Abstract—Maritime surveillance of coastal regions requires the processing of data from a large number of heterogeneous surveillance sources. The generation of effective Maritime Domain Awareness requires that the tracks from these sources must be fused. An automated fusion process that supports Maritime Domain Awareness requires that the tracks from heterogeneous sources be modeled in such a way as to support automated track association decision process. Collection of sensor contact reports are therefore grouped into tracks and a mathematical model of these tracks are used to support track correlation which leads to track association. This paper presents the approach for track modeling based on genetic algorithm optimization and discusses its performance.

Keywords: Track modeling, estimation, genetic algorithm, optimization, maritime surveillance, Maritime Domain Awareness.

I. INTRODUCTION

With increased interest in maintaining Maritime Domain Awareness (MDA), an increasing number of sensor and information sources are being employed to provide surveillance coverage of very large maritime areas. These surveillance systems can include the use of shore, underwater, ship, airborne and space based systems relying on radar, electronic support measures electro-optic systems, human observers, and vessel self reporting such as Automatic Identification System reports.

Since no one system can provide all the coverage and information required to maintain the desired MDA, it is necessary to employ a combination of many systems. These systems will have regions of overlapping coverage which will result in multiple tracks being reported for a single maritime platform. If the information from these systems is displayed as they are reported by each of these systems, the advantage gained by having multiple sources can soon be outweighed by the ambiguity resulting from having multiple reports on the same platform.

To achieve a single track for a single platform requires that the tracks from the sources must be reliably fused. A system performing level 1 fusion, as defined by the US Joint Directors of Laboratories Data Fusion Group, for maritime surveillance has been presented in references [1], [2]. The track-to-track fusion system described in [2] describes an ongoing research effort that is intended to automate the manual process described in [3]. The first step in this process is to associate tracks that correspond to the same vessels. Since many sources provide only kinematic information, this track association process may be based only on the track history of the vessel. A common track modeling approach must be applied to represent vessel paths in order to allow comparison of track history.

This paper proposes a track modeling approach developed to support track-to-track fusion in an extremely heterogeneous maritime surveillance environment. It justifies and proves the efficiency of this approach based on optimization with a genetic algorithm.

II. DATA DESCRIPTION

Much of the literature on track fusion (e.g. [4], [5]) addresses the problem from the perspective of improving track accuracy. In this paper we are interested in modeling maritime tracks in order to support the decision to fuse two tracks. The choice of modeling procedure is based on the following three objectives: Handling Varied Data Characteristics, Exploiting Predictable Ship Behavior, and Ease of Track-to-track Comparison.

1) Handling Varied Data Characteristics
Surveillance systems generally report the vessel’s position, course and speed along with a corresponding covariance matrix. Ideally this information would be reported in sequence at a common update rate and with similar accuracy. In the context of wide area maritime surveillance, the reporting systems rarely meet this criteria. Information can vary from near real-time (e.g. Space-based systems) to extremely time late (e.g. post flight reporting from airborne surveillance). A reporting system may provide course but no speed, positional covariances but no velocity covariance, positional covariances that differ by an order of magnitude or a position that is based on a human operator’s judgment. The required track modeling approach must therefore be extremely robust.

2) Exploiting Predictable Ship Behavior
Ships usually follow shipping lanes when transiting or remain in a specific region in the case of fishing vessels. In both cases they navigate using waypoints and their movement can be characterized by leg segments between
each waypoint. In the case of transiting vessels the legs are relatively long. In the case of vessels engaged in fishing activities the legs are relatively short. Also, transiting vessels travel at a predictable speeds in the vicinity of 18 knots while vessels engaged in fishing activity move at lower speeds of under 5 knots. The required track modeling approach must therefore be able to exploit these characteristics.

3) Ease of track-to-track Comparison

Although the surveillance systems have overlapping areas of surveillance, in general, the resulting tracks from these systems may overlap for only a small portion of the track histories or not overlap at all. One source may provide a track history that extends for hours and possibly days while the other track that it is being compared to may be based on as little as a single report. The required track modeling approach must therefore support a track comparison metric that can handle such varied track histories.

A review of the literature failed to identify an approach that can meet these three objectives. References such as [6] and [7] provide interesting insight into the problem but do not fully meet the stated requirements. Therefore, based on these three general objectives, we are proposing that:

1) tracks are modeled by a collection of parameterized line segments to match waypoint behaviour;
2) the modeling process can be performed in batch mode, without out of sequence reporting.

This approach provides track models that are better suited to a track-template fusion approach.

III. Model

From a mathematical point of view, a track is a bi-dimensional (longitude, latitude) time series \(T(0, n) : t_j \rightarrow (x_j, y_j), j \in [0, n-1]\) where \(n\) is the number of contacts (track reports), \(t_j\) is the time value and \((x_j, y_j)\) is the spatial position (longitude, latitude) for the \(j^{th}\) contact. Therefore, in the maritime surveillance context, a track should be modeled as a piece-wise linear (or segmented) time series \(\hat{T}\):

\[
T(0, n) \cong \hat{T}(0, n) := \bigcup_{i=1}^{k} T(a_i, b_i),
\]

where \(k \in [0, n-1]\) is the number of segments and \(0 = a_1 < b_1 = a_2 < \ldots < b_k-1 = a_k < b_k = n\).

Straight lines \(\hat{T}(a_i, b_i)\) (or \(\hat{T}_i\) to lighten the notation) can be estimated from data by interpolation or regression (see Figure 1).

a) Linear interpolation: The \(i^{th}\) segment \(\hat{T}_i\) approximating the subsequence \(T(a_i, b_i)\) is the line connecting contacts from \(t_{a_i}\) to \(t_{b_i}\).

b) Linear regression: The segment approximating the subsequence \(T(a_i, b_i)\) is taken to be the best fitting line in the least squares sense.

Linear regression is the best option since it provides a better utilization of information coming from the Area of Uncertainty (AOU) associated with each contact report (see section VI-A for more insights on AOU).

Classical approaches to model segmented time series are resumed in Keogh et al [8]. Most time series segmentation algorithms can be grouped into one of the following three categories:

1) Sliding Windows: A segment is grown until it exceeds some error bound. The process repeats with the next data point not included in the newly approximated segment.
2) Top-Down: The time series is recursively partitioned until some stopping criteria is met.
3) Bottom-Up: Starting from the finest possible approximation, segments are merged until some stopping criteria is met.

For large amount of data, which is our situation, the 2 last algorithms are not affordable. In both cases, almost all possible solutions have to be evaluated and there is no simple way to parallelize the search. Also, for all 3 algorithms, the solution highly depends on the stopping criteria or the error bound. In facts, for a large tolerance on minimal errors, the algorithm will converge to a solution quickly but the solution may be a local extremum. On the other hand, for a low tolerance on error, the algorithm will probably converge to the global extremum, but the execution time may be unaffordable and there is an increased danger of over fitting. In our context (refer to section II), tuning the stopping criteria value is something that we would like to avoid since it is too data dependant.

IV. SEGMENTATION AS AN OPTIMIZATION PROBLEM

To fit contact positions of a track, there exists many combinations of segments configurations. We can’t afford to try them all and compare them two by two with classical statistical tests, such as \(t\) and \(F\) tests. We need an efficient way to produce hypotheses (segments configurations) and compare them in order to converge to the best one, in terms
of fitting.

To model a track is thus equivalent to resolving the following optimization problem: Find the number and the position of breaking points defining linear segments such that the fitting error on contacts position is minimized.

V. OPTIMIZATION WITH CONSTRAINTS

The optimization algorithm to model tracks should be chosen carefully in order to respect constraints imposed by the maritime surveillance context. In order to make the right choice, the following optimization constraints should be taken into account:

1) Large amount of data:
   Even though maritime surveillance is an online task, the algorithm has to model a large number of tracks that may contain a lot of reports.

2) There is no explicit function describing a contact’s position in a track:
   In order to apply classical optimization methods (using a gradient function), we need an explicit function to optimize. Such explicit function \( f(x) \), where \( x \) is a vector representing the breaking points position, is impossible to define in this context. Indeed, for every track, \( f(x) \) depends on the linear regression process, which depends on the data. Consequently, track modeling with optimization methods requiring an explicit function to optimize is unfeasible.

3) Training is not possible:
   The majority of machine learning algorithms need a training (or learning) phase. In order to be able to give results for general data input, a machine learning method will need to learn on a finite data set with known output, called a training set. There are two major consequences that may prevent acceptable learning machine performance on a wide continuous maritime watch:
   
   i) There are important differences in traffic patterns, types of vessels and surveillance processes depending on the surveillance region. For instance, we can summarize impact on track modeling of these differences with:
      
      • Region 1: The main vessel traffic consists of fishing boats and commercial ships transiting in sea lanes using waypoints. The area to cover is vast and includes many different activities combined with differing time latencies between sensor feeds.
      • Region 2: Main vessel traffic is commercial ships following sea lanes using waypoints. Ships are funneled into straits making traffic density greater at choke points, but existence of choke points makes ship detection easier.
   

From constraints 1), 2), 3) and section II, we can conclude that the optimization approach to model tracks should:

1) not require any explicit modeling function to optimize
2) be efficient without training
3) not require tolerance threshold
4) allow backward and forward estimations
5) process data in batch.

VI. NOVEL APPROACH WITH GENETIC ALGORITHM

The approach chosen is to find the optimal position for breaking points, defining linear segments, with a hybrid algorithm based on both a genetic algorithm (GA) and a simulated annealing algorithm. This algorithm is called SAGACIA [9].

While this approach has been used for pattern recognition and many other applications, to our knowledge, it is the first attempt to model piece-wise linear time series with a genetic algorithm.

The power of GAs (note: all comments for GAs are applicable to SAGACIA) comes from the fact that the technique is robust and can deal successfully with a wide range of difficult problems [10]. GAs are not guaranteed to find the global optimum solution to a problem, but they are generally good at finding acceptably good solutions to problems acceptably quickly. The basic mechanism of a GA is so robust that, within fairly wide margins, parameter settings are not critical. The small dependency of parameter settings on the GA performance is a major advantage in the maritime surveillance context described in this paper.

Moreover, with GAs there is no need to define a tolerance bound on the error. It is well known that the quality of the solution of an optimization method depends highly on the tolerance threshold. With GAs, the cost function has the discriminate role by choosing the best hypothesis regarding all existing hypotheses, instead of an absolute independent parameter. This auto-discrediting property is clearly an asset for maritime watch on wide area.

The different states of the track are represented with chromosomes. Chromosomes are made up from contacts
ordered in time. Each gene represents a track’s contact and has 2 possible values, 0 or 1. The value 1, often called a node, indicates the beginning of a segment (breaking point). An example the chromosome representation of track is illustrated at Figure 2.

![Figure 2. Chromosome representation of a track](image)

The fitness or cost associated with a chromosome $x$ is easy to model in the GAs framework. It is defined as

$$
\text{cost}(x) = f(\text{error}(x)) \cdot \text{penalty}(x).
$$

In our case, $x$ is a Boolean vector representing the position of the breaking points of the piece-wise linear model.

The error associated to the model in the cost function is the $\chi^2$ of the segment $i$

$$
\text{error}(seg_i) = \sum_{j=1}^{n_i} \left\| \frac{(x_j, y_j) - (\bar{x}, \bar{y})(t_j)}{\sigma_{xj}^2, \sigma_{yj}^2} \right\|^2,
$$

where $n_i \leq n$ is the number of contacts for the $i^{th}$ segment and $(\bar{x}, \bar{y})(t)$ is the reported position $(x, y)$ at time $t$ estimated by the line model $\hat{T}_i$. For each segment, $(\bar{x}, \bar{y})$ are defined by the parameters for the slope and intercept in both $x$ and $y$ directions. Details can be found in [1].

The function $f$ of cost function (4) is the reduced-$\chi^2$:

$$
f(\text{error}(seg_i)) = f(\chi_i^2) = \frac{\chi_i^2}{2(n_i - 2)}. \tag{7}
$$

The reduced-$\chi^2$, or the $\chi^2$ per degree of freedom, estimates $s^2/\sigma^2$, where $s^2$ is an estimated variance of the fit, which is also an estimate of the variance of the parent distribution $\sigma^2$. If the fitting function accurately predicts the means of the parent distribution, then the estimated variance, $s^2$, should agree well with the variance of the parent distribution, $\sigma^2$, and their ratio should be close to one.

The evaluation of the reduced-$\chi^2$ requires that each segment is made of at least 3 reports.

### B. Penalty Estimation

To perform time extrapolation (backward and forward) consistently with the overall behavior of the track and to handle the jittering that may occur at the beginning of the track, models with long segments should be favoured. Therefore, the penalty function satisfies

$$
\text{penalty}(seg_i) = \ell(\hat{T}(0, n)) / \ell(T_i). \tag{8}
$$

Taking the length $\ell(\cdot)$ of a segment (or a collection of segments) as the number of reports necessary to build it, we have $\text{penalty}(seg_i) = \sqrt{n_i/n}$.

Consequently, from (4), (7) and (8), the cost function to minimize for fitting track reports is

$$
\text{cost}(\hat{T}(0, n)) = \sum_{i=1}^{k} \frac{\chi_i^2 \sqrt{n_i/n_i}}{2(n_i - 2)}, \tag{9}
$$

for a track model of $n$ reports and $k$ segments with $n_i \geq 3$ reports each.

### C. Large Time Gap

When $t_{j+1} - t_j \geq \delta$, where $\delta > 0$ has been fixed at 2 hours, reports are connected together with a straight line. If a track updates only every 2 hours or less frequently, linear regression is not suited to model this track. In order to include...
this information into the genetic algorithm model, every gene corresponding to a report that has been updated after more than 2 hours are forced to be a node.

D. SAGACIA

One can see SAGACIA [9] as a GA without a crossover operator, with annealing introduced at the best-ranking individuals selection stage.

A generation of chromosomes is compared with a population of neighbors. The neighbors population is the product of a genetic mutation from the current chromosome population. A random value between 0 and 1 is associated to every genes in each chromosome, and if the mutation rate $M_R < 1$ exceeds this random value, then a mutation disturbance is applied to the gene.

When comparing two individuals (a chromosome from the current and the neighbors population), the one with the lowest cost wins unless the associated probability of acceptance $P_A$ is high enough (e.g. $\geq 0.75$). The $P_A$ of acceptance (or selection) of a chromosome current over an other one (neighbor), is defined by:

$$P_A = \begin{cases} 1, & \text{cost(current) \leq cost(neighbor)}; \\ \exp(-|\Delta \text{cost}|), & \text{else}. \end{cases}$$

The temperature $T$ is a function that decreases with the number of iterations, $L$, performed by the algorithm. The annealing is introduced through $P_A$: when the temperature $T$ is high enough and the costs are sufficiently close, the chromosome can switch to an higher energy configuration (higher cost topology).

We chose $T = T_0(0.8)^L$, where the initial temperature $T_0$ is a high value. Having $P_A \geq 0.75$ is thus equivalent to having $|\Delta \text{cost}| \leq -\ln(0.75)T_0(0.8)^L$.

The algorithm has converged when the state with the lower cost is the same for 10 generations in a row. This state defines the optimal position for breaking points of the piece-wise linear model $T(0,n)$.

As in classical GA methods, the initial population size $P_0$ and the mutation rate $M_R$ have an influence on the performance of SAGACIA. Moreover, the initial temperature $T_0$ may also influences SAGACIA’s behavior.

VII. RESULTS

This section concerns the choice of optimal parameters $(P_0, M_R, T_0)$ of SAGACIA for maritime surveillance watch. It is important to keep in mind that those parameters are not related to a vessels behavior, but strictly concern the algorithm. The optimization problem to solve was conceived to abstract practical concerns such as ship patterns and tolerance thresholds. Therefore, the optimization of parameters $(P_0, M_R, T_0)$ are purely related to the algorithm performance for a breaking point position optimization problem.

A. Data Types

We listed 3 types of tracks from data: fishing, straight and manoeuvring. Fishing tracks refer to fishing boat tracks. As we mentioned in section II, fishing boats manoeuver on short segments which may not be long enough to optimize the GA parameters.

Straight tracks refer to the kind of track where all variances related to the measurement error ($\sigma_x, \sigma_y$) are more important than the variance between all reports positions (variance of the fit $s^2$). In other words, modeling a straight track would result in a low ($< 1$) reduced-$\chi^2$. With important uncertainty measure associated with each report’s position, the optimal track model, or the global minimum for the cost function, is easy to reach. This paper presents results for one straight track, HF22582, of 100 reports (see figure 3).

From an optimization point of view, manoeuvring tracks are the most interesting ones. They represent the opposite situation created by straight tracks: variances related to the measurement error are less important than the variance between all reports positions. The novel approach was tested with 2 manoeuvring tracks: DOMINION with 71 reports and TORONTO with 57 reports (see figures 4,5).

Figure 3. HFS22582 a straight track of 100 reports.

B. Experimentation

The rest of this paper concerns the evaluation of SAGACIA’s performance in an off-line context, with straight and manoeuvring tracks. Performance will be quantified with two quantities: convergence rate and CPU time needed to converge.

For each track, for a given set of parameters $(P_0, M_R, T_0)$, 100 runs of SAGACIA were executed. The initial population size $P_0$ varies from 10 to 300, and the mutation rate values tested are $M_R = 0.05, 0.10, ..., 0.60$. A run is successful if the algorithm converges to the global minimum. The convergence rate is defined by the number of successes over the 100 runs.
The goal of these experiments is to define the optimal set of parameters \((P_0, M_R, T_0)\), in an off-line and real time context, where short execution time is crucial.

### C. Initial Population Size

There is a lot of literature (e.g. [11], [12]) documenting the choice of the optimal population size for genetic algorithms. Usually, it is recommended to have at least as many individuals in the population as the number of genes. Since the GA (and SAGACIA) searches through chromosome topographies to find the best solution, it has to be able to produce those topographies. Therefore, a large initial population size \((P_0 > n)\) is more likely to help the GA to find the global optimum in fewer generations. But on the other hand, a large population size slows the execution time of the algorithm. In our case, we would like to minimize \(P_0\) to have good performance within a reasonable execution time.

To increase the convergence rate, we introduced in the initial population the Sliding Windows (S-W) solution (see section III). To produce the S-W solution, a segment is grown until it exceeds some error bound. The error bound is an increase of 20\% of the reduced-\(\chi^2\).

Figures 6 and 7 illustrate SAGACIA performance for track DOMINION (maneuvering), without the S-W solution in the initial population. As anticipated, performance and execution time increase with \(P_0\).

**Figure 4.** DOMINION a manoeuvring track of 71 reports.

**Figure 5.** TORONTO a manoeuvring track of 57 reports.

**Figure 6.** Data show strong evidence that convergence rate \(C\) increases linearly with \(P_0\). F-test gives P-Value of \(1.20 \times 10^{-5}\), for the null hypothesis \(H_0: \text{slope} = 0\), confirming the presence of a linear relation. Estimated relation is \(C = 54.48 + 0.054P_0\) with \(R^2 = 0.50\). Errors associated with each convergence rate is the standard error for a binomial distribution \(B(n, 100)\).

**Figure 7.** Relation between execution time \(t\), for a single run when it reaches the global minimum, and \(P_0\) is described by \(t = 0.75 + 0.11P_0\) with \(R^2 = 0.99\). Introduction of the Sliding Windows solution into the initial population considerably improves SAGACIA’s performance. It is a general fact that for some problems, evaluating solutions that are near an existing solution may be very efficient. In that case, it may give a performance advantage to simulated annealing approaches, compared to recombination (crossover) based methods, such as classical GA. This argument favors the choice of SAGACIA over a simple GA to solve the track model optimization problem.
Observations for a mutation rate of 0.2, for the 3 tracks HFS22582, DOMINION and TORONTO, lead to the conclusion that the convergence rate is independent of the population size when the sliding windows solution is in the initial population. Figures 8 and 9 show results for DOMINION and HFS22582 respectively.

Figures 8 and 9 show results for DOMINION and HFS22582 respectively.

Figure 8. Data show strong evidence that convergence rate relative to population size is normally distributed. Under the null hypothesis $H_0 : \mu = \text{median}$, the t-test gives a P-Value of 0.18 for track DOMINION and 0.76 for track TORONTO. In both cases, the absence of evidence against the null hypothesis, and the lack of significative skewness and kurtosis problem allow us to conclude to the normality of $C(P_0)$ and therefore that $P_0$ has no influence on $C$ when the S-W is in the initial population (errors are omitted for clarity).

Figure 9. Neglecting the effect of population sizes 10 and 20, we are allowed to state that population size has no effect on the performance. Population sizes 10 and 20 are outliers.

Averaged convergence rate for track HFS22582 is always 100. This is consistent with straight tracks predictions: when variance associated to each observation (report) is greater than the variance of the linear fit creates easy fitness landscapes.

To have a better understanding of the optimization algorithm’s behaviour, 100 runs were performed for $P_0 = 20, 30, ..., 290, 300$, with mutation rate varying from 0.05 to 0.6. The temperature $T_0$ was fixed at 60. Each time, we introduced the sliding windows solution in the initial population.

The first important observation is that when the S-W solution is in the initial population, the convergence rate is independent of the population size. For each $M_R$, the normality, skewness and kurtosis were tested and all concluded to the normality of the data.

When the initial population does not contain the S-W solution, the convergence rate increases linearly with the population size. For instance, in the case of DOMINION with $M_R = 0.2$ (see Figure 6), $C = 54.48 + 0.054P_0$. Under the same conditions, SAGACIA gave convergence rates normally distributed around 71.50 when S-W is part of the initial population (see Figure 8). Therefore, to reach the same convergence rate, the algorithm without S-W would need a population size of approximately 319. For instance, compared to the $P_0 = 30$ needed with S-W, a population size of 319 represents an increase of roughly 787% in execution time (refer to Figure 7).

The second important observation for this experimentation is that for each mutation rate tested, population sizes 10 and 20 are outliers. From track HFS22582 (see Figure 9) we can see that population size 10 and 20 may be outliers. Results collected from experimentations with the track TORONTO lead to the same conclusion. Figure 10 clearly shows evidence that population sizes 10 and 20 are outliers from normal distributions. Although population sizes 10 and 20 are biased, for each mutation rate tested, convergence rates for population sizes from 10 to 300 are still normally distributed. However, according to our results, population size should not be fixed at less than 30. Therefore, according to our data analysis and for performance reasons, population size should be 30.

D. Mutation Rate

The last observation from the experimentation described in the previous section is that the optimal mutation rate is 0.2. As illustrated on figure 11, convergence rates averaged over populations from 30 to 300 (10 and 20 being biased), for each mutation rate for both tracks TORONTO and DOMINION, peak at the same value. It allows us to conclude that $M_R = 0.2$ is the optimal rate.

E. Temperature

Temperature $T_0$ (10) has to be fixed high enough to introduce annealing. Since the annealing is paired with mutation, we believe that temperature is not a crucial parameter in the success of SAGACIA. Indeed, in Li and Jiang’s paper, there is no suggestion about the initial
temperature value. Since $|\triangle \text{cost}| \leq 0.29 T_0 (0.8)^L$, the optimal value for $T_0$ is tightly related to cost values. For the same reasons machine learning is not an option to model tracks (see section V), we would like to avoid tuning $T_0$ from cost values.

Tests were performed for $T_0 = 0.01, 0.05, 0.1, 0.15, ..., 0.6$, for track TORONTO and DOMINION. From those tests, only general trends were observed. For small mutation rate values, SAGACIA does not perform well with low initial temperatures and for large mutation rates, it produces poor results for high $T_0$. Otherwise, no specific trends were detected through experimentation. The absence of significant results about the optimal initial temperature value allows us to conclude that $T_0$ does not have a major role in the performance of SAGACIA (as long as it does not take extreme values).

The context of wide area maritime surveillance makes the genetic algorithm a good choice to solve this optimization problem. The results with a hybrid approach, combining genetic algorithm and simulated annealing are presented. This method does not require any explicit function to optimize, is efficient without training, and does not require tolerance threshold; and the transposition into binary chromosomes is straightforward. Moreover, due to the fitness landscape created by vessel routes, parameters such as initial population size and temperature have a negligible impact on the modeling performance, which contribute to the algorithm robustness required. Those reasons justify the use of this approach to model segmented time series and results show that it is well suited for the maritime surveillance context.

The track modeling approach presented in this paper was implemented in an operational setting and tested with live real data from a wide area over a period of days. These results illustrate the viability and the effectiveness of the proposed approach.

REFERENCES


