Meta Level Tracking with Multimode Space-Time Adaptive Processing of GMTI Data

Alex Wang and Vikram Krishnamurthy
Electrical and Computer Engineering
University of British Columbia
alexw@ece.ubc.ca and vikramk@ece.ubc.ca

Bhashyam Balaji
Radar Systems
Defence R&D Canada Ottawa
Bhashyam.Balaji@drdc-rddc.gc.ca

Abstract – Ground surveillance of the battlefield provides military analysts with information that is critical to the success of a mission; the type of the information includes the enemy force structure, enemy offensive combat formation, and maneuvering events. The conventional approach uses mainly the synthetic aperture radar (SAR) and electro-optical (EO) sensors to perform detection and identification of stationary targets on the battlefield. Ground moving target indicator (GMTI) radar with space-time adaptive processing (STAP), on the other hand, allows a more complete perception of the battlefield by adding the capability to detect moving objects over a large area. In particular, the simultaneous detection and estimation of angular location of a ground moving target via adaptive cancellation of ground clutter is demonstrated, where a single reflector antenna with a multimode feedhorn is used in a GMTI radar. Based on the GMTI radar output, we illustrate the use of stochastic parsing algorithm with stochastic context free grammar (SCFG) as an unifying framework for data association, target tracking, and situation awareness.

Keywords: Ground Moving Target Indicator (GMTI), Space-Time Adaptive Processing (STAP), Stochastic Parsing, Intent tracking, Bayesian Inference

1 Introduction

In order to develop battlefield awareness, military analysts need a great deal of information; for example the enemy force structure, enemy offensive combat formation, and enemy’s maneuvering events. However, the conventional sensors deployed are mainly the synthetic aperture radar (SAR) and the electro-optical (EO) sensors, and they are limited in the sense that they are only capable of performing detection and identification of stationary targets. Relatively recently, ground moving target indicator (GMTI) radar with space-time adaptive processing (STAP) enables the near-real time detection of ground moving objects over a large area. This capability allows the radar operator to follow the trajectories of the enemy units, and in turn, determine the tactical intents of the units.

In this paper, we develop a meta level tracker based on the GMTI radar with multimode STAP (Details of the multimode STAP can be found in [4]). Based on these GMTI reports, tracking has to be performed in order to develop battlefield awareness. Currently, the tracking algorithm focuses mainly on state-space model, where the kinematics of a single or multiple targets are recursively estimated with algorithms such as Kalman, HMM, or VS-IMM [5, 6]; VS-IMM is the more sophisticated model where the kinematic model of the moving object depends on the road direction and the terrain type. With these models, the capability of near-real time tracking of the ground moving units can be accomplished. However, this is often not sufficient for battlefield awareness. For tracking moving units on a battlefield, the inference of their tactical intents is often more appreciated. Even though the state space filters can estimate the moving units’ position and velocity, they are incapable of distinguishing, for example, a circular track from a linear one; the importance of such “labeling” can be exemplified by a forward quarter intercept or a pincer [7] which is characterized by two cooperating vehicles maneuvering in arcs.

The state space based models, in general, involves segmenting a trajectory a sequence of states and analyzing the transitions between them with Markov assumption. However, because the vehicles’ trajectory has complex spatial patterns that are not Markov, for example the long range dependency in an arc, we need a more general framework. We will demonstrate that the stochastic context free grammar (SCFG) is a suitable choice. SCFG has been a major computational tool in language and speech analysis, and DNA sequence aligning in bioinformatics [8, 9]. It can capture the long range dependencies and the recursively embedding structures in patterns by allowing more complicated rules than the Markov models. As is known from
formal language theory, SCFG is more general than a Markov model and it has higher modeling power.

The main results of the paper are: 1) A SCFG model characterizing the ground vehicles’ motion patterns, 2) Extended Earley-Stolcke algorithm performing data association, tracking, and situation awareness on GMTI data, and 3) Implementation of the parsing algorithm in C++ that allows numerical studies of battlefield awareness with stochastic parsing. The paper is organized as follows: Sec. 4 describes the overall system framework, Sec. 4 details its implementation with stochastic parsing, Sec. 5 provides the numerical studies, and Sec. 6 concludes the paper.

2 Architecture of the Battlefield Awareness System

The objective of our study is to determine the tactical plans of the ground moving units, and the basic assumption is that tactical plans consist of motion patterns that can be recognized by syntactic pattern recognition techniques. The motion pattern can be decomposed into a fixed set of basic geometric patterns such as a line or an arc. For example, Fig. 1 illustrates some common battalion formations, and each is characterized by certain types of motion patterns [10]; the line abreast and wedge formation are offensive combat formations with each vehicle moving in linear trajectory; pincer, on the other hand, consists of two vehicles maneuvering in mirroring arc trajectories. With the geometric pattern labeling, it is possible to determine if the enemy is in offensive, defensive or reconnaissance operation, and it could provide the commander with real time analysis of the battlefield situation.

Based on the GMTI detections and estimations, the aim of the battlefield awareness system is to map a ground moving target’s radar returns to its possible tactical intent. If we examine multiple frames of the GMTI data over time, the ground vehicles’ intent could be inferred by studying their motion patterns. Under this methodology, the battlefield awareness problem is reduced to a pattern recognition problem, which, in general, can be classified into two major categories: decision theoretic and syntactic pattern recognition [11]. The decision theoretic approach extracts features from the patterns, and perform pattern recognition by classifying the pattern according to the partitioned feature space. The syntactic approach, on the other hand, decomposes a pattern to its constituent sub-patterns, and recognizes the pattern by building its syntactic structure. However, because of the complexity of the battlefield, and the lack of training data, pure decision theoretic approach is not practical as the number of features required is often very large. As a result, the syntactic approach that describes a complex pattern with simpler sub-patterns in a hierarchical fashion is very attractive; the knowledge of sub-patterns is often difficult to learn, but is priorly known by military analysts.

In this paper, we will apply a mixed approach. The basic idea is to study the syntactic structure of the trajectory by decomposing it into tracklets and analyze the temporal dependencies among them; the decision theoretic approach is used to determine the tracklets, and the syntactic approach is applied to analyze the dependencies. One simple example illustrating the syntactic structure of the trajectory is given in Fig. 2: Fig. 2a) shows the set of the primitive motion features (tracklets) assumed in the paper, and Fig.2b) shows a possible trajectory realization and its syntactic analysis. It should be noted that the trajectory realization in terms of tracklets is estimated based on a generative decision theoretic model. The syntactic analysis is called stochastic parsing, and the inferred syntactic structure is a parse tree.

The battlefield awareness system aims to take the radar returns of a ground moving target and constructs a parse tree describing the geometric pattern of its trajectory. The framework of the system is illustrated in Fig. 3, and it consists of five main components: the GMTI STAP processor, a signal-to-symbol filter, a motion pattern knowledge-base, an intent inference parser, and a radar resource allocator. The GMTI radar detects ground moving targets and returns their estimated range, angle, and Doppler. The signal-to-symbol filter keeps a track of the detected target, and continuously outputs its associated tracklet that best describes its kinematic state. The motion pattern knowledge-base stores the prior knowledge of the relevant motions in terms of production rules. Intent inference parser, based on the syntactic structure of the tracklets and
3 Formal Grammar Modeling of GMTI

The section describes the developed intent inference algorithm that incorporates the SCFG modulated state space model and STAP detection. The discussion will be top-down in terms of the system architecture; Sec. 3.1 formulates the syntactic pattern description of the motion patterns with SCFG, and Sec. 3.2 discusses signal-to-symbol filter, i.e. the tracking of the moving target and the generation of the SCFG primitives with the IMM/Extended Kalman filter.

3.1 Syntactic Pattern Description of Motion Patterns

The knowledge-base of motion patterns is characterized by a grammar $G$. $G$ is a four-tuple $<N, T, P, S>$ [12, 13]. $N$ is a finite set of nonterminals, $T$ is a finite set of terminals; $N \cap T = \emptyset$. $P$ is a finite set of production rules, and $S \in N$ is the starting symbol. The grammars are divided into four different types according to the forms of their production rules [13, 12]. The grammar of interest in this paper is the SCFG, whose production rules have the form $A \rightarrow \eta$ with probability $P(A \rightarrow \eta)$ where $A \in N$ and $\eta \in (N \cup T)^+$; $\Sigma^+$ indicates the set of all finite length strings of symbols in $\Sigma$, excluding strings of length 0. The meaning of the production rule can be seen from an example: Suppose we have a concatenated pattern $xAy$, where $x$ and $y$ are any combination of nonterminals and terminals, and $A$ is a nonterminal, a one step derivation using the rule $A \rightarrow \eta$ yields $xAy \rightarrow xyy$.

The correspondence of the grammar and the geometric motion patterns of the ground moving targets is as follows: The tracklet estimated is modeled by the terminals, the labeling of the geometric patterns by the nonterminals, and the hierarchical structure of the tactical moves by the production rules. More specifically, one possible nonterminal set of geometric patterns is $\mathcal{L} = \{\text{line (L), arc (A), square (Sq)}\}$, and how they expand to form patterns is specified by the production rules, which are specified as follows:

$$
T = \{a, b, c, d, e, f, g, h\} \\
N = \{L_a, L_b, \ldots, L_h, A_{ur}, A_{ul}, S_{qr}, S_{qt}, S\}, \\
P = \{S \rightarrow L_a \mid L_b \mid \ldots \mid L_h \mid A_{ur} \mid A_{ul} \mid S_{qr} \mid S_{qt} \mid S\}, \\
L_u \rightarrow u \mid L_u \mid u \quad \text{for } u \in T, \\
A_{ur} \rightarrow aA_{ur}g \mid hA_{ur}h \mid aq \mid hh \mid h, \\
A_{ul} \rightarrow gA_{ul}a \mid hA_{ul}h \mid ga \mid hh \mid h, \\
S_{qr} \rightarrow \text{TOP UD BOT DU} \\
S_{qt} \rightarrow \text{BOT DU TOP UD} \\
\text{TOP} \rightarrow L_a \mid L_g \\
\text{BOT} \rightarrow L_a \mid L_g \\
\text{UD} \rightarrow L_e \\
DU \rightarrow L_a \}
$$

The terminal set $T$ is the set of tracklets used to track the ground moving targets, and it is illustrated pictorially in Fig. 2a). The nonterminal $L_u$ generates lines in the direction $u$ for all $u \in T$, and $A_{ur}(A_{ul})$ generates an arc pointing upward and to the right (left). $S_{qr}$ and $S_{qt}$ are the clockwise and counter-clockwise square respectively, and the nonterminals TOP, UD, BOT and DU represent the top, up-down, bottom, and down-up components of a square. It should be noted that the grammar is a small subset for illustrative purpose, where only one orientation of the arcs and the squares is included; it is straightforward to include other orientations.

Let $t$ and $t'$ be any terminal, and $N$ and $N'$ be any nonterminal. It should be noted that the rule with the form $N \rightarrow nN'$ models the local Markov dependency; the emission of the output symbol $n$ as the system transitions from the state $N$ to $N'$. The long range and self-embedding dependencies found in the pattern such as an arc, on the other hand, has the form $N \rightarrow t NN't \mid tt'$. This arc motion pattern is particularly interesting because it is a palindrome, i.e., a sequence of symbols that has the same order in both the forward and backward directions. Such motion is difficult for Markov models to classify because Markov models cannot just generate palindromes [12]. Moreover, SCFG also allows classification based on template matching. The square motion pattern described above demonstrates...
how a more abstract pattern can be specified in a template consists of more primitive patterns. The ability of the SCFG to model Markov dependency, long range and self-embedding dependency, and template matching shows its modeling power, and it also demonstrates the user-friendliness of the modeling technique to non-experts in the signal processing field.

### 3.2 Signal-to-Symbol Bayesian Filter

The grammar’s terminal set $T$ is the set of tracklets as illustrated in Fig. 2a), and this section describes the signal-to-symbol Bayesian filter that maps the GMTI STAP measurement to one of the tracklets in $T$. This mapping is formulated with the concept of directional process noise and the interacting multiple model (IMM). In the usual case, assuming Gaussian noise, the standard kinematic models assume equal variance for the process noise in all unit directions because they assume equal probabilities among the unit directions at which the target is moving. In order to model the tracklets, on the other hand, the process noise is assumed to have different noise variance along and perpendicular to the direction of the particular tracklet. More specifically, each tracklet is modeled by an IMM model whose process noise has variance that is inherent to that tracklet. Details of the formulation is given below.

Let $k$ denotes discrete time, the assumed target dynamics is

$$\mathbf{x}_k = F \mathbf{x}_{k-1} + G v_{k-1}(t_k).$$

$\mathbf{x}_k = (x_k, y_k, \dot{x}_k, \dot{y}_k)'$ denotes the ground moving target’s position and velocity in Cartesian coordinates, and assuming constant velocity model, the transition matrix model and the noise gain are, respectively,

$$F = \begin{pmatrix} 1 & 0 & T & 0 \\ 0 & 1 & 0 & T \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}, G = \begin{pmatrix} T^2/2 & 0 \\ 0 & T^2/2 \\ 0 & 0 \\ 0 & T \end{pmatrix}.$$

The variable $t_k \in \{0, \pi/4, \pi/2, 3\pi/4, \pi, 5\pi/4, 3\pi/2, 7\pi/4\}$ denotes the tracklets in the terminal set, and it indexes the moving target’s possible directions of travel. The estimation of $t_k$ is the main objective of the signal-to-symbol filter, because it is the basic building blocks making up the motion patterns. As will be shown in Sec. 4, the intent inference parser will parse the estimated $t_k$, and estimate the motion patterns of the target. The process noise $v_k$ is a white Gaussian process with the covariance matrix

$$\mathbf{Q} = \rho_{t_k} \cdot \begin{pmatrix} \sigma_a^2 & 0 \\ 0 & \sigma_a^2 \end{pmatrix} : \rho_{t_k}^T,$$

with $\rho_{t_k} = \begin{pmatrix} -\cos t_k & \sin t_k \\ -\sin t_k & -\cos t_k \end{pmatrix}$, where $\sigma_a^2$ is the uncertainty along the direction indicated by $t_k$ and $\sigma_a^2$ is orthogonal to it.

The observation model describes the output of the GMTI STAP measurements, and the observation is

$$z_k = h(x_k) + w_k,$$

where

$$h(x_k) = \begin{bmatrix} r_k \\ \dot{r}_k \\ \theta_k \end{bmatrix} = \begin{bmatrix} \sqrt{x_k^2 + y_k^2 + z_k^2} \\ \frac{x_k + y_k z_k - 2 \dot{z}_k}{\sqrt{x_k^2 + y_k^2 + z_k^2}} \\ \arctan(\frac{y_k}{x_k}) \end{bmatrix}. \quad (1)$$

$r_k$ is the range, $\dot{r}_k$ is the Doppler, $\theta_k$ is the azimuth angle, and $w_k \sim N(0, R)$. The covariance matrix $R$ is a diagonal matrix with the diagonal elements equal to the variances in range, range rate, and azimuth angle, which are denoted as $\sigma_{r_k}^2$, $\sigma_{\dot{r}_k}^2$, and $\sigma_{\theta_k}^2$, respectively. Moreover, in order to compensate for the radar’s platform motion, $x_k' = x_k - x_k^0$ where $x_k^0$ is the $x$ coordinate of the sensor platform at time $k$; similarly for $y_k$ and $z_k$. However, one issue with Equation 1 is that it is highly nonlinear and extended Kalman filter is needed to process the observations. In order to minimize the linearization required, the measurement model is converted and the more preferred measurement model is as follows:

$$z_k' = h'(x_k) + w_k'$$

where

$$h'(x_k) = \begin{bmatrix} r_k \sin \theta_k \\ r_k \cos \theta_k \\ \dot{r}_k \end{bmatrix} = \begin{bmatrix} x_k \\ y_k \\ \dot{r}_k \end{bmatrix} \quad (2)$$

and $w_k' \sim N(0, R')$ is the measurement noise in the converted model. The converted covariance matrix is

$$R' = \begin{pmatrix} \sigma_{x_k}^2(R') & \sigma_{xy}(R') & 0 \\ \sigma_{xy}(R') & \sigma_{y_k}^2(R') & 0 \\ 0 & 0 & \sigma_{\dot{r}_k}^2 \end{pmatrix},$$

whose elements are

$$\sigma_{x_k}^2 = r_k^2 \sigma_{\dot{r}_k}^2 \cos^2 \theta_k + \sigma_{\dot{r}_k}^2 \sin^2 \theta_k,$$

$$\sigma_{xy}^2 = (r_k^2 - r_k^2 \sigma_{\dot{r}_k}^2) \sin \theta_k \cos \theta_k,$$

$$\sigma_{y_k}^2 = r_k^2 \sigma_{\dot{r}_k}^2 \sin^2 \theta_k + \sigma_{\dot{r}_k}^2 \cos^2 \theta_k.$$

In order to run extended Kalman filter, the Jacobian of the converted measurement function is needed and it is given by

$$\nabla_x h'(x_k) = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix},$$

which can be computed by differentiation.

As will be shown in Sec. 4, the terminal probability $w^j_k = P(t_k = j | z_{1:k})$ models the input uncertainty for the parsing process, and the position estimate $\mathbf{x}_{k|k}$ is stored in the low and high marks of the Earley state.
for enforcing consistency of the tracks. According to
the kinematic model, we can compute the two variables
based on the interacting multiple models (IMM) [5].

Recall that the purpose of the signal-to-symbol filter is
to map the GMTI STAP measurements to a tracklet.
Since the tracklet is defined by the IMM mode \( t_k \), given
the IMM filter output, the output of the filter can be
either \( w^i_k \) for all \( j \) or \( \operatorname{arg \max}_j w^i_k \), which are called soft
and hard parsing respectively.

### 4 Stochastic Parsing for Battle

field Awareness

Given the SCFG’s four tuples, the problem of battle-
field awareness is reduced to a parsing problem where
the moving target’s intent is summarized by a parse
tree. The objective of this section is to develop the intent
inference parser module as shown in the framework illustrated in Fig. 3. Recall from Sec. 3.2, the output of the signal-to-symbol filter is a sequence of IMM modes
and their probability distribution. The IMM mode
sequence is denoted as \( t_1, t_2, \ldots, t_k \), where \( k \) is the discrete time,
and the parse tree will be built on top of it.
In the battlefield awareness scenario, because the IMM
modes only arrive as the process unfolds in the battle-
field, the Earley-Stolcke parsing algorithm [14, 8] is
chosen as it can parse data from left to right recursively.
In this section, the parser and its interactions with the
IMM/extended Kalman filter will be described, including
the extensions required to equip the conventional parser
with the track maintenance functionalities. The four main extensions are 1) model the uncertainties of
the GMTI detection, 2) data association of concurrent
detections, 3) track initiation of target trajectory, and
4) trade-off between the completeness of the tracks and
the computation resources. The extensions are largely
based on those described in [8], but altered to fit the
specific case of GMTI.

Parsers can be roughly categorized into two types:
top down and bottom up. The Earley Stolcke parser is
a top down parser, and the control structure it uses to
store the incomplete parse trees is defined as [14, 15]

\[
i : X_k \rightarrow \lambda.Yu[l, h, \alpha, \gamma].
\]

\( X \) and \( Y \) are nonterminals, \( \lambda \) and \( u \) are substrings of
nonterminals and terminals, \( \cdot \cdot \cdot \) is the marker that specifies
the end position for the partially parsed input and
that position is indexed by \( i \), \( k \) is the starting index
of the substring that is generated by the nonterminal \( X \). \( l \)
is the kinematic state of the moving target at the
beginning of the track, and \( h \) is the newly estimated
kinematic state. Fig. 4 illustrates an example state
\( i : X_k \rightarrow A.B \), where \( A \) and \( B \) are nonterminals; the
indices \( k \) and \( i \) specifies the beginning and the end of
the substring respectively for which the nonterminal \( X \)
can "explain" so far, and the index marker \( \cdot \cdot \cdot \) specifies
the part of \( X \)’s production rule that has been applied
to explain the substring. With the marker in front of
the nonterminal \( B \), \( B \) is not yet applied, and the state
is still incomplete. It should be noted that, without
any modification, the parser generates all the possible
states and match them with the input terminals, and
those not matching will be discarded. \( \alpha \) is called for-
ward probability and it is the sum of probabilities of
all incomplete parse trees containing \( t_1, \ldots, t_i \), and \( \gamma \) is
called inner probability and it is the sum of probabil-
ities of all incomplete parse trees containing \( t_k, \ldots, t_l \).

Based on this definition, a similarity function \( f(d) \) is
introduced to measure the consistency of the kinematic
states and it provides valuable information for the im-
plementation of data association and rejection of false
detection. Many models may be applied to exploit the
spatial correlation [16], and in this paper, power expo-
nential function, \( f(d) = \exp\left(-\frac{d^2}{\theta^2}\right) \), is applied, where
\( \theta_1 > 0 \) and \( \theta_2 \in (0, 2] \) are determined experimentally,
and \( d \) is the euclidean distance between the kinematic
states. The production rule, on the other hand, is mod-
ified to model false detection. For every production rule
that involves the generation of terminals, a nonterminal
\( F \) is added. For example, the rule \( L \rightarrow LF.L \) is mod-
ified to include \( L \rightarrow LF.L \), and \( F \rightarrow a[b|c|d|e|f|g|h \).

The rule simply states that if the terminal at the loca-
tion is a false detection, replace it with \( F \) and carry on
the parsing as before.

To illustrate the parsing steps, a simple example of
parsing a very short input string "bb" is provided.
To initialize the parsing process, the dummy state
\( 0 : 0 \rightarrow .S[l_c, h_c, 1, 1] \) is inserted; the dummy state sim-
ply says that at the index position 0, the start symbol
is applicable to parse the input string, and \( l_c \) and \( l_h \)
are the extracted kinematic states of the target from the
STAP detection hits. With the dummy state in place,
the Earley Stolcke parsing algorithm starts building the
parse tree by iteratively going through three operations:
prediction, scanning, and completion. The operations
are applied sequentially, and each operation works on
the set of states produced by the previous operation.
The parsing of the string "bb" is illustrated in Table 1.
Given a set of states (or just the initial dummy state
at index 0), the prediction operation searches for states
whose index marker has a nonterminal to the right of
it, and those nonterminals, with their production rules,
are used to generate a set of predicted states. From the predicted states, the scanning operator looks if there are states whose index marker has a terminal to the right of it. For those states whose terminal matches the input string at the indexed position, their index marker is advanced by one position and produced as a scanned state. Lastly, from the set of scanned states, the completion operation looks if there are states whose index marker is at the end of its production rule. If any are found, the states that generated those scanned states will have their index advanced by one position. The details of the three operations are discussed in turn.

The **Prediction** operator adds states that are applicable to explain the unparsed input string. For all states of the form

\[ i : X_k \rightarrow \lambda.Yu [l,h,\alpha,\gamma] \]

where \( \lambda \) and \( u \) may be empty, \( Y \) is the nonterminal that could possibly generate the next terminal in the input string. The operator adds \( Y \)'s production rule, assume it's \( Y \rightarrow v \), as a predicted state

\[ i : Y_i \rightarrow .v [l,h,\alpha',\gamma'] \]

The \( \alpha' \) and \( \gamma' \) are updated according to

\[ \alpha' = \sum_{\lambda,u} \alpha(i : X_k \rightarrow \lambda.Zu)R_L(Z,Y)P(Y \rightarrow v) \]

and

\[ \gamma' = P(Y \rightarrow v), \]

where \( R_L \) is a reflective transitive closure of a left corner relation and it computes the probability of indefinite left recursion in the productions. (The detail of the relation is omitted as it has little significance in this paper. Interested readers can refer to [14].) The new predicted state inherits the kinematic states because it explains the same portion of the target track. The pruning capability of the parser can be implemented here; the predicted states can be discarded if its forward probability is lower than a threshold. The value of the threshold is a system parameter that balances the loading of the system and the completeness of the tracks. In addition, the prediction stage is a good place for implementing the functionality for the parser to capture unknown beginning of vehicle's motion trajectory. At each time instant when the prediction operation is run, a dummy state of the form \( \forall k \ k : k \rightarrow .S \) can be inserted if there are STAP detection hits that cannot be associated with any partial parse tree (The data association is implemented next in the scanning operation). With this dummy state, the parser is not limited to capture patterns that were started at the time instant 0, and it allows the prediction operator to play the role of track initiation for the track maintenance.

The **Scanning** operator matches the terminal in the input string to the states generated from the prediction operator. For all states of the form

\[ i : X_k \rightarrow \lambda.au [l,h,\alpha,\gamma] \]

where \( \lambda \) and \( u \) can be empty, the state

\[ i + 1 : X_k \rightarrow \lambda.a.u [l,x_a,\alpha',\gamma'], \]

is added if the terminal at the index \( i + 1 \) of the input string is \( a \). The \( \alpha' \) and \( \gamma' \) are updated according to

\[ \alpha' = \alpha(i : X_k \rightarrow \lambda.au)P(a) \]

and

\[ \gamma' = \gamma(i : X_k \rightarrow \lambda.au)P(a), \]

where \( P(a) \) is the probability of the input. It is noted that by including \( P(a) \) in updating \( \alpha \) and \( \gamma \), the parsing process also takes the input uncertainty into account. Moreover, \( X_k \neq F \) (Recall \( F \) is the added nonterminal signifying false detection) if \( f(lh = x_a) \) is greater than a threshold, and \( X_k = F \) otherwise, \( x_a \) is the kinematics state of the terminal \( a \) estimated by the IMM/Extended Kalman filter. More specifically, the similarity function is combined with the false detection nonterminal to implement the nearest neighbor data association filter. More complicated algorithms can be implemented but they're not pursued here. Moreover, the input uncertainty \( P(x_a) \) models the STAP detection probability.

The **Completion** operator advances the marker position of the pending predicted states if their derived states match the input string completely. The operator, if observed the scanned state

\[ i : Y_j \rightarrow .v [l_2,h_2,\alpha'',\gamma''] \]

and the pending prediction state

\[ j : X_k \rightarrow \lambda.Yu [l_1,h_1,\alpha,\gamma], \]

will generate the complete state

\[ i : X_k \rightarrow \lambda.Yu [l_1,h_2,\alpha',\gamma']. \]
The purpose of the completion operator is to find those pending prediction states and advance their marker. It is important to notice the relationship of the indices in the scanned state and the pending prediction state. The indices of the pending prediction state says the nonterminal \( Y \) is applied at the position \( j \), and the indices of the scanned state says the nonterminal \( Y \) matches the substring from the index \( j \) to \( i \). As a result, the two states generate the complete state that says the pending prediction state can now explains the substring from the index \( k \) to \( i \). The associated \( \alpha \) and \( \gamma \) probabilities are updated according to

\[
\alpha' = f(h_1, h_2) \sum_v \alpha(i : X_k \rightarrow \lambda.Zu)R_U(Z,Y)\gamma''(i : Y_j \rightarrow v).
\]

and

\[
\gamma' = f(h_1, h_2) \sum_v \gamma(i : X_k \rightarrow \lambda.Yu)R_U(Z,Y)\gamma''(i : Y_j \rightarrow v).
\]

respectively, where \( R_U \) is a reflective transitive closure of a unit production relation and it computes the probability of an infinite summation due to cyclic completions. (Interested reader can refer to [14] for more detail.) The similarity function here models the consistency between the pending prediction state and the completed state. If the likelihood probabilities of the completed state is lower than a threshold, it will be pruned to trade track completeness with computation reduction, and this implements the track termination functionality of the track maintenance.

5 Numerical Studies

To evaluate the algorithms described in this paper, DRDC sets up flight trials to collect GMTI measurements. The air-borne GMTI radar is used, and the aircraft platform is moving at a constant speed of 200 knots, or 100 m/s. The radar is a two channel radar with one transmitter and two receivers, and it operates with the Spotlight/SAR mode. (More information about the GMTI and the radar can be found in [4].)

The pulse repetition frequency of the GMTI radar is set to 1KHz. The ground moving target is a truck, and it is moving with trajectories that form various different geometric patterns. The derived radar output is range, range rate, and azimuth angle, and the elevation angle is neglected because it does not provide any extra information than the range measurement; the target is moving along on the ground, or moving on a known plane intersecting the sphere centered at the aircraft, and if the pointing angle and range resolution is known, a particular range bin is equivalent to an elevation angle for the target. The numerical studies in this section demonstrates how stochastic parsing with target tracking can discern the geometric patterns from the radar measurements, and in turn establish the battlefield awareness. The parsing algorithm is implemented in C++.

The tracking result of the meta level tracker is illustrated in Fig. 5 based on a run of the DRDC flight trials. However, it should be pointed out that the measurement data was augmented with GPS information of the airborne platform and the ground moving vehicle. This was because the application of STAP to the raw data did not yield measurements with enough accuracy, especially the angle, for the entire duration of the measurement. The inaccuracy is due to the many challenges of obtaining adequate clutter cancellation and consistently good parameter estimation of ground movers in a highly heterogeneous urban environment. However, the measurement data is still useful because it illustrates the meta-level tracking concepts. Fig. 5 illustrates the likelihood probabilities of different motion patterns as a square trajectory is parsed. The solid line of the figure on the bottom left panel is the measured GMTI track, and the dotted line is the output of the IMM/Extended Kalman filter. The bottom right panel shows the estimated IMM modes; only four modes are shown in this case for the ease of display. The top panel, on the other hand, shows the correct motion pattern maintaining its high probability as the probabilities of other patterns drop because the input sequence does not support them. The other motion patterns such as the vertical line and the clockwise square patterns also had high probabilities initially because the initial segment of the input terminal string matches their syntactic patterns. However, as more terminals are parsed, their probabilities drop.

The parsing results demonstrate how true geometric patterns...
patterns could be extracted by a maximum likelihood estimator. The long range dependency in the terminal strings is exploited to identify the patterns. The knowledge of the extracted geometric patterns could be feedbacked to the tracker for better state estimation.

Fig. 6 demonstrates the reduction in estimator covariance with the knowledge of the extracted geometric pattern; the solid line shows the covariance of the tracker as the target is moving in a square, and the dotted line is the covariance of the assisted tracker. The jumps in covariance correspond to the times when the target is making sharp turns, and the knowledge about the target trajectory’s geometric pattern allows the tracker to make better predictions of the turns, and thus reduce covariance.

6 Conclusion

We model the ground vehicles under the GMTI surveillance as string generating devices whose motion patterns are characterized by SCFG. The control structure of the Earley’s parsing algorithm is extended with the IMM/Extended Kalman filter to unify data association, target tracking, and situation awareness. The battlefield awareness is developed as GMTI data is collected and parsed, and which provides the commanders with real time information about the enemy’s possible intent and tactics. Lastly, the parsing algorithm is implemented in C++ for numerical studies of the algorithm, and the identification a square is demonstrated.

References


