)}}(\mathbf{x}_{k-1}^{(i)}, \mathbf{v}_k^{(i)})$ is the state transition function in (1) and $\mathbf{P}\{q_k^{(i)} = v|q_{1:k-1}^{(i)}\}$ is the one-step prediction in (9).

The measurement update for the target existence is given by

$$\zeta_{k|k}^{(i)}(l) = \mathbf{P}\left\{\mathbf{z}_k|\mathbf{x}_k^{(i)}, e_k^{(i)} = l\right\}\mathbf{P}\left\{\mathbf{s}_k^{(i)}|\mathbf{s}_{1:k-1}^{(i)}, \mathbf{z}_{1:k-1}\right\} \quad (21)$$

Finally, the measurement update for the extended state is done using the incremental weight update such that

$$\mathbf{P}\left\{\mathbf{s}_{1:k}^{(i)}|\mathbf{z}_{1:k}\right\} \propto \frac{\mathbf{P}\left\{\mathbf{z}_k|\mathbf{s}_{1:k-1}^{(i)}, \mathbf{z}_{1:k-1}\right\}\mathbf{P}\left\{\mathbf{s}_{1:k}^{(i)}|\mathbf{z}_{1:k-1}\right\}}{\pi_{\mathrm{RBPF}}\left\{\mathbf{s}_k^{(i)}|\mathbf{s}_{1:k-1}^{(i)}, \mathbf{z}_{1:k-1}\right\}} = \sum_l \mathbf{P}\left\{\mathbf{z}_k|\mathbf{x}_k^{(i)}, e_k^{(i)} = l\right\}\zeta_{k|k-1}^{(i)}(l)\tilde{w}_{k-1}^{(i)} \quad (22)$$

In (21) and (22), the term $\mathbf{P}\{\mathbf{z}_k|\mathbf{x}_k^{(i)}, e_k^{(i)}\}$ is evaluated using the measurement likelihood in (14). An algorithmic description of the Rao-Blackwellised particle filter is provided in Algorithm 2.

V. Numerical Examples

In this section, two types of numerical examples are considered. First, a reduced state space "toy" example is detailed to illustrate the concepts in the paper in a tutorial fashion. Then, we perform simulations on a realistic real-world scenario. The detection and tracking performance of syntactic track-before-detect is evaluated at various SNR levels in a Monte-Carlo fashion. There are two competing architectures used in all simulations. The first type of architecture called "Markov-MMPF" (Markov-switching multiple model particle filter) considers multiple models switching according to a Markov chain model as described by (7). The second architecture is called "SCFG-MMPF" (SCFG-switching multiple model particle filter) which considers mode switching behavior driven by an SCFG.

A. Reduced State-Space "Toy" Example

Consider the one-dimensional non-linear switching stochastic volatility model [30] commonly used in quantitative finance. The mode (called the drift parameter) represents a "volatility" state which is assumed to be either a low-volatility state $q_k = a$ or a high-volatility state $q_k = b$. The modes cause the log-volatility to switch between two states in a linear auto-regressive process

$$\mathbf{x}_k = \alpha(q_k) + \phi\mathbf{x}_{k-1} + \sigma\mathbf{v}_k,\ \mathbf{v}_k \sim \mathcal{N}(0, 1)$$

where $\alpha(q_k = 0) = -5.0$ and $\alpha(q_k = 1) = -2.0$, $\phi = 0.5$ and $\sigma^2 = 0.1$. The binary-valued mode q_k is assumed to arise from an SCFG with form $a^n b^+ a^n$ having grammatical rules $S \rightarrow aSa$ [0.6] | X [0.4] and $X \rightarrow bX$ [0.5] | b [0.5]. Such a model captures scenarios in which the volatility of a certain asset follows a pattern of being in the low volatility state for equal time periods with a high volatility period of arbitrary length in between. We are interested in tracking the log-volatility under such a scenario. The observations are conditionally independent given the latent state $\mathbf{x}_k$ such that

$$\mathbf{z}_k = \exp\left(\frac{\mathbf{x}_k}{2}\right)\mathbf{w}_k,\ \mathbf{w}_k \sim \mathcal{N}(0, 1).$$

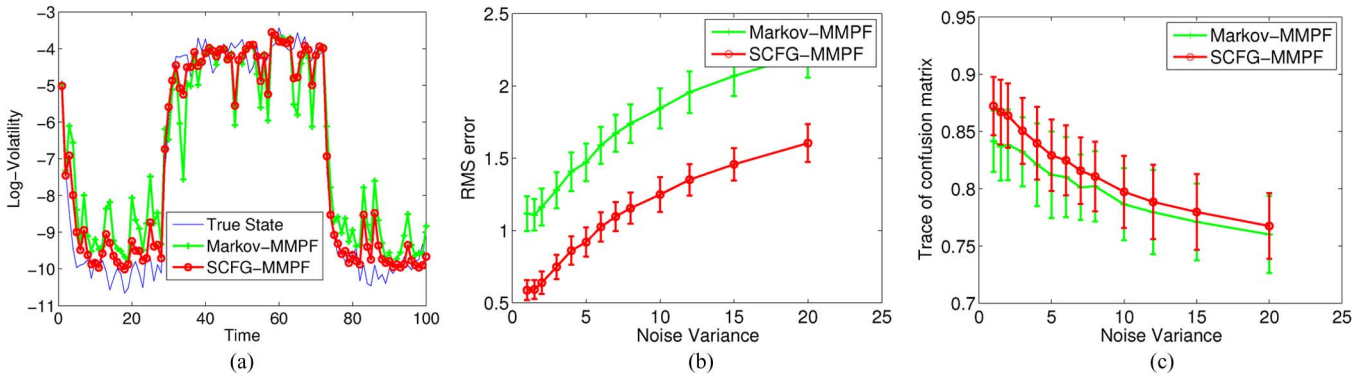

Fig. 7. (a) shows the simulated and estimated log-volatility at a noise variance of 2.0 for one of the 1000 Monte Carlo runs. It can be visually observed that the SCFG-MMPF performs better than the Markov-MMPF. (b) shows the RMS error between the true target state and the SCFG and Markov chain versions. Similarly, (c) shows the mode estimation rate. In either case, the SCFG version performs better than the Markov chain version.

The conditional probability distributions for the state variables $\mathbf{x}_k$ and the observations $\mathbf{z}_k$ are given by

$$\begin{aligned}\mathbf{P}\{q_1\} &\sim \mathbb{P}_1(v) \text{ using (8)},\\ \mathbf{P}\{\mathbf{x}_1|q_1\} &\sim \mathcal{N}\left(\mathbf{x}_1; 0, \frac{\sigma^2}{1-[\alpha(q_1)+\phi]^2}\right),\\ \mathbf{P}\{\mathbf{x}_k|q_k,\mathbf{x}_{k-1}\} &\sim \mathcal{N}\left(\mathbf{x}_k; \alpha(q_k)+\phi\mathbf{x}_{k-1}, \sigma^2\right),\\ \mathbf{P}\{\mathbf{z}_k|\mathbf{x}_k\} &\sim \mathcal{N}(\mathbf{z}_k; 0, \exp \mathbf{x}_k)\end{aligned}$$

Using the model specified above, 1000 Monte-Carlo runs were simulated by changing the measurement noise variance from 1.0 to 20.0 and keeping the state noise variance fixed at 1.0. We show the results of the Monte-Carlo experiment in Fig. 7. The continuous-valued state $\hat{\mathbf{x}}_k$ estimates are evaluated by computing the root mean square (RMS) error for each Monte-Carlo run as

$$\text{RMSE} = \sqrt{\frac{1}{K}\sum_k \|\hat{\mathbf{x}}_k - \mathbf{x}_k\|_2^2} \tag{23}$$

The discrete-valued mode is compared using the notion of a confusion matrix. If the discrete mode q takes $|\mathcal{Q}|$ values, then a $|\mathcal{Q}| \times |\mathcal{Q}|$ confusion matrix $\mathcal{M}_q(i,j)$ enumerates the number of times the true mode i is estimated as mode j. The mode detection rate is then taken as the trace of the normalized confusion matrix. The normalization in each row i is with respect to the total number of times mode $q = i$ appears in the simulated sequence. A perfect mode estimation would result in $trace(\mathcal{M}_q^{\text{normalized}}(i,j)) = 1.0$.

The SCFG-MMPF of Algorithm 1 (without the existence variable computations) is able to track the log-volatility with a lower root mean square error (RMSE) and also detects the modes with a higher accuracy. The Markov-MMPF uses the same algorithm but the proposal density for the mode comes from the corresponding Markov transition matrix representing Markovian switching of the volatility modes. We also observe that the variance of the RMSE over all Monte-Carlo runs at a particular noise variance is smaller for the SCFG-MMPF.

B. Syntactic TBD Examples

We show, through numerical simulations, that constraining a particle filter TBD solution along trajectory models results in a reduced error in mode q_k, existence e_k and state $\mathbf{x}_k$ estimates compared to competing architectures. The detection performance is analyzed by calculating the probability of detection $\mathbf{P}_d$ over 1000 Monte Carlo runs of the corresponding TBD filter. If H_0 represents the hypothesis that a target is absent $(e_k = 0)$ and H_1 represents the hypothesis that a target is present $(e_k = 1)$, then the detection probability $\mathbf{P}_d$ is given by the number of times that H_1 is true and the syntactic TBD filter chooses H_1 for each of the K frames of available sensor measurements. The tracking performance is measured using the root mean square error (RMSE) of the target position and velocity.

In all simulations, the time step $\delta T = 0.1$ seconds. The process noise co-variance is constructed using $\sigma_a = 1.0$ and $\sigma_o = 0.01$. A sequence of 100 frames is generated in each experiment with $M = N = 40$, Δ is dependent on the extents of the trajectory for visualization purposes, the target intensity is assumed constant at $I = 20$ and the sensor spread parameter $\sigma_h = 0.7$. The Rayleigh clutter power $P = 3$. The sensor spread function is also truncated to affect only the closest 8 neighbors of the center pixel (i,j).

An example of a raw sensor measurement is shown in Fig. 9. Each frame depicts the returned intensities at a single time instant. A white pixel indicates a high returned intensity while a black pixel indicates a small returned intensity. An operator looking at thresholded sensor measurements would find it impossible to extract target tracks from visual inspection alone. The syntactic TBD particle filter uses the following parameters. The target existence is modeled by a birth probability $P_{\text{birth}} = 0.1$ and a death probability $P_{\text{death}} = 0.1$. The intensity threshold $\gamma = 0.2\max(\mathbf{z}_k)$ and the maximum velocity is $\mathbf{v}_{\max} = [1,1]^T$ units/sec. The number of particles used in the filter $N_p = 1000$. Finally, a track is initiated and a target is declared present if the estimated target existence $\hat{e}_k > 0.7$. A simulated rectangle trajectory and the TBD track output is shown in Fig. 8(a). The input to the TBD tracker is the sequence of sensor measurements shown in Fig. 9. This particular configuration amounts to a 2 dB signal-to-noise ratio. We observe that the SCFG derived mode sequence is able to track the target better than the case of a Markov chain based mode sequence.

In Fig. 8(c), the results of running 1000 Monte Carlo runs of a multiple model particle filter are shown as the SNR is changed.

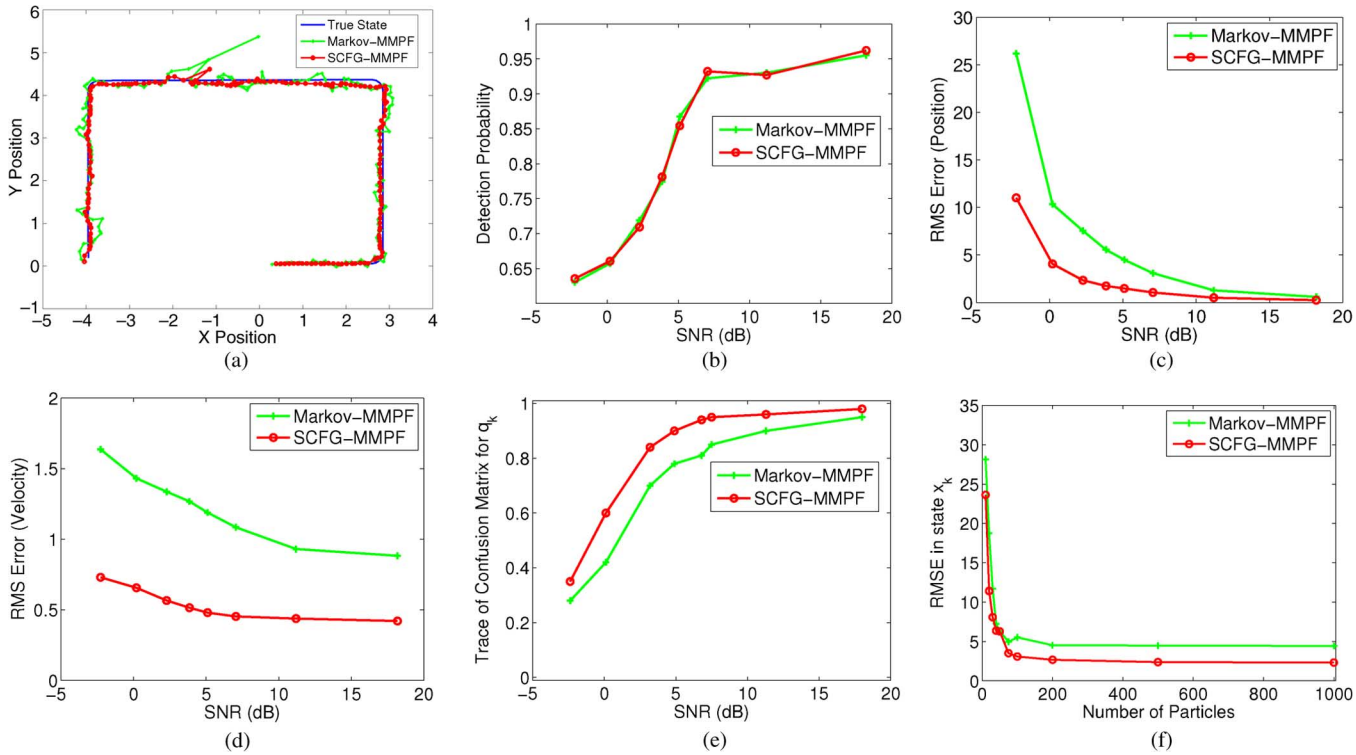

Fig. 8. (a) shows the simulated and estimated track at an SNR of 2 dB of one of the 1000 Monte Carlo runs. (b) shows the change in the detection probability with SNR. (c) shows the RMS error in position (X and Y combined) of the SCFG and Markov chain versions. Similarly, (d) shows the RMS error in velocity (X and Y combined) of the competing models. In either case, the SCFG trajectory models perform better than the Markov chain version. (e) shows the error in the mode q_k in terms of the trace of the confusion matrix as explained in Section V-A. (f) shows the effect of the number of particles on the performance of the proposed filters. The performance (in terms of RMSE of the target state $\mathbf{x}_k$ saturates at approximately 200 particles.

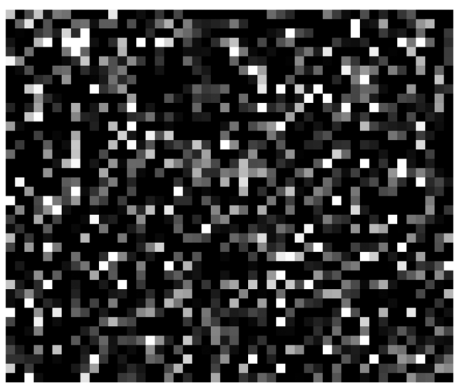

Fig. 9. Each sensor measurement is an image of returned intensity values. The dark bins corresponds to weak radar returns while the lighter bins correspond to strong radar returns.

A range of SNR values is swept by changing the target intensity and the noise variance. The plot of $\mathbf{P}_d$ versus SNR is shown in Fig. 8(b) which does not show a significant difference between the competing models. The detection probability reflects on the performance of the filters in estimating the existence variable e_k. The existence variable behaves like a switching measurement which is treated in the same manner (through Markovian switching parameterized by P_{birth} and P_{death}) in both Markov and SCFG-MMPF. As a result, there is little difference in the estimation error of e_k. However, the SCFG trajectory models perform much better in terms of estimation error in the state variable $\mathbf{x}_k^{(i)}$. The decrease in the root mean squared error (RMSE) of the X and Y position is shown in Fig. 8(c). A similar decrease in the RMSE of the X and Y velocity estimates is obtained for the SCFG trajectory models over the Markov chain based model as depicted in Fig. 8(d). In addition, the SCFG-MMPF also estimates the mode q_k more accurately than the Markov-MMPF as seen in Fig. 8(e).

The SCFG-MMPF uses the expensive $\mathcal{O}(N^3)$ Earley-Stolcke parser as a bootstrap proposal density generator as opposed to the constant-time $\mathcal{O}(1)$ lookup afforded by the Markov chain transitions. However, as seen in Fig. 8(f), the performance of both filters saturates after 200 particles. Consequently, the Markov-MMPF cannot achieve better performance by naively increasing the number of particles.

VI. Conclusion

This paper has addressed the question: suppose one is interested in estimating the state of a dimly lit target that is moving according to a specified class of trajectories (intent). How can a TBD algorithm be derived to exploit this information? We presented stochastic-context free grammar models arising in natural language processing to model complex spatial trajectory patterns. The syntactic modeling framework facilitates incorporating meta-level information from human operators regarding suspicious trajectory patterns into the TBD problem. A novel particle filtering algorithm coupled with the Earley-Stolcke parser was derived to estimate the target state and existence. Such a modeling approach can be viewed as a generalization of jump Markov state-space models to jump SCFG state-space models. We also derive a Rao-Blackwellised version of our

proposed particle filter to reduce the variance in the estimates. Finally, the numerical simulations demonstrate a marked increase in tracking performance (RMSE) over competing Markov chain based trajectory models.

In its present form, the proposed particle filter requires running N_p Earley-Stolcke parsers in parallel for each particle. We attempted to exploit analytical sub-structure in the SCFG mode sequence by deriving a Rao-Blackwellised version using a decision-directed scheme. In such a scheme, previous estimates of the mode $\hat{q}_{1:k}$ drive a single Earley-Stolcke parser rather than maintaining separate Earley-Stolcke parsers for each particle. However, we found that errors in the past estimates of the mode cause the filter to diverge. In future work, we seek to explore approximations enabling more efficient conditioning on the mode.

APPENDIX A
THE EARLEY-STOLCKE PARSER

In this section, for completeness, we provide a brief description on the operation of the Earley-Stolcke [6] parser and its use in computing the prefix probability $\mathbb{P}_k$. The Earley-Stolcke parser scans an input terminal string $q_1, \ldots, q_k, \ldots, q_K$ from left to right and is able to compute the probability of the string $\mathbf{P}\{q_{1:K}|\mathcal{L}^{\mathrm{SCFG}}\}$ given the parameters of the SCFG. As each symbol q_k is scanned, a set of states $u_k = \{u_k^n, n = 1, \ldots, |u_k|\}$ are created which represent all the different derivations that can explain the observations until instant k.

Each state $u_k^n \in u_k$ represents (1) a re-writing rule $r_m \in \mathcal{R}$ such that the portion of the input string being currently scanned is derived from its right hand side, (2) a dot (marker) demarcating a position in the right hand side of that production rule. The position of the dot represents the portion of the right hand side that has already been recognized and (3) a pointer back to the position in the input string at which we began to look for an instance of the application of that production rule. Each state is an incomplete portion of a sequence of rule choices which could have generated the input string. These states are the control structure used by the Earley-Stolcke parser to store the incomplete derivation trees. They are denoted by the notation ${}_i^j X \rightarrow \lambda \cdot Y\mu[\alpha, \gamma]$. 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. Each state is also associated with a forward probability α and an inner probability γ which are explained in more detail in the sequel. For the purposes of dealing with the start symbol, the Earley-Stolcke uses an initial dummy state ${}_0^0 S' \rightarrow \cdot S[\alpha = 1, \gamma = 1]$.

Earley-Stolcke Operations

The states in an Earley set u_k are processed in order, by performing one of three operations on each state. These operations may add more states to u_k and may also put states in a new state set u_{k+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. The *prediction* operation is applied to states when there is a non-terminal to the right of the dot. It causes the addition of one new state to u_k 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 k, since the state was created in u_k. Thus the predictor adds to u_k all productions which might generate sub-strings beginning with q_{k+1}. More formally, for a state ${}_i^k X \rightarrow \lambda \cdot Y\mu$ in the state set u_k, the predictor adds a new state ${}_k^k Y \rightarrow \cdot\nu$ for each of the alternative production rules $(Y \rightarrow \nu) \in \mathcal{A}$. A link is thus created between these states. The state ${}_k^k Y \rightarrow \cdot\nu$ is called a *predicted* state.

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

The third operation, the *completion* operation, is applicable to a state if its dot is at the end (${}_i^k X \rightarrow \lambda Y \mu.$) of its production. Such a state is called a "complete" state. For every complete state, the parser back-tracks to the state set i indicated by the pointer in the complete state, and add all states from u_i to u_k 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_k 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_i which caused us to look for a X, and we move

Algorithm 3: Earley-Stolcke Parser

function EARLEY-STOLCKE PARSER $(u_{0:k-1}, q_k)$
Scanning
  for ${}_j^{k-1} X \rightarrow \lambda \cdot a\mu[\alpha, \gamma] \in u_{k-1}$ **do**
    Add ${}_j^k X \rightarrow \lambda a \cdot \mu[\alpha, \gamma]$ if $\mathbf{P}\{q_k|a\} > 0$.
    $\alpha' = \alpha \mathbf{P}\{q_k|a\}$
    $\gamma' = \gamma \mathbf{P}\{q_k|a\}$
  $\mathbb{P}_k = \sum_{n \in u_k} \alpha({}_j^k X \rightarrow \lambda a \cdot \mu)$
Completion
  for ${}_j^k Y \rightarrow \nu \cdot [\alpha'', \gamma''] \in u_k$ **do**
    for ${}_i^j X \rightarrow \lambda \cdot Z\mu[\alpha, \gamma] \in u_j$ **do**
      if $R_U(Z, Y) \neq 0$ **then**
        Add ${}_i^k X \rightarrow \lambda Z \cdot \mu[\alpha', \gamma']$
        $\alpha' += \alpha\gamma'' R_U(Z, Y)$
        $\gamma' += \gamma\gamma'' R_U(Z, Y)$
Prediction
  for ${}_j^i X \rightarrow \lambda \cdot Z\mu[\alpha, \gamma] \in u_i$ **do**
    Add ${}_k^k Y \rightarrow \cdot\nu[\alpha', \gamma']$ if $R_L(Z, Y) \neq 0$,
    $\alpha' += \alpha R_L(Z, Y) \mathbf{P}\{Y \rightarrow \nu\}$
    $\gamma' = \mathbf{P}\{Y \rightarrow \nu\}$
  return $u_{k+1}, \mathbb{P}_k$

the dot over the X in these states to show that it has been successfully scanned. This process implies that the application of a production rule in the past has been validated. A completion operation adds a new state ${}_i^k X \rightarrow \lambda Y \cdot \mu$ (called a completed state) using ${}_i^j X \rightarrow \lambda \cdot Y\mu$ and ${}_j^k Y \rightarrow \nu\cdot$. A link pointing from ${}_j^k Y \rightarrow \nu\cdot$ to ${}_i^k X \rightarrow \lambda Y \cdot \mu$ is also created. In such a manner, the Earley-Stolcke parser continues until all the observation symbols have been scanned. If the final state set u_K contains the state ${}_0^0 S' \rightarrow S\cdot$, then the algorithm terminates successfully. It represents a successful parse of the sentence $q_1, \ldots, q_k, \ldots, q_K$.

Earley-Stolcke Probabilities

We mentioned earlier that each state is associated with a forward probability $\alpha({}_i^j X \rightarrow \lambda \cdot \mu)$ which is the sum of the probabilities of all paths of length i which end in the state ${}_i^j X \rightarrow \lambda \cdot \mu$ and generate observations $q_1, \ldots, q_i$. The inner probability $\gamma({}_i^k X \rightarrow \lambda \cdot \mu)$ of a state is defined as the sum of the probability of all paths of length $k - i$ which start in ${}_i^k X \rightarrow \cdot\lambda\mu$ and end in state ${}_i^k X \rightarrow \lambda \cdot \mu$ and hence derive the observations $q_i, \ldots, q_{k-1}$.

The recursive updates of the forward probability α and inner probability γ for each of the state operations is summarized in Algorithm 3. $R_L(Z, Y)$ and $R_U(Z, Y)$ are pre-computed $|\mathcal{N}| \times |\mathcal{N}|$ matrices representing factors to account for looping production rules [6]. The prefix probability $\mathbb{P}_k$ referred to in Section III-D can be computed as the sum over the forward probabilities of all scanned states in u_k as derived in [6].

REFERENCES

[1] Y. Bar-Shalom, T. Kirubarajan, and X. Li, *Estimation With Applications to Tracking and Navigation*. New York, NY, USA: Wiley , 2002.

[2] B. Ristič, S. Arulampalam, and N. Gordon, *Beyond the Kalman Filter: Particle Filters for Tracking Applications*. Norwood, MA, USA: Artech House, 2004.

[3] U. Grenander, *Elements of Pattern Theory*. Baltimore, MD, USA: Johns Hopkins Univ. Press, 1996.

[4] A. V. Aho and J. D. Ullman, *The Theory of Parsing, Translation, Compiling*. Englewood Cliffs, NJ, USA: Prentice-Hall, 1972.

[5] 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.

[6] A. Stolcke, "An efficient probabilistic context-free parsing algorithm that computes prefix probabilities," *Computat. Ling.*, vol. 21, no. 2, pp. 165–201, 1995.

[7] S.-C. Zhu and D. Mumford, "A stochastic grammar of images," *Found. Trends Comput. Graph. Visual.*, vol. 2, no. 4, pp. 259–362, Jan. 2006.

[8] M. Skolnik, *Introduction to Radar Systems*. New York, NY, USA: McGraw-Hill, 1962.

[9] P. Maybeck and D. Mercier, "A target tracker using spatially distributed infrared measurements," *IEEE Trans. Autom. Control*, vol. 25, no. 2, pp. 222–225, 1980.

[10] Y. Barniv, "Dynamic programming solution for detecting dim moving targets," *IEEE Trans. Aerosp. Electron. Syst.*, vol. AES-21, no. 1, pp. 144–156, Jan. 1985.

[11] M. Bruno, "Bayesian methods for multi-aspect target tracking in image sequences," *IEEE Trans. Signal Process.*, vol. 52, no. 7, pp. 1848–1861, 2004.

[12] D. J. Salmond and H. Birch, "A particle filter for track-before-detect," in *Proc. Amer. Control Conf.*, Arlington, VA, USA, Jun. 2001, pp. 3755–3760.

[13] S. Davey, M. Rutten, and B. Cheung, "A comparison of detection performance for several track-before-detect algorithms," in *Proc. 11th Int. Conf. Inf. Fusion*, 2008, pp. 493–500.

[14] M. Walsh, M. Graham, R. Streit, T. Luginbuhl, and L. Mathews, "Tracking on intensity-modulated sensor data streams," in *Proc. IEEE Aerosp. Conf.*, 2001, vol. 4, pp. 4/1901–4/1909.

[15] B.-N. Vo, B.-T. Vo, N.-T. Pham, and D. Suter, "Joint detection and estimation of multiple objects from image observations," *IEEE Trans. Signal Process.*, vol. 58, no. 10, pp. 5129–5141, 2010.

[16] S. Davey, M. Rutten, and N. Gordon, "Track-before-detect techniques," in *Integrated Tracking, Classification, Sensor Management*, M. Mallick, V. Krishnamurthy, and B.-N. Vo, Eds. New York, NY, USA: Wiley, 2013, pp. 43–74.

[17] E. Grossi, M. Lops, and L. Venturino, "A novel dynamic programming algorithm for track-before-detect in radar systems," *IEEE Trans. Signal Process.*, vol. 61, no. 10, pp. 2608–2619, 2013.

[18] Y. Cheng and T. Singh, "Efficient particle filtering for road-constrained target tracking," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 43, no. 4, pp. 1454–1469, Oct. 2007.

[19] P. Skoglar, U. Orguner, D. Tornqvist, and F. Gustafsson, "Road target tracking with an approximative Rao-Blackwellized particle filter," in *Proc. 12th Int. Conf. Inf. Fusion*, Jul. 2009, pp. 17–24.

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

[21] J. Yin, X. Chai, and Q. Yang, "High-level goal recognition in a wireless LAN," in *Proc. 19th Nat. Conf. Artif. Intell.*, 2004, pp. 578–583, AAAI Press.

[22] E. Marhasev, M. Hadad, G. A. Kaminka, and U. Feintuch, "The use of hidden semi-Markov models in clinical diagnosis maze tasks," *Intell. Data Anal.*, vol. 13, no. 6, pp. 943–967, 2009.

[23] S.-W. Joo and R. Chellappa, "Recognition of multi-object events using attribute grammars," in *Proc. IEEE Int. Conf. Image Process.*, 2006, pp. 2897–2900.

[24] 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, 2011.

[25] M. Fanaswala and V. Krishnamurthy, "Detection of anomalous trajectory patterns in target tracking via stochastic context-free grammars and reciprocal process models," *IEEE J. Sel. Topics Signal Process.*, vol. 7, no. 1, pp. 76–90, 2013.

[26] S. Maskell, "Sequentially structured Bayesian solutions," Ph.D. dissertation, Cambridge Univ., Cambridge, U.K., Feb. 2004.

[27] C. S. Wetherell, "Probabilistic languages: A review and some open questions," *ACM Comput. Surv.*, vol. 12, pp. 361–379, Dec. 1980.

[28] L. E. Baum, T. Petrie, G. Soules, and N. Weiss, "A maximization technique occurring in the statistical analysis of probabilistic functions of Markov chains," *Ann. Math. Statist.*, vol. 41, no. 1, pp. 164–171, 02 1970.

[29] A. Doucet, S. Godsill, and C. Andrieu, "On sequential Monte Carlo sampling methods for Bayesian filtering," *Statist. Comput.*, vol. 10, no. 3, pp. 197–208, 2000.

[30] Y. Bao, C. Chiarella, and B. Kang, "Particle filters for Markov switching Stochastic volatility models," "Quantitative Finance Research Centre, Univ. Technol., Sydney, Australia, Tech. Rep., Jan. 2012.

Mustafa Fanaswala (S'09) received the bachelor's degree in electrical and electronics engineering from the American University of Sharjah, UAE, in 2007, and the M.A.Sc. degree in electrical engineering from Carleton University, Ottawa, in 2009.

He is currently pursuing the Ph.D. degree at the Department of Electrical Engineering, University of British Columbia, Vancouver, Canada. His research is focused on modeling long-range interactions in time-series with applications in target tracking and intent inference.

Vikram Krishnamurthy (F'04) is a professor and Canada Research Chair at the Department of Electrical Engineering, University of British Columbia, Vancouver, Canada. His current research interests include statistical signal processing and stochastic control with applications in social networks and dynamics of protein macromolecules.

Prof. Krishnamurthy served as distinguished lecturer for the IEEE Signal Processing Society and Editor-in-Chief of the IEEE JOURNAL SELECTED TOPICS IN SIGNAL PROCESSING. He received an honorary doctorate from KTH (Royal Institute of Technology), Sweden in 2013.