Intent Inference: Detection of anomalous trajectory patterns in target tracking

Mustafa Fanaswala, Vikram Krishnamurthy

Abstract

This paper develops novel stochastic models and associated signal processing algorithms for detecting intent in terms of anomalous trajectory patterns in target tracking. Stochastic context-free grammar models are presented for anomalous trajectories using the concept of tracklets. Bayesian signal processing algorithms for each model are also described to solve the related classification and prediction problems with polynomial complexity.

I. INTRODUCTION

To motivate the paper, consider the following forensic surveillance application called pattern of life analysis. A pattern of life analysis involves identification of a target’s daily interaction with its environment. Such a pattern of life analysis can be used to predict a target’s behavior based on habit or schedule. Moreover, abnormal behavior that deviates from routine habit usually indicates an event of interest. Consider a street like that shown in Fig. 1a. The normal behavior of the local traffic between the points \((x_1, y_1)\) and \((x_T, y_T)\) is observed to flow in a straight road between the two points. However, if the local population has insider information about terrorist activity (such as installing an improvised explosive device) near the marked embassy, then the local pattern of traffic flow changes to avoid the marked embassy as shown in Fig. 1b. The ability to detect such changes in either single or aggregate target behavior requires a parsimonious representation of target trajectories. Such a representation is provided by stochastic context free grammar (SCFG) models which are also scalable towards dealing with trajectories of different shapes and sizes. Developing SCFG models and associated signal processing and classification algorithms is the main aim of this paper.

Why Use Stochastic Context Free Grammars (SCFGs)?

In signal processing literature, a common assumption made is that the trajectory of the target is highly indicative of the target’s intent. In this paper, the intent of the target is put into one-to-one correspondence with its trajectory. Consequently, when we refer to the trajectory of a target, it has an associated intent. A complete target trajectory is composed of a premeditated set of sub-trajectories to achieve a certain goal. However, premeditation implies a

M. Fanaswala and V. Krishnamurthy are with the Department of Electrical and Computer Engineering, University of British Columbia, Vancouver, BC, V5W 2R1.
E-mail: mustafaf@ece.ubc.ca
long-range time correlation between parts of the trajectory. Existing models to capture inter-dependencies between parts of a trajectory are derived from the common assumption of Markovian behavior. The evolution of the target trajectory is modeled using Gauss-Markov state-space models which assume that the current part of the trajectory is only dependent on the immediate past. While this is usually true for the underlying motion of the target, such an assumption is not entirely valid when modeling entire trajectories. SCFGs provide a more powerful framework to deal with long-range correlation between points in the target trajectory. A trajectory is considered to be made up of a sequence of tracklets. The precise pattern in which tracklets appear together determine the shape, size and final destination of a trajectory. The main goal of this paper is to use SCFGs to model the underlying process which governs the pattern in which the tracklets are combined to form a trajectory.

Modeling anomalous trajectories with SCFGs has several potential advantages: The recursive embedding structure of the possible anomalous trajectory patterns is more naturally modeled in SCFG. As shown in this paper, SCFGs can model arc, rectangles, closed trajectories and other anomalous trajectories like move-stop-move. A move-stop-move trajectory is a tactic used by targets to evade radar detection. SCFGs are more efficient in modeling hidden branching processes when compared to stochastic regular grammars or hidden Markov models with the same number of parameters. The predictive power of a SCFG measured in terms of entropy is greater than that of the stochastic regular grammar [1]. An SCFG is equivalent to a multi-type Galton-Watson branching process with finite number of rewrite rules, and its entropy calculation is discussed in [2].

Trajectory modeling for intent inference is mainly approached in two ways: a) anomaly detection [3]–[5] or b) model-based inference [6]–[8]. In the former, a specific trajectory is not identified. Rather, all possible trajectories of a target are categorized as either normal or anomalous. For example, in [4], a support-vector machine approach is taken to classify aberrant trajectories from normal trajectories. The latter approach of model-based intent inference is taken in this paper. It involves specifically identifying anomalous trajectories of interest.

A key feature of model-based inference is to obtain a semantic interpretation of a complex pattern through the use of simpler sub-patterns. For example, in [7], a dynamic Bayesian network is used to identify scenarios where
a shopper either enters a retail store, leaves a store or passes by the store. In this paper, we consider tracklets as the simpler sub-patterns comprising a trajectory (which is semantically equivalent to intent). The tracklets are explained in detail in Sec. II-B. The work presented in this paper builds upon the work in [9] on target tracking using stochastic context-free grammars in radar tracking applications.

The use of SCFGs as a modeling tool for detecting anomalous trajectories requires the ability to compute model likelihoods. The inside-outside (IO) algorithm [1] is the conventional method used to compute the probability of an observed trajectory belonging to a given grammar model. However, the IO algorithm requires the stochastic grammar to be in a restrictive form. The Earley-Stolcke parser [10] on the other hand is able to deal with arbitrarily structured grammars and is the algorithm used in this paper for the Bayesian estimation of the model probabilities.

II. ANOMALOUS TRAJECTORY CLASSIFICATION FRAMEWORK

In this section, a system-level description of the anomalous trajectory classification problem is presented. We first describe the radar tracking framework and provide a mathematical description of trajectory classification in Sec. II-A In Sec. II-B, complete details on a typical radar tracking system is presented. We also describe how to compute the position and velocity tracklets.
A. Trajectory Classification in Radar Tracking

In this section, an overview of the trajectory classification problem as it pertains to a radar tracking scenario is described. A representation of the proposed system is shown in Fig. 3. A radar is assumed to make measurements $y_t$ related to the position and velocity of targets in a particular region of interest (ROI). These measurements $y_t$ are utilized by a base-level tracker $T$ to estimate the actual position and velocity of the targets. Such a set-up is the conventional “tracker” module used in many tracking applications. We introduce an additional module called the tracklet estimator $H$ which produces quantized position estimates and velocity directions $z_t$ using the output $\Pi_t$ of a base-level tracker. These, in turn, are used by a meta-level inference engine to determine target intent.

The base-level tracker is a nonlinear Bayesian filter (such as a particle filter) which can be represented as an operator $T$ which uses radar measurements $y_t$ to update a posterior distribution $\Pi_t$ over the position and velocity of the target by

$$\Pi_t = T(\Pi_{t-1}, y_t).$$

(1)

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

$$z_t = H_i(\Pi_t),$$

(2)

where $i \in \{\text{position, velocity}\}$. We consider two different tracklets viz., the position tracklets which are quantized by the operator $H_{\text{position}}$ and velocity tracklets which are quantized through $H_{\text{velocity}}$. 
The aim of this paper is to provide models for the process $z_t$ that can be used to classify anomalous trajectories. The target trajectory is associated with an intent depending on the context of the environment. For example, a circling behavior might be indicative of a reconnaissance operation in the vicinity of a sensitive asset like a check-post. A boat that is loitering near the shore-line could also be indicative of a smuggling operation. The trajectory of a target is defined as the time evolution of its path as measured by the state variables of position, velocity and acceleration. A geometric shape like an arc can be used to model a U-turn (or doubling back behavior). Furthermore, rectangles can be used to model circling behavior around regular man-made structures. In addition, the intended destination or even the starting point of a target can be used to infer intent if the target is known to be a friend or a foe.

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

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

B. Radar Tracking System Overview

In this section, a complete radar tracking system is described to complement the arguments and contributions of the paper. As shown in Fig. 3, a particular instance of a target trajectory is considered which goes from its starting point $(x_1, y_1)$ to a destination $(x_T, y_T)$. There may be other intermediate way-points identified as well. Such a trajectory is destination-constrained because the starting, intermediate and ending destinations are known. Moreover, the target trajectory is of a rectangular shape although it is not a closed rectangle. This imposes a shape-constraint on the trajectory (assumed due to existing road constraints).

A typical radar tracking application involves certain assumptions on the dynamics of the target. The target dynamics are summarized using its kinematic state in the vector $s_t$ and a description of how $s_t$ evolves. The general form of the target dynamics is

$$s_{t+1} = f_t(s_t, w_t), \quad (4)$$

where $s_t = [x_t, y_t, \dot{x}_t, \dot{y}_t]^T$ is the state vector, $f_t(\cdot)$ is a possibly time-varying function and $w_t$ is a noise process with a known distribution $P\{w_t\}$. The state variables $x_t, y_t$ refer to the position of the target while $\dot{x}_t, \dot{y}_t$ refer to the velocity of the target in cartesian co-ordinates.

A radar cannot measure the kinematic state of a target directly. It can however make measurements $y_t$ related to the state $s_t$ which are typically of the form

$$y_t = h(s_t) + v_t, \quad (5)$$
where \( \mathbf{y}_t = [r_t, \dot{r}_t, \theta_t] \) is the measurement vector, \( h(\cdot) \) is a non-linear measurement function and \( \mathbf{v}_t \) is a noise process with a known distribution \( \mathbb{P}\{\mathbf{v}_t\} \). The measurement variable \( r \) measures the radial distance of the target to the radar (called range), \( \dot{r} \) is the rate of change of range and \( \theta \) is the azimuth angle (angle between the position vector of the target and the x-axis).

A base-level tracker (like an extended Kalman filter) is then used to track the state of the target \( s_t \) using the radar measurements \( \mathbf{y}_t \). The output of such a tracker consists of a posterior probability distribution \( \Pi_t = \mathbb{P}\{s_t|\mathbf{y}_t\} \). The state estimate can then be evaluated from the posterior distribution using a conditional expectation. The state estimates \( \mathbb{E}\{s_t|\mathbf{y}_t\} \) form the input to the tracklet estimator.

The tracklet estimator \( \mathcal{H} \) is a quantization module which outputs either position or velocity tracklets. Tracklets are used as sub-units which comprise target trajectories. The SCFG models utilize velocity tracklets as sub-units of the trajectory shape while RP models utilize position tracklets as sub-units of a goal-directed trajectory (with a known destination). Each type of tracklet is described below.

**Position Tracklet Estimator:** Consider a region of interest (ROI) extending from the origin \((0, 0)\) to \((x_{\text{max}}, y_{\text{max}})\) in the Cartesian plane as shown in Fig. 4a. The interval \((0, x_{\text{max}})\) on the x-axis is divided into \(N_x\) partitions and the interval \((0, y_{\text{max}})\) on the y-axis is divided into \(N_y\) partitions. As a result, the entire ROI is partitioned into regions indexed by \( i \in \mathcal{I} = \{1, \ldots, N_x \times N_y\} \). Each region \( i \) has a center \((C_{ix}, C_{iy})\). For every estimate of the output distribution \( \Pi_t \) output by the base-level tracker, a conditional mean of the target position is computed. The estimated target position \( x, y \) is compared to the center of each region \( i \). The position tracklet estimator then outputs the index of the region \( i \) which is closest to the estimated position of the target,

\[
z_t = \arg\min_i |x_t - C_{ix}| + |y_t - C_{iy}|,
\]

where \( z_t \) are the “true” quantized grid positions of the target. RP models then use the position tracklets \( z_t \) as sub-units of a target trajectory. However, the position tracklets can only be observed as noisy estimates due to quantization effects and inherent tracking errors.

**Velocity Tracklet Estimator:** The velocity tracklet estimator also uses the posterior distribution estimates \( \Pi_t \) to compute conditional mean estimates of the target velocity \( \dot{x}, \dot{y} \) in the horizontal and vertical directions. These velocity estimates are then used to find the direction of motion of the target. The possible directions of motion of the target are quantized into 8 radial angular directions from the set \( \mathcal{Q} = \{\vec{a} = 0, \vec{c} = \frac{\pi}{4}, \vec{b} = \frac{\pi}{2}, \vec{f} = \frac{3\pi}{4}, \vec{c} = \pi, \vec{g} = \frac{5\pi}{4}, \vec{d} = \frac{3\pi}{2}, \vec{h} = \frac{7\pi}{4}\} \). The radial directions are shown in Fig. 4b and each is labeled for notational convenience with a lowercase alphabet and an additional “\( \vec{\phantom{a}} \)” to denote that is a unit directional vector. The velocity tracklet estimator thus outputs

\[
z_t = \arg\min_{q \in \mathcal{Q}} |\arctan\left(\frac{\dot{y}_t}{\dot{x}_t}\right) - q|,
\]

where \( z_t \) are the “true” velocity directions (generated by an appropriate SCFG model in \( \mathcal{G} \)). Moreover, these velocity tracklets are also noisy due to quantization and tracking errors. This issue will be addressed in the sequel.
III. Trajectory Modeling and Inference Using Stochastic Context-Free Grammars

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

A. Review of Stochastic Context-Free Grammars

In this section, we briefly review stochastic context-free grammars. A textbook treatment can be found in [11]. A context-free grammar $G_{CFG}$ is a 4-tuple $(\mathcal{N}, \mathcal{V}, S, A)$, where $\mathcal{N}$ is a finite set of non-terminals ($N_i, i = 1, \ldots, |\mathcal{N}|$), $\mathcal{V}$ is a finite set of terminals ($v_i, i = 1, \ldots, |\mathcal{V}|$) such that $(\mathcal{N} \cap \mathcal{V} = \emptyset)$, $S \in \mathcal{N}$ is the chosen start symbol (initial non-terminal) and $A$ is a finite set of production rules $a_m$ of the form $(A \rightarrow \alpha)$, $A \in \mathcal{N}$ and $\alpha \in (\mathcal{N} \cup \mathcal{V})^+$. The set $(\mathcal{N} \cup \mathcal{V})^+$ denotes all finite length strings of symbols in $(\mathcal{N} \cup \mathcal{V})$, excluding strings of length 0 (the case where strings of length 0 is included is indicated by $(\mathcal{N} \cup \mathcal{V})^*$). The $\rightarrow$ symbol denotes a re-write operation which replaces the non-terminal $A$ with the string $\alpha$. A stochastic context-free grammar is defined as a pair $(G_{CFG}, p)$, where $p : A \rightarrow [0,1]$ is a probability function over the production rules $(A \rightarrow \alpha) \in A$ such that $\forall A \in \mathcal{N}, \sum_{i=1}^{n_A} p(A \rightarrow \alpha_i) = 1$. The number of alternative production rules associated with $A$ is denoted $n_A$.

A grammar is a generative model which produces an output sequence of terminal symbols. A symbol refers to an element of the set $(\mathcal{N} \cup \mathcal{V})$ which can be either a single non-terminal or a single terminal. A sequence of such symbols is called a string $\alpha \in (\mathcal{N} \cup \mathcal{V})^+$. When the string is complete and composed entirely of terminals (such that no further symbols can be concatenated or produced in the string), it is called a sentence. The grammar generation process begins with the start symbol $S \in \mathcal{N}$. A rule $(S \rightarrow \alpha_1)$ is then chosen from the subset of rules in $A$ whose left-hand side is the start symbol $S$. This symbol $S$ is then replaced by the corresponding string $\alpha_1$ on
the right-hand side of the chosen rule. If $\alpha_1$ contains only terminal symbols, then the grammar generation process terminates. In this case, the sentence generated by the grammar $G_{CFG}$ is the string $\alpha_1$. However, if $\alpha_1$ contains non-terminals, then each non-terminal is replaced by a new string according to the choice of a production rule in $A$ until all non-terminals are consumed by the generation process. This can be diagramatically represented as a parse tree as shown in Fig. 5. A parse tree is a graphical representation of the string generation process in which non-terminal symbols branch out through the choice of production rules until only terminal symbols are obtained. The non-terminal symbols form the root nodes of the parse tree while the terminal symbols form the leaves of the parse tree. The resulting final string of terminals is considered to be the sentence generated by the grammar. The set of all such sentences which can be generated by a particular grammar $G_{CFG}$ is termed a language $L_{CFG}$.

In the rest of the paper, only left-most derivations of strings are considered. A left-most derivation implies that grammar generation occurs by re-writing the left-most non-terminal in each generation step. Formally, a left-most derivation of a given string $x$ is denoted by $d(x) = (a_1, a_2, \ldots, a_M)$ which is a sequence of production rules $a_m \in A, m = 1, \ldots, M$ such that $(S \xrightarrow{a_1} \alpha_1 \xrightarrow{a_2} \alpha_2 \xrightarrow{a_3} \ldots \xrightarrow{a_M} x)$. Each $a_m \in A$ re-writes the left-most non-terminal in $\alpha_{m-1} \in (N \cup V)^+$. Each left-most derivation $d(x)$ is associated with a corresponding single parse tree $\psi(x)$ [12].

The probability of a parse tree $\psi(x)$ is the same as the probability of the associated left-most derivation $d(x)$, given by

$$P_{G_{CFG}} \{x, \psi(x)\} = P_{G_{CFG}} \{x, d(x)\} = \prod_{m=1}^{M} p(a_m).$$

The context-free nature of the stochastic grammar results in conditional independence on the choice of the production rules used in the derivation. In other words, the choice of a particular production rule for a non-terminal in the string is independent of the other symbols present in the string. As a result, the probability of the entire derivation can be factored into the probability of individual rule choices as indicated in (8).
be computed as

\[
P_{G_{CFG}}(\{x\}) = \sum_{\psi(x) \in \Psi(x)} P_{G_{CFG}}\{x, \psi(x)\}
\]

\[
= \sum_{d(x) \in D(x)} P_{G_{CFG}}\{x, d(x)\},
\]

where \(\Psi(x)\) represents the set of all possible parse trees that can be used to explain a certain sentence. Similarly, \(D(x)\) represents the set of all possible left-derivations that could generate the sentence \(x\). A sentence is termed \textit{unambiguous} if only one parse tree (or corresponding derivation) could explain its generation. If multiple parse trees can generate the same sentence, it is called an \textit{ambiguous} sentence. A grammar is termed unambiguous only if all its sentences are unambiguous. This ambiguous nature makes stochastic grammars interesting because a probabilistic ranking can be used for discriminative and predictive purposes.

\section*{B. SCFG Models for Anomalous Trajectories}

In this section, we model various trajectories of interest using stochastic context-free grammar models. We make the assumption that certain kinds of trajectory shapes are correlated with target intent. This can be seen, for example, in the case of a target making a U-turn. The associated \(U\) shape (henceforth called an \textit{arc}) of the trajectory can be approximated as a trapezoidal shape as shown in Fig. 2a. An SCFG can be used to model all trajectories of such a characteristic shape (with implicit scale-invariance) and can thus be used to recognize \(U\)-shaped trajectories. While \(U\)-turns are common trajectories to observe in civilian traffic, they generally imply anomalous behavior in military targets. Another example of target intent involving arc-shaped trajectories is a tank formation maneuver called a \textit{pincer} operation \cite{13}. The second shape of interest in this paper, is that of rectangles. Most man-made structures like roads and buildings, constrain the path of targets to have a rectangular shape. We show that an SCFG model can be constructed for such rectangular trajectories to which we can attach intent depending on the surveillance environment. Moreover, we can also construct models for closed trajectories and the move-stop-move trajectory.

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

\textbf{Line Trajectory:} A target traveling in a straight path creates linear trajectories with local Markov dependency, and it is characterized by rules of the form \(S \rightarrow \bar{a}S \mid \bar{a}\) with \(\bar{a}\) representing the target’s direction of motion. An example string of a target traveling in a straight horizontal line for four sampling instants is “aaaa”. The production rules of a line grammar generates a language that is equivalent to that of a hidden Markov model formulation \cite{14} (or equivalently a regular grammar). A regular grammar is constrained to have only one non-terminal on either side of a production rule. The linear shape can be represented as the language \(L_{\text{line}} = \{x \in \bar{a}^n\}\). This notation implies
that all strings generated by a line grammar $G_{\text{line}}$ will have the form $\vec{a}^n$. The notation $\vec{a}^n$ implies that the terminal symbol $\vec{a}$ appears $n$ times consecutively in a sequence. The two other geometric shapes of interest are arcs and rectangles. These shapes possess long range and self-embedding dependencies that require production rules which regular grammars (and hence Markov models) cannot represent.

**Arc Trajectory:** An arc-shaped trajectory can be expressed as a language $L_{\text{arc}} = \{ x \in \vec{a}^n \vec{b}^+ \vec{c}^n \}$, where there is an equal number of matching upward $\vec{a}$ and downward $\vec{c}$ tracklets and an arbitrary number of forward tracklets $\vec{b}$. The $+$ symbol denotes an arbitrary number of $\vec{b}$ symbols. The symbol $x$ represents any arbitrary string belonging to the language. Such a language can be generated by the grammar shown in Fig. 6(a). The grammar can be constructed based on techniques reviewed in [15].

**Rectangular Trajectory:** The m-rectangle language (with associated grammar shown in Fig. 6(b)) is $L_{\text{m-rectangle}} = \{ \vec{a}^m \vec{b}^+ \vec{c}^m \vec{d}^+ \}$ and it can model any trajectory comprising of four sides at right angles (not necessarily a closed curve) with at least two opposite sides being of equal length. Why do we consider m-rectangles instead of rectangles? This is because the language comprising of only rectangles is not context-free. The language comprising of only rectangles can be generated by a more specific class of grammars called context-sensitive grammars. A proof of this can be found in [15]. As a result, algorithms of polynomial complexity for recognizing such trajectories cannot be constructed. However, we can construct heuristic rules by fixing the number of $\vec{b}$'s and $\vec{d}$'s in the grammar to generate rectangles and squares. More importantly, we can quantize the tracklets more coarsely to generate closed trajectories.

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

**Move-Stop-Move Trajectory:** A move-stop-move trajectory results from another coarse representation of the tracklets which allows us to model a common evasion tactic used by targets. If the target stops moving (or its velocity drops below a threshold), then the tracker is unable to track it. As a result, targets seeking to evade a radar often intersperse periods of movement with periods of no movements. The sporadic stopping between two periods with movement can be modeling as a self-embedding grammar of the form shown in Fig. 6(c). A move-stop-move trajectory in which the target stops for four sampling instants would take the form “$\text{mmmmxxxxmmm}$”, where each $m$ refers to a movement in any one of the directions in $Q$ and $x$ refers to a stop.

As seen in Fig. 6, the rules needed to generate shapes such as arcs and m-rectangles have a syntax that is more complex than a regular grammar (because they have more than one non-terminal on the right hand side). These grammars are self-embedding context free grammars that cannot be represented by a Markov chain. A context-free grammar is self-embedding if there exists a non-terminal $A$ such that $A \Rightarrow \eta A \beta$ with $\eta, \beta \in (N \cup Q)^+$. The relation $\Rightarrow$ in simpler terms, such a notation is used to signify that even though there might not exist a production rule of the exact form $A \Rightarrow \eta A \beta$, existing production
rules can be used on each other to come up with such a production rule. For the arc production rules presented, the self-embedding property can be seen by the second production rule $X \rightarrow AXC$ where $X$ can repeatedly
call itself to lengthen the output string.

C. Well-posedness conditions for SCFGs

In this section, we provide two constraint conditions which ensure that the SCFG is well-posed. The first condition is called the consistency constraint which ensures that the grammar is able to terminate in a finite length string. The second condition constrains the final destination of the target in an average sense. These constraints result in a system of multi-linear inequalities

\[ f(p_1, p_2, \ldots, p_{|P|}) \leq 0, \]

where \( f(\cdot) \) refers to a multi-linear function and \( p_1, p_2, \ldots, p_{|P|} \) refer to the unknown production rule probabilities of the grammar. Multi-linear equations are known to be non-convex. However, due to the nature of the grammar, a feasible solution set can always be found in our case.

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

\[ S_t = \sum_{\tau=1}^{t} \kappa z_\tau. \quad (10) \]

The co-ordinate system used is the Cartesian plane which is represented by the unit vectors \( \vec{a} \) and \( \vec{b} \) as shown in Fig. 4b. Consequently, every trajectory consisting of a sequence of tracklets \( z_t \) can be written as an integer combination of \( \vec{a} \) and \( \vec{b} \). Each of the other unit directional vectors can also be written as a linear combination of \( \vec{a} \) and \( \vec{b} \) using

\[
\begin{align*}
\vec{c} &= -\vec{a}, \\
\vec{d} &= -\vec{b}, \\
\vec{e} &= \frac{1}{\sqrt{2}}\vec{a} + \frac{1}{\sqrt{2}}\vec{b}, \\
\vec{f} &= -\frac{1}{\sqrt{2}}\vec{a} + \frac{1}{\sqrt{2}}\vec{b}, \\
\vec{g} &= \frac{1}{\sqrt{2}}\vec{a} - \frac{1}{\sqrt{2}}\vec{b}, \\
\vec{h} &= \frac{1}{\sqrt{2}}\vec{a} - \frac{1}{\sqrt{2}}\vec{b}.
\end{align*}
\]

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

\[
S_T = \sum_{t=1}^{T} z_t = (x_T - x_1)\vec{a} + (y_T - y_1)\vec{b},
\]

where \(T\) is the total length of the string and \(N_{q_j}\) is the number of times the target moves in direction \(q_j \in Q\). Our interest is from a modeling perspective and hence we would like to constrain the total distance the target travels. It turns out that the probabilistic nature of the grammar production rules allows us to constrain the total distance traveled if we can bound the expected value of the total distance traveled. This is obtained from (12) by replacing each quantity with its expected value viz., the expected number of times \(E\{N_{q_j}\}\) that the target travels in each of the directions \(q_j \in Q\). The computation of the required expectations is detailed in the Appendix.

D. Bayesian Signal Processing of SCFG models

Given an observation sequence of the target’s estimated velocity directions \(z = z_1, \ldots, z_t\), can we classify the target’s trajectory? The set of permissible grammar models is given by \(\mathcal{G}\) which contains all the anomalous trajectories described in Sec. III-B. Mathematically, we seek the grammar posterior probability

\[
G^* = \arg \max_{G_k \in \mathcal{G}} P\{G_k|z_1, \ldots, z_t\},
\]

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

\[
P\{z_1, \ldots, z_t; G_k\} = \sum_{y \in (\mathcal{N} \cup \mathcal{V})^*} P\{z_1, \ldots, z_t, y; G_k\}.
\]

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

REFERENCES


