This document is published in:
2012 15th International Conference on Information Fusion (FUSION),
Singapore, 9-12 July 2012. IEEE, 2012, pp. 1822 - 1829.

Improving Multiple-Model Context-Aided Tracking through an Autocorrelation Approach

Enrique D. Martí, Jesús García
Group of Applied A.I.
University Carlos III of Madrid
Av. Universidad 22, 28270 Colmenarejo
Email: {emarti, jesus.garcia}@inf.uc3m.es

John L. Crassidis
Dept. of Mech. & Aerospace Eng.
State University of NY at Buffalo
Amherst, NY 14260-4400
Email: johnc@buffalo.edu

***Abstract*—This paper continues a previous work, where the context-aided tracker "ConTracker" was used to detect suspicious behaviors in maritime vehicle trajectories. ConTracker takes into account map-based contextual information –which includes water depth, shipping channels and areas/buildings with a high strategy value– to determine anomalies in ship trajectories. The different areas act as repellers or attractors that modify the expected trajectory of the tracked vessel.**

In the original scheme, a multiple-model adaptive estimator (MMAE) is used to estimate the noise parameters of the tracking system: sudden increases on the output reflect unexpected maneuvers –such as entering a forbidden area– that are translated as alarms. The work presented here shows the results obtained by implementing a generalized version of the multiple-model adaptive estimator (GMMAE). While the former approach uses information of the last cycle to update the weight/importance of each model, our proposal calculates a likelihood value based on the time-domain autocorrelation function of the last few indicators. GMMAE provides a much faster response, which ultimately leads to a general performance boost: alarms are faster and clearer. Compared with previous works, GMMAE is particularly effective returning back to normal state after an alarm has been raised: this results in alarms with a better defined duration. Results are presented over several simulated trajectories, featuring a variety of realistic anomalies which are correctly identified. They include direct comparison with the previous approach, for an objective demonstration of the achieved improvement.

I. Introduction

Monitoring maritime traffic can be a complicated task due to the potentially high number of vessels to be tracked simultaneously. Thus, it is a matter of critical importance to develop automated systems that can process such scenarios, and focus the attention of human operators into situations where their expertise is required. Many available solutions deal with this problem observing only the trajectory of the tracked vessel, either restricting to location information [1], [2] or incorporating kinematic data [3]. More advanced techniques [4] take into account the effect of terrain morphology in vehicle trajectory and offer better results, suggesting that there is room for further improvements by integrating information about relevant factors.

Our idea is based on two works which follow this line of thought: improve tracking quality by incorporating new types of information about the environment, instead of refining the use of the common data elements. The first one can be found in [5], where a convoy of vehicles is tracked using the auxiliary concept of repelling/attracting areas. The second one, as explained in [6], consists in combining qualitative and quantitative features of terrain (slope, vegetation and others) in order to produce a "trafficability" value that determines the maximum velocity of a vehicle over it.

The tool used in this work is a maritime context-aware tracker called *ConTracker* [7] (from CONtext TRACKER). ConTracker mix the aforementioned concepts, making it capable of detecting anomalous maneuvers and inferring the reason that made that particular piece of trajectory look suspicious. This piece of software tracks targets (low level fusion, L1 according to JDL model [8]), but also detect anomalous behaviors and propose hypothesis explaining the underlying cause (L2/L3, situation assessment).

Current implementation of ConTracker works with a radar that measures the position of vessels, and incorporates the context information provided by a map, in order to help determining the path that a given target is expected to follow. When a boat shows a suspicious movement pattern, it is red flagged and an alarm is raised. Additionally, in those –more frequent– cases when a vessel is following a regular path, context information can improve the tracking quality.

The definition of "unexpected" maneuver managed by ConTracker is based on the difference between a predicted motion pattern (using Kalman filters) and the received observations (radar measures). Using the model/measure errors of the Kalman filters, it is possible to assign a likelihood to the observed difference between prediction and measurement – which will be marginal for large discrepancies.

However, if we assume a sufficiently large error for the motion model, the situation will be within the bounds of normality. So, we can consider the real magnitude of error model as a dynamic value: it is small when movement follows the assumed model, but is large if the vessel operates unexpectedly.

ConTracker estimates the optimal value of this error thanks to a Multiple-Model Adaptive Estimator (MMAE) [9], [10]. ThisThis algorithm maintains a bank of filters with different configurations of process noise. The filters are evaluated on each time step to provide a combined estimation of vessel

trajectory, and also calculate the value that better describes process noise at that point. The optimal value for process noise and its change ratio identify unexpected maneuvers and, thus, behaviors susceptibe to be red-flagged.

In previous works with ConTracker, authors detected and identified some performance problems related with how MMAE is applied. This paper replaces this technique with the more sophisticated Generalized Multiple-Model Adaptive Estimator (GMMAE)

Next section explains the composition and functioning of ConTracker. Following, the limitations of MMAE are analized, after which GMMAE will be introduced. The fifth section outlines the setup for ConTracker, detailing parameters and changes over previous versions. The two last sections show the obtained results and extract some conclusions about the work.

II. ConTracker

This section briefly explains the functioning of ConTracker. For further details, please review previous work [7], [11].

The following points will review some design and construction aspects of ConTracker: which/how contextual information is used, the design of the filter model, a brief introduction to MMAE algorithm, and a glimpse of the hypothesis generator.

A. Context Information

All the context information used by ConTracker is contained in a map of the monitored zone. Figure 1 shows the map corresponding to the tested scenario. This map is divided in square cells with a side of approximately $500m$. Each cell has a "trafficability" value $0 \leq \nu \leq 1$ that is calculated using four different features:

1) water depth (low depth represented as brown shading)
2) marked shipping channels (as the one that crosses the map from cells $(12, 4)$ to $(2, 15)$)
3) Anti-Shipping Reports (ASR) (e.g. cells $(4..5, 1..2)$)
4) presence of High-Value Units (HVU) (as in $(1..2, 11)$)

Trafficability values are specific for the type of vehicle that is being tracked. For instance, tugboats are supposed to move inside marked channels, while a skiboat crossing one is considered an anomaly.

The trafficability value of a cell limits the maximum speed at which a vessel is able to move over it. Looking at figure 2, when a vessel is in cell 5, its speed will be affected by trafficability ν_5. The trafficability of the 8 neighbor cells is used to estimate the course $\hat{G}_{tg}$, also called "preferred direction". The direction previously followed by the vessel $\hat{G}^-$ is combined with the preferred direction, to produce the expected course $\hat{G}^+$ (also called "nudged direction").

The procedure for calculating these directions is fairly simple. It can be found in [11].

B. Filter Algorithm

ConTracker use Kalman filters to integrate the predicted state of the vessel with the last radar observation, and perform the tracking. The features of typical sailing trajectories makes possible to use a near-constant velocity prediction model, with a state vector containing longitude (λ), latitude (ϕ), longitudinal speed (v_λ) and latitudinal speed (v_ϕ):

$$x = \begin{bmatrix} \lambda & \phi & v_\lambda & v_\phi \end{bmatrix}^T \tag{1}$$

Where vessel speed has module $\|v\| = \sqrt{v_\lambda^2 + v_\phi^2}$, and direction marked by angle θ (with $\cos\theta = \frac{v_\lambda}{\|v\|}, \sin\theta = \frac{v_\phi}{\|v\|}$). Taking into account the maximum speed due to trafficability in current cell ν_5, we can define the following discrete time motion model:

$$x_{k+1} = \begin{bmatrix} \lambda + v_\lambda \Delta t \\ \phi + v_\phi \Delta t \\ \nu_5 \|v\| \cos\theta \\ \nu_5 \|v\| \sin\theta \end{bmatrix} + w_k \tag{2}$$

Where $w_k \sim \mathcal{N}(0; Q_w)$ represents the zero-mean Gaussian process noise that accounts for changes in velocity. The covariance matrix Q_w is assumed to be diagonal.

Previous model can be reduced to the following prediction matrix F, for being used in a Kalman filter:

$$F = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{3}$$

Radar measure model can be described by:

$$y_k = H x_k + v_k \tag{4}$$

Where $v_k \sim \mathcal{N}(0; Q_v)$ represents the measurement error and H is the measure matrix:

$$H = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \tag{5}$$

ConTracker modifies previous model to incorporate the contextual information: the nudged direction $\hat{G}^+$ is used to recalculate θ. Also, trafficability of current cell is used to modify the expected speed of the vessel. The resulting model is similar to a Kalman Filter, but adapting the propagation step to take into account these factors. For further details, see [11].

C. Multiple Model Adaptive Estimator

Using previous model, a vessel approaching a non-navigable zone will be "repelled" by the preferred direction vector with increasing strength. If such a behavior continues, the discrepancy between prediction and observation will be translated as an increment in the process noise. Multiple Model Adaptive Estimation (MMAE) is a technique that estimates unknown model parameters by evaluating several filters in parallel, each one keeping a particular configuration of the parameters. In our case, these parameters are the components $q = \begin{bmatrix} q_1 & q_2 \end{bmatrix}^T$ of the process noise variance –the main diagonal of Q_w.

At each time step, MMAE tries to estimate how probable is for each particular configuration $q^{(\ell)}$ to describe the real values of the unknown parameters, by considering the received observations $Y_k = \{y_1, y_2, \ldots, y_k\}$. This is, $p(q^{(\ell)}|Y_k)$. Put in simple terms, it is possible to apply Bayes' rule to the

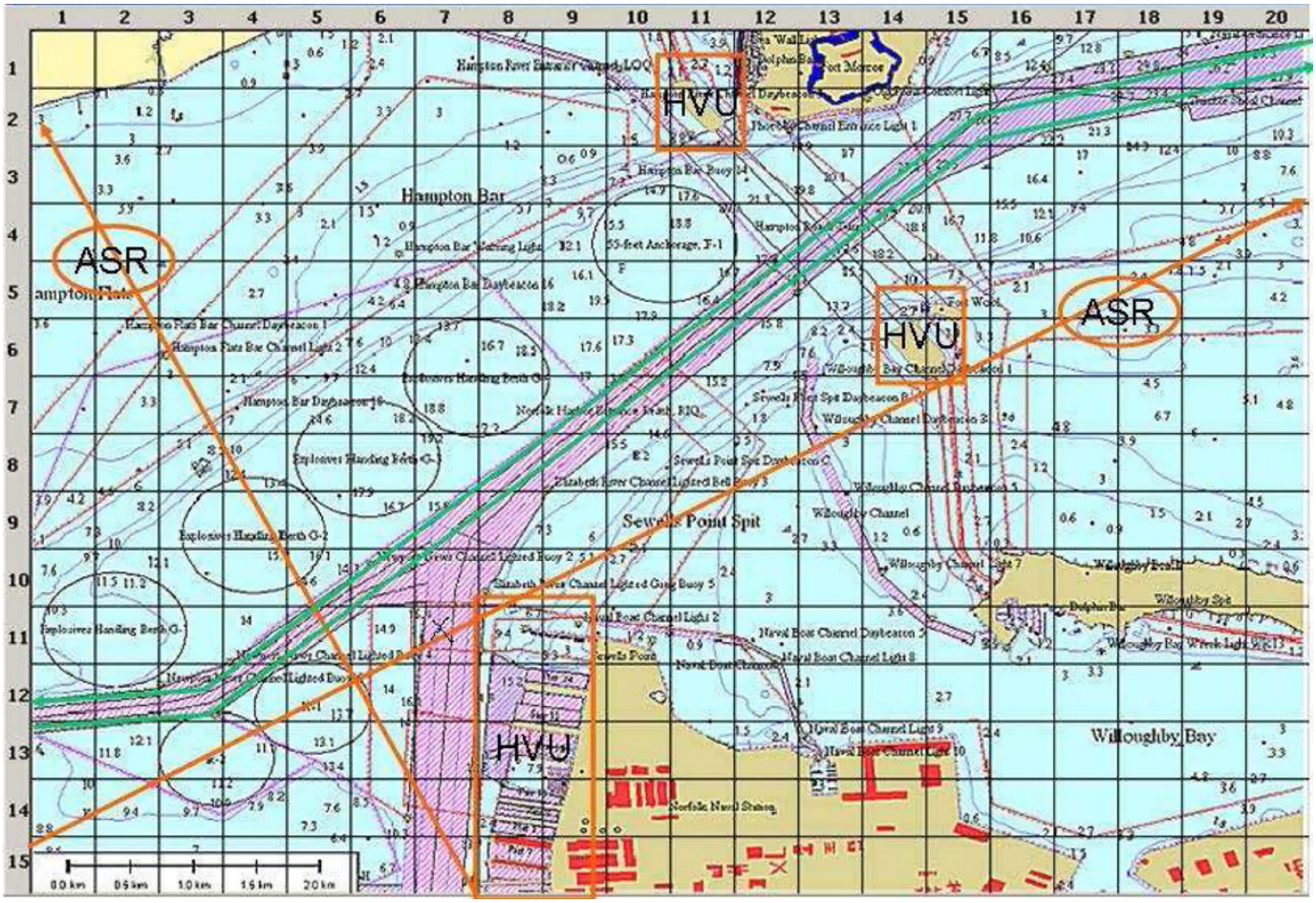

Fig. 1: Maritime scenario created from database. Shows the partition in cells, and contains trafficability information

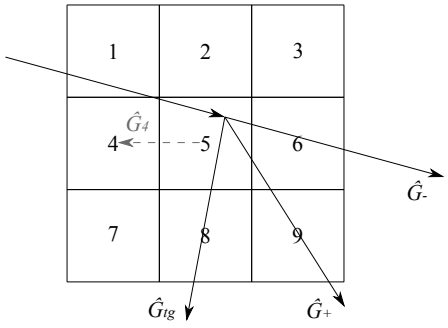

Fig. 2: Context of a cell –central, number 5– is composed by its 8-neighborhood. Nudged velocity direction $\hat{G}^+$ is determined by previous course $\hat{G}^-$ and preferred velocity direction $\hat{G}_{tg}$

probability distribution $p(q|Y_k)$. Further simplifications will yield a recursive formula that allows to define a set of weights $w^{(\ell)}$ for the different configurations of process noise $q^{(\ell)}$, so that it describes the probability distribution for the real value for $p(q|Y_k)$. The so called "adaptive law" of MMAE updates the weight of a filter using the likelihood of the last received measure with respect to the prior estimate:

$$w_k^{(\ell)} = w_{k-1}^{(\ell)} \cdot p(y_k|x_k^-) \tag{6}$$

And normalizing the weights after that to obtain $\bar{w}^{(\ell)} = \frac{w^{(\ell)}}{\sum_j w^{(j)}}$. Finally, ConTracker calculates its estimate for q as the weighted average of MMAE filters:

$$\bar{q} = \sum_j \bar{w}^{(j)} \cdot q^{(j)} \tag{7}$$

D. Hypothesis Generator

Once filtering step is over, ConTracker has to decide if the behavior of the vessel is anomalous. Instead of taking this decision based on the magnitude of the estimated noise $\|\hat{q}\|$, past versions of ConTracker opted to use the difference between consecutive time steps $\Delta\hat{q} = \|\hat{q}[k]\| - \|\hat{q}[k-1]\|$. The reasons for preferring this delta over the raw value is that MMAE reacts very slowly after an alarm has been raised, and keeps a "high noise" profile that makes difficult to discern the end of the alarm and new events (more details on this subject can be found in section III, and in figure 3). The proposal described in this paper solves the problem and makes possible to use the noise value $\|\hat{q}\|$.

III. Limitations of Multiple Model Adaptive Estimator

The original MMAE-based proposal suffered several problems that made it less robust and could spoil its performance. The first and maybe the most important issue is related with how the MMAE weights its internal filters (see II-C). At first, weights are initializaed as a uniform distribution. After that, they are recursively modified using the likelihood of each filter estimate with respect to the last measure, which at time step k is calculated for each filter (l) as:

$$\omega_k^{(\ell)} = \omega_{k-1}^{(\ell)} \cdot p(y_k|\hat{x}_k^{(\ell)}) \tag{8}$$

$$\omega_k^{(\ell)} = \frac{\omega_k^{(\ell)}}{\sum_{j=1}^{M} \omega_k^{(j)}} \tag{9}$$

This approach has the drawback of keeping memory of every past likelihood. The weight for each filter can be written as:

$$\omega_k^{(\ell)} = \frac{1}{M} \prod_{t=1}^{k} p(y_t|\hat{x}_t^{-(\ell)}) \tag{10}$$

On the event of a sudden change in procss noise, many cycles can be required to compensate the difference between accumulated weights. Since ConTracker is trying to detect changes in the value of the optimal noise, it would make much more sense to priorize last likelihoods.

Despite this problem, we can see in previous works that ConTracker using MMAE reacts very fast to vessels starting to show an anomalous behavior –although it takes many cycles to cool down to normal noise levels after the target has returned to a normal motion pattern.

Apart from the aforementioned reason, this uneven performance has an additional cause: the optimal noise covariance is calculated as the weighted mean of the filters' values. In this domain, relevant changes in noise levels are marked by their difference in orders of magnitude. Suppose a toy example were we have 2 particles representing different configurations of noise covariance $\sigma = (\sigma_X; \sigma_Y)$ as following: $p^{(1)} = (10^{-10}; 10^{-10}); p^{(2)} = (10^{-20}; 10^{-20})$ with non-normalized weights $\tilde{w}^{(1)} = 10^{-4}; \tilde{w}^{(2)} = 1$. A desirable noise estimate should be much closer to be small –that of the second filter– than large. However, MMAE will return the weighted average, which turns out to be $\sum_{l=1}^{N} p^{(\ell)} \cdot w^{(\ell)} \approx 10^{-14}$. This is a million times larger than the most probable value.

This secondary effect is not a bad thing *per se*: in fact, the dragging effect of "large noise" particles improves the reaction time for alarms. However, right after the red-flagged vessel returns to normal state, the particles representing large noises will enormously bias the estimation. Figure 3 shows the problem in the left plots, and the results obtained with the presented technique in the right. Note that Y axis has logarithmic scale (the real difference is huge). The advantages of the presented algorithm include:

1) The solution converges to the residual noise of prediction model within 200 seconds of simulation, while MMAE is far from it after 1200 seconds of trajectory,
2) periods of anomalous and normal behavior are clearly differentiated –right plots make easy to identify two alarms, one at $t = 400s$ and $t = 850s$.

The solution to this problem has two parts: first, weights will be calculated using only the last few likelihoods so that events far in the past will not affect the prediction. Second, the likelihoods for adapting the weights will be calculated using GMMAE –much faster than MMAE–. Next subsection presents GMMAE algorithm.

IV. Generalized Multiple Model Adaptive Estimator

Due to space constraints, this section briefly presents GMMAE algorithm without providing a real mathematical derivation nor any proof. Reader is encouraged to review [12], [13], since they introduce and explain the same formulation used in this work.

The final goal of GMMAE is to improve the likelihood function used for updating the weights in MMAE. This likelihood, when MMAE is composed by Kalman filters, reduces to:

$$L_{MMAE} = p(y_k|x_k^-) = \frac{1}{\sqrt{2\pi \det(C_0)}} \cdot \exp\left(-\frac{1}{2} e_k^T C_0^{-1} e_k\right) \tag{11}$$

Where C_0 is the covariance matrix defined as $H_k P_k^- H_k^T + R_k$ and $e_k = y_k - H_k \hat{x}_k^-$ –as defined in last subsection.

The problem with this formulation is that it only uses the last observation. GMMAE includes not only residual e_k, but a number $i+1$ of them: $\epsilon_i = \begin{bmatrix} e_k^T & e_{k-1}^T & \dots & e_{k-i}^T \end{bmatrix}$. Accordingly, it substitutes the covariance matrix C_0 with the autocorrelation matrix $\mathcal{C}_i$, so that:

$$L_{GMMAE} = \frac{1}{\sqrt{2\pi \det(\mathcal{C}_i)}} \cdot \exp\left(-\frac{1}{2} \epsilon_i^T \mathcal{C}_i^{-1} \epsilon_i\right) \tag{12}$$

The autocorrelation matrix can be defined by blocks as:

$$\mathcal{C}_i = \begin{bmatrix} C_{k,0} & C_{k,1} & C_{k,2} & \dots & C_{k,i} \\ C_{k,1}^T & C_{k-1,1} & C_{k-1,2} & \dots & C_{k-1,i-1} \\ C_{k,2}^T & C_{k-1,2}^T & C_{k-2,0} & \dots & C_{k-2,i-2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ C_{k,i}^T & C_{k-1,i-1}^T & C_{k-2,i-2}^T & \dots & C_{k-i,0} \end{bmatrix} \tag{13}$$

Being each block $C_{k,i}$ the expectation $E\{e_k e_{k-i}^T\}$. Performing these expectations leads to:

$$C_{k,i} = \begin{cases} H_k P_k^- H_k^T + R_k & i = 0 \\ H_k F_{k-1} \left(P_{k-1}^- H_{k-1}^T - K_{k-1} C_{k-1,0}\right) & i = 1 \\ H_k \left[\prod_{j=1}^{i-1} F_{k-j} \left(I - K_{k-j} H_{k-j}\right)\right] & \\ \times F_{k-i} \left(P_{k-i}^- H_{k-i}^T - K_{k-i} C_{k-i,0}\right) & i > 1 \end{cases} \tag{14}$$

An additional advantage of this formulation over other alternatives which also use several time steps, is that it exploits the correlation between measurements at different times.

This correlation matrix has to be calculated for each of the filters composing the GMMAE estimator. This means that some matrices must be replaced by their estimates (for instance, $P_k^- \rightarrow \hat{P}_k^{-(\ell)}$ for the l-th filter). There is a problem

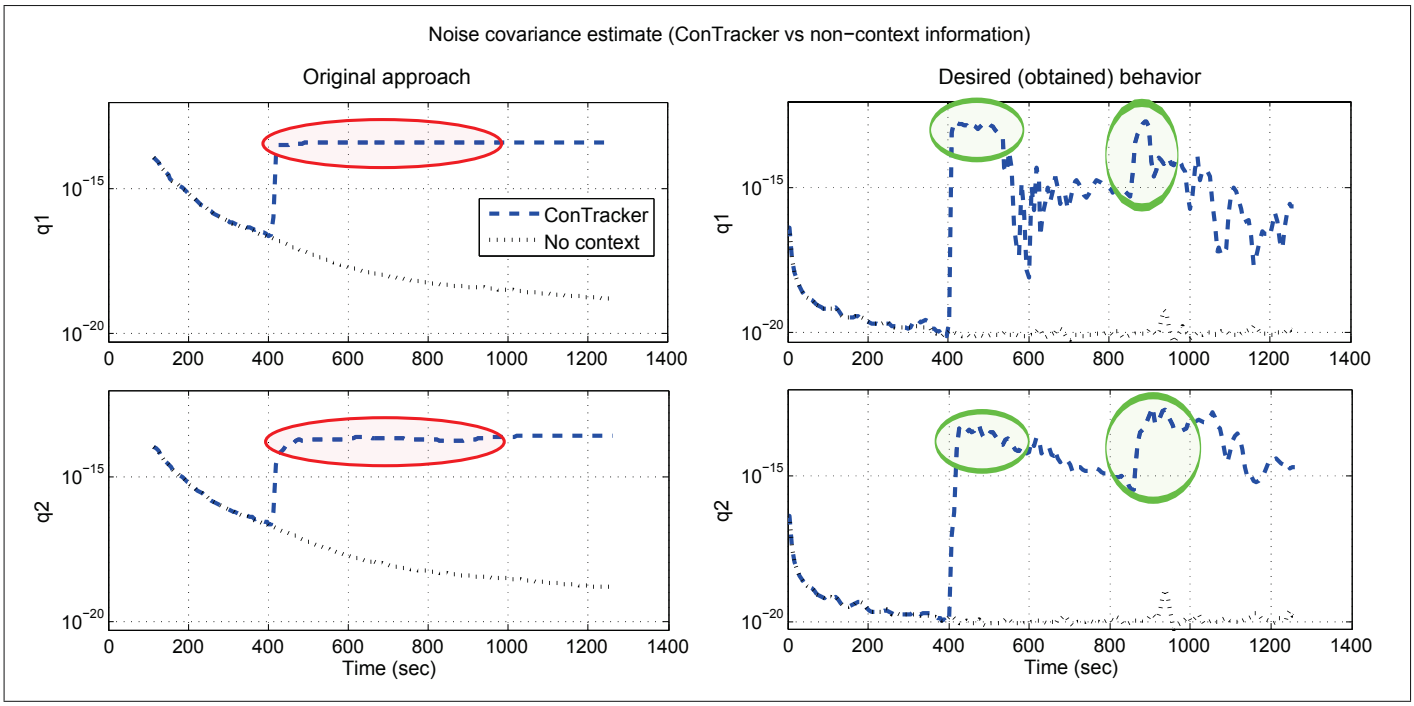

Fig. 3: Noise estimate in skiboat 2 trajectory using MMAE (left) and GMMAE proposal (right). The faster convergence avoids false positives and shows clearly where the second alarm must be raised, around t=850

with above definition: the optimal Kalman gain H_k is not know, but if we write instead the estimation obtained in the update state of the Kalman filter, the correlation terms $C_{k,i}$ with $i > 0$ will become zero. Reference [13] recomends an alternative which solves the problem.

As a final remark, when the number of correlation steps is 1, then $\mathcal{C}_k, 0 = C_0$, this is, MMAE likelihood. Because of this, when section VI show results for a GMMAE featuring 1 correlation step, it is equivalent to using MMAE.

V. Setup

This section presents important details about how ConTracker has been adjusted for our experiments. Next subsections describe how the filters are distributed over the process noise variance space, and the following one describes the modification made to the weighting of the internal Kalman filters.

A. Distribution of noise covariance parameters

MMAE estimate the optimal value for some configuration parameters by creating a set of individual filters, each one with a fixed value for those parameters. Every instance can be seen as a point in the multidimensional space of configuration parameters.

How these these points are distributed is a question of major importance. In order to fill the search space adecquately, previous works used a Hammersley sequence that generates a quasi-random uniform distribution. This method has some advantages over pure random distribution, such as maximizing the coverage and the distance between contiguous points even when just a few particles are generated.

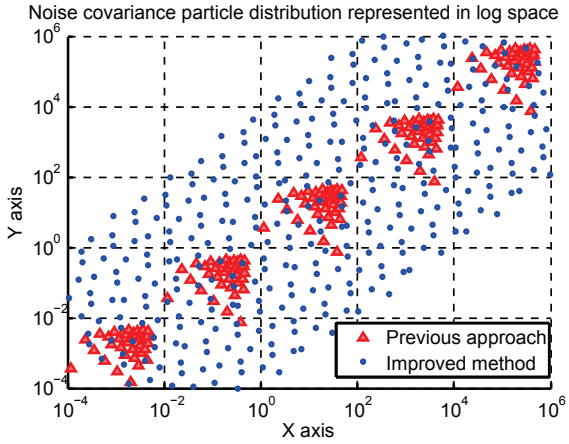

Fig. 4: Previous approach vs. current proposal. Filling performance over the search space in logarithmic scale

Hammersley method approximates a uniform distribution, but the problem requires sampling at different scales –it is impossible to fill the whole space (e.g. $[10^{-20}; 10^{-10}]^2$) using the finest granularity (10^{-20})–. This problem was tackled by generating several "patches" of samples at the required scales. For this work, a single patch is transformed so that it resembles a uniform distribution in the logarithm space. Figure 4 shows the particles used in one of the tested scenarios, generated by both methods. Both techniques offer similar performance

when they are correctly tuned (particles located in the correct part of the parameter space). Nonetheless, our proposal makes easier the process of positioning the patch right.

B. Determining the weights of Kalman filters

As mentioned in section III, the long cooldown time of previous versions of ConTracker was caused by the accumulation of evidences in weight calculation. Our current proposal includes considering only a window with the last likelihoods for weighting the Kalman filters. The canonical recursion derived from Bayes theorem, when taken back in time and removing intermediate normalization steps, is similar to:

$$w_k^{(\ell)} = \prod_{t=1}^{k} w_0^{(\ell)} \cdot p(y_t - 1|\hat{x}_t^{-(\ell)}) \tag{15}$$

Instead, we take only the last N observations, so that things happened in the past are not taken into account:

$$w_k^{(\ell)} = \prod_{t=k-N}^{k} p(y_t - 1|\hat{x}_t^{-(\ell)}) \tag{16}$$

A value $3 \leq N \leq 6$ offers a good tradeoff between a fast response time and stable behavior (noise can be fatal for the red-flagging criterion).

VI. Results

The modified version of ConTracker has been tested using the same dataset employed in previous works: a set of six trajectories of three different types of vehicle –skiboat, sailboat, tugboat– with different behaviors. This epigraph analyzes how the different parameters of GMMAE ConTracker affects the results, and compares them with those obtained in previous works.

A. General performance

The overall performance of the tracker has been improved. We can find a great example in the skiboat 2 trajectory. It appears in the map (figure 1) as the straight line that goes from cell $(15, 1)$ to $(4, 20)$. Previous works achieved to raise two alarms: one at t=400s, when the skiboat crosses the marked channel at $(11, 7)$ and the other at t=850s near the High Value Unit in cells $(6..7, 15)$. However, the slow decay of noise estimate made the tracker ignore the moment when the skiboat crossed the Anti Shipping Report area in cell $(5, 17)$, 200 seconds later. Figures 5, 6 show the covariance and red-flags obtained with current GMMAE implementation. It detects 4 anomalous events:

- Crossing marked channel at $400s < t < 450s$
- Passing near HVU in $(10..11, 8)$, at $470s < t < 500s$
- Passing near HVU in cells $(6..7, 15)$, at $880s < t < 950s$
- Crossing the ASR around $t = 1050s$

It is important to see that the reason for the alarms is correctly inferred (2 for channels, 3 for HVUs, 4 for ASRs). A simple visual inspection of figures 5, 6 shows that alarms can be detected using raw noise magnitude $\|q\|$ –something not possible with MMAE, which had to calculate the noise change ratio $\Delta\|q\|$.

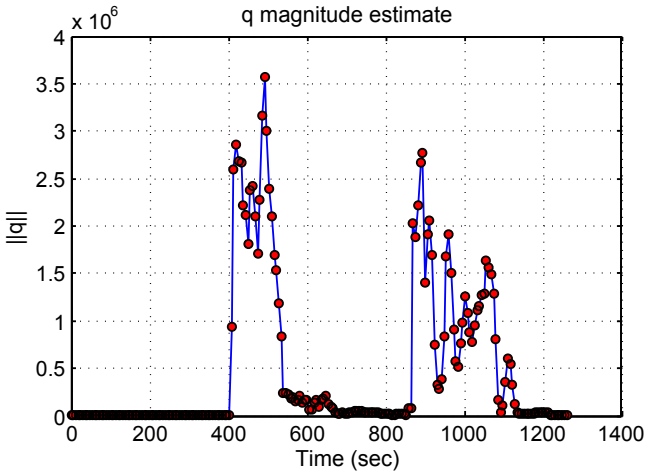

Fig. 5: ConTracker estimate for the magnitude of the process noise in skiboat 2 trajectory

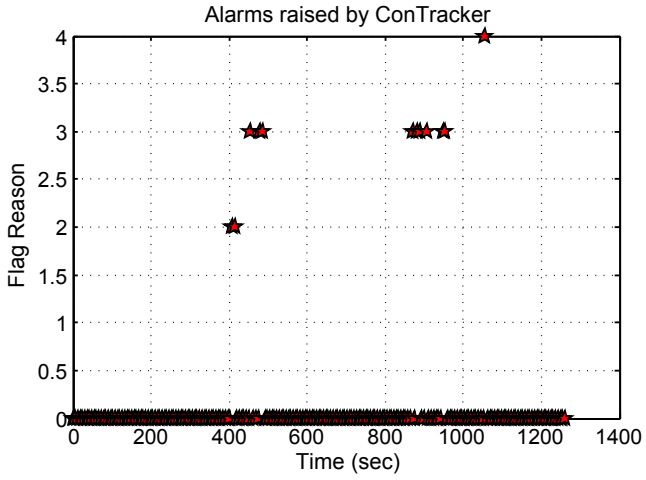

Fig. 6: ConTracker red-flag events and the inferred reasons in skiboat 2 trajectory

Previous work [11] analyzed three other trajectories. In two of them, the new proposal returns similar results –as there is no margin of improvement–. The first one is skiboat 1, where the vessel crosses two marked channels and one ASR. Three alarms are clearly raised with both techniques. The second case is a sailboat that is stuck in low waters: ConTracker red-flags it in several occasions spread along the duration of the whole trajectory.

In the remaining trajectory, a tugboat is navigating a marked channel withouth raising any alarms. Our results, however, raise an alarm when the ship briefly enters the bottom-right corner of cell $(3, 13)$. The reason is that the cell has near-zero trafficability because most of its surface is outside the channel. Strictly speaking, this alarm is not correct: it is a false positive

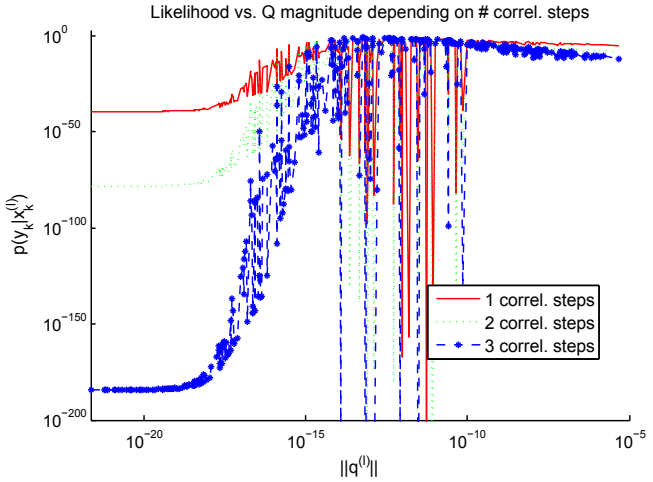

Fig. 7: Instant likelihood for filters in a GMMAE when the skiboat of trajectory #2 is red-flaged around t=400s. Filters sorted according to the magnitude of their process noise $||q||$

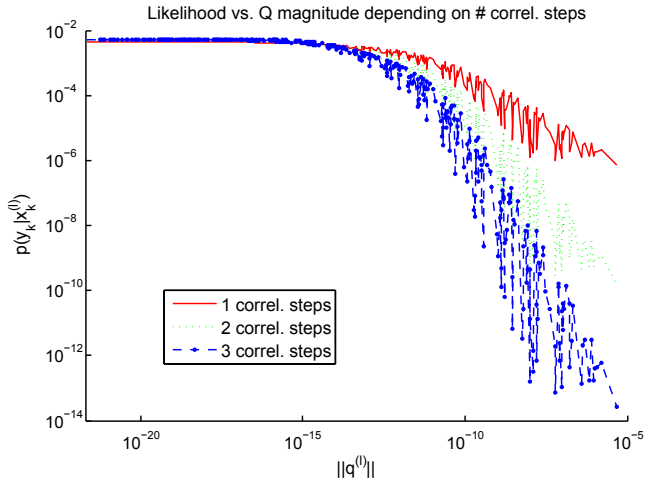

Fig. 8: Instant likelihood for filters in a GMMAE when the skiboat of trajectory #2 is navigating inside normal parameters, around t=80s.

caused by the coarse granularity of the cell partitioning –and because of this, difficult to get rid of it–. However, it shows that the proposed system has increased its sensitivity with respect to previous version.

B. Number of correlation steps in GMMAE

This section analyzes how the result is affected by the size of the autocorrelation matrix in GMMAE. As the number of steps is increased, the weights of GMMAE filters will differentiate better between adecquate and incorrect values of q. Figures 7 and 8 show us how the number of correlation steps used in the weight update phase affects the behavior of GMMAE –remember that with 1 correlation step, GMMAE likelihood is the same as MMAE. These figures plot the instant likelihood values $p(y_k|x_k^{(\ell)})$ used to update the weight

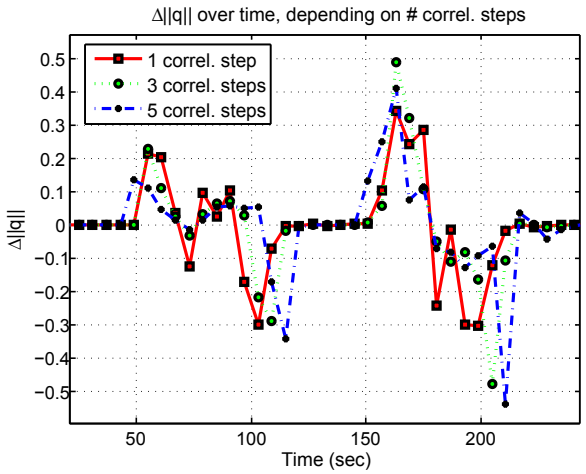

Fig. 9: Detail of the change ratio for $||q||$ in skiboat 1 trajectory. Correlating a larger number of steps leads to smoother, less aggressive estimations

of the Kalkman filters in two different moments: when a vessel starts to missbehave (figure 7), and during normal navigation (figure 8). The likelihood of a filter is tightly related with the magnitude of its process noise –apart from certain variance that gives the plot a noisy look–. However, it can be seen that increasing the number of correlation steps enlarges the likelihood gap between filters representing suitable and non-suitable process noises.

Increasing the number of correlation steps represents a clear advantage, because the magnitude of the weight of badly adapted particles is decreased in several orders of magnitude. This will improve response time, and also the estimation of real $||q||$ value. On the other hand, arbitrary increments in the number of correlation steps can be dangerous because they introduce more past measures in weight calculation. In our experiments, the scan period of the radar providing observations is around 6s. Under this conditions, a 5-step autocorrelation process employs data over a time span of 30 seconds. In figure 9 is possible to see that noise estimate is smoother, although this can cause the filter to ignore some "fast" anomalies that would, otherwise, been detected.

In general, we can say that a number between 3-5 correlation steps is adecquate in a wide range of scenarios, in spite that it is recommendable to tune it depending on the desired sensitivity of the tracker.

C. Number of particles

Increasing the number of filters that compose the GMMAE has an effect quite similar to increasing the number of correlation steps: it makes the estimate of ConTracker less agile, although it appears to result in a more robust detector –reduce the risk of false positives–. Plot in figure 10 also shows that an smaller number of particles mark the beginning of red-flag periods clearer (for instance, the two peaks around $t = 600s$). However, with more particles it is easier to infer the duration

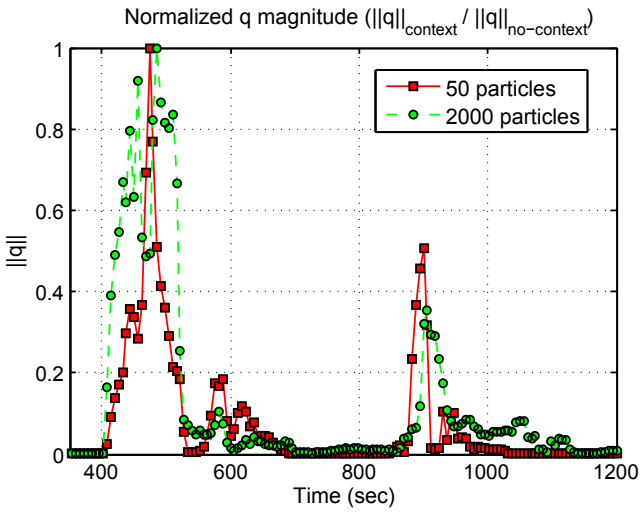

Fig. 10: Detail of the normalized magnitude of q in skiboat 2 trajectory, for two different numbers of particles

of the alarms. It is difficult to objectively estimate the best number of particles, since the optimal setting varies between scenarios. The final choice, as in the case of the number of correlation steps, will depend on the desired characteristics of the tracker and also on the available computational power.

VII. Conclusions

ConTracker is a tool which uses low level information to track vessels, and then alert of suspicious behaviors taking into account contextual infromation. GMMAE was proposed as a possible solution to the weaknesses of previous ConTracker implementation, and this work proves that theory right. The implemented version features other improvements: a new scheme for positioning the filters of GMMAE in the state space, and using a limited window of observations for calculating the weights of the filters.

The results presented here show an overall performance improvement, with faster response and an increased detection ratio of red-flag events. Two main parameters were identified: the number of components in GMMAE, and the number of steps comprised by the autocorrelation matrix, that is later used for calculating the likelihood of the filters. They affect the behavior of ConTracker in a similar way: a larger value makes the filter more robust against noise, but also less agile –risking to lose anomalous events–.

In spite that the number of particles/correlation steps have significative impact on the results, ConTracker offers reasonable accuracy over a wide range of configurations. Because os this, the optimal setup must be analized on a per case basis, taking into account other factors as the available computational power.

Finally, one of the test triggered a false alarm caused by the coarse granularity of context information: a refined trafficability grid could improve results even further.

Acknowledgment

This work was supported in part by Projects CICYT TIN2011-28620-C02-01, CICYT TEC2011-28626-C02-02, CAM CONTEXTS (S2009/TIC-1485) and DPS2008-07029-C02-02.

The authors wish to thank Dr. Jemin George for its invaluable support, and making available its software and datasets. There is also a special mention for Prof. Yang Cheng from Mississippi State University, which kindly offered a software implementation of GMMAE algorithm.

References

[1] R. Moose, H. Vanlandingham, and D. McCabe, "Modeling and Estimation for Tracking Maneuvering Targets," *IEEE Transactions on Aerospace and Electronic Systems*, vol. AES-15, no. 3, pp. 448–456, May 1979. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4102176

[2] R. Singer, "Estimating Optimal Tracking Filter Performance for Manned Maneuvering Targets," *IEEE Transactions on Aerospace and Electronic Systems*, vol. AES-6, no. 4, pp. 473–483, Jul. 1970. [Online]. Available: http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=4103555

[3] T. Kirubarajan, Y. Bar-Shalom, K. Pattipati, and I. Kadar, "Ground target tracking with variable structure IMM estimator," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 36, no. 1, pp. 26–46, 2000. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=826310

[4] P. Nougues, "We know where you are going: tracking objects in terrain," *IMA Journal of Management Mathematics*, vol. 8, no. 1, pp. 39–58, Jan. 1997. [Online]. Available: http://imaman.oxfordjournals.org/cgi/content/abstract/8/1/39

[5] D. Tenne and B. Pitman, "Tracking a convoy of ground vehicles," Cernter for Multisource Information Fusion, The State University of New York at Buffalo, Amherst, NY, Tech. Rep., 2002.

[6] A. M. Fosbury, J. L. Crassidis, and C. Springen, "Ground target tracking using terrain information," in *2007 10th International Conference on Information Fusion*. IEEE, 2007, pp. 1–8. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4408079

[7] J. George, J. Crassidis, and T. Singh, "Threat assessment using context-based tracking in a maritime environment," in *FUSION '09. 12th International Conference on Information Fusion.*, 2009, pp. 187–194. [Online]. Available: http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=5203608

[8] M. E. Liggins, J. Llinas, and D. L. Hall, *Handbook of Multisensor Data Fusion: Theory and Practice*, 2nd ed. CRC Press, 2008. [Online]. Available: http://www.amazon.com/Handbook-Multisensor-Data-Fusion-Engineering/dp/1420053086

[9] C. Chang and M. Athans, "State Estimation for Discrete Systems with Switching Parameters," *IEEE Transactions on Aerospace and Electronic Systems*, vol. AES-14, no. 3, pp. 418–425, May 1978. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4101985

[10] R. Lashlee and P. Maybeck, "Space structure control using moving bank multiple model adaptive estimation," in *Proceedings of the 27th IEEE Conference on Decision and Control*. IEEE, pp. 712–717. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=194402

[11] J. George, J. L. Crassidis, T. Singh, and A. M. Fosbury, "Anomaly Detection using Context-Aided Target Tracking," *Journal of Advances in Information Fusion*, vol. 6, no. 1, pp. 39–56, 2011.

[12] J. Crassidis and Y. Cheng, "Generalized Multiple-Model Adaptive Estimation Using an Autocorrelation Approach," in *2006 9th International Conference on Information Fusion*. IEEE, Jul. 2006, pp. 1–8. [Online]. Available: http://ieeexplore.ieee.org/xpl/freeabs_all.jsp?arnumber=4085937

[13] B. N. Alsuwaidan, J. L. Crassidis, and Y. Cheng, "Generalized Multiple-Model Adaptive Estimation using an Autocorrelation Approach," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 47, no. 3, pp. 2138–2152, Jul. 2011. [Online]. Available: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5937288