Optimal observer trajectory in bearings-only tracking for manoeuvring sources

O. Trémois
J.-P. Le Cadre

Abstract: In the bearings-only tracking context, the source state is only partially observed through nonlinear measurements which are the estimated bearings. For a manoeuvring Markovian source, the source trajectory is estimated by means of classical dynamic programming. However, the quality of the estimation is stongly dependent of the observer trajectory, thus mixing estimation and control. But, in this context, the separation principle (for estimation and control) does not hold. In fact, the problem consists in controlling a partially observable Markov decision process. Application of this framework to search theory has yet been considered in the literature. However, even if the problem presents strong similarities with an approach used in the optimisation of the search effort for a (Markovian) moving source, it is focused on the estimation of the whole source trajectory instead of its detection at the end of the scenario. To compensate this intrinsic difficulty, the observation is richer. Consequently also, the optimisation problem presents important difficulties, i.e. memory and computation requirements. Thus the authors aim to develop a feasible framework, based on the Smallwood and Sondik approach, capable of handling real problems. To attain this objective, a specific algorithm is developed and the dimension of the bearings-only tracking is drastically reduced. The applicability of the approach is demonstrated on realistic sonar scenarios.

1 Introduction

The basic problem of target motion analysis (TMA) is to estimate the trajectory of an object (e.g. position and velocity for a rectilinear motion) from noise corrupted sensor data. Classical bearings-only TMA methods are restricted to constant motion parameters (usually position and velocity). However, most of the interesting sources have manoeuvring abilities so that the performances of classical TMA may be dramatically degraded.

IEE Proceedings online no. 19990262

DOI: 10.1049/ip-rsn:19990262

Paper first received 16th February and in revised form 10th September 1998

O. Trémois was with IRISA/CNRS and is now with SACET, Ecole Louis de Broglie, Campus de Ker Lann, 35270 Bruz, France

J.-P. Le Cadre is with IRISA/CNRS, Campus de Beaulieu, 35042 Rennes cedex, France; e-mail: lecadre@irisa.fr

The study of tracking methods for manoeuvring sources is a classical theme and a huge amount of literature is devoted to this subject [1–7]. The fields of applications are as varied as radar systems, infrared systems and sonar. Various methods have been developed to deal with this problem. In his article [8], Singer modelled the acceleration by a stochastic process with a known exponential autocorrelation function. The switching methods [9–12] or more recently the interactive multiple model method and its extension, the selected filter IMM [13], are other examples of the vitality of the domain.

According to this panorama, it is obvious that the tracking of manoeuvring sources has motivated important and fruitful efforts. However, it is worth noting that these efforts are mainly on radar system applications. The context of sonar systems is rather different, since it is frequently a passive system which observations (basically the estimated bearings) depend nonlinearly on the source state. Furthermore, considering a long-time source–observer encounter (which is usual in a sonar scenario) the source manoeuvre possibilities are quite numerous and diverse. A similar remark can be made for the detection of the source manoeuvres which requires a suitable estimation of the source states, itself needing a sufficient signal to noise ratio and, overall, a sufficient time between consecutive source manoeuvres.

All these considerations advocate for the modelling of the whole source trajectory including source manoeuvre uncertainty.

With that aim, a convenient framework for TMA is the hidden Markov model (HMM) one, widely used in other contexts as speech processing, frequency line tracking [14–17], detection and tracking of dim moving sources [18, 19] or target tracking in active sonar [20, 21] systems. To apply HMM methods to the bearings-only tracking (BOT) context, a basic idea consists of a two-level discretisation of the state-space [22–24]. The probabilities of the position transitions are deduced from the probabilities of the velocity transitions. These transitions themselves correspond to the manoeuvrability constraints inherent to the source. It is thus possible to avoid too precise source manoeuvre modelling to the benefit of rather coarse stochastic modelling.

In the BOT context, the source state is only partially observed through noisy nonlinear measurements, which are the estimated bearings. Given a measurement sequence, the estimation problem consists of finding the sequence of states which maximises the conditional probability. This is achieved by means of a classical dynamic programming (DP) algorithm. This approach is an elegant solution to the manoeuvering target tracking problem, since it does not require any prior

information on the source manoeuvres. Even if the estimation of the source states sequence may basically appear as a direct application of dynamic programming principles, a major problem comes from the optimisation of the observer trajectory.

Since the problem consists mainly of controlling a partially observed Markov chain, a natural framework is the Markov decision process one, and more specifically the partially observable Markov decision process (POMDP).

The case of complete information has been previously considered [22]. In this case, the source states sequence is assumed to be known to the decision process (not to the estimation). The aim of this analysis is mainly to provide a catalogue of optimised observer trajectories and to investigate the theoretical problems occurring with the application of the DP principle. In the MDP context, the problem is to determine the control policy, such that the sequence of decisions (or control) minimises a cost functional. This problem is classically solved by dynamic programming equations. However, our problem differs from the classical one in the nature of the cost functional, which is defined on a matrix space. These matrices are Fisher information matrices (FIM), associated with the estimation of the source state sequence.

The choice of the matrix functional H is very critical. It is necessary that the matrix functional satisfies a monotonicity property, which considerably reduces the choice. Furthermore, in practical situations, the source state is not directly observable. The only available information consists of the estimated bearings, and the MDP problem then becomes far more complicated. A natural approach is that of POMDP for which exact algorithms have been developed.

Our problem presents some similarities to the search of a moving target. However, in contrast to the 'classical' search theory, we are now coping with the estimation of the whole source trajectory instead of its detection at the end of the scenario. To compensate for this intrinsic difficulty, in the TMA context the observation is richer because, wherever the target is, its detection is assumed to be certain and an estimation of its bearing is obtained. So, the function to maximise is not the probability of detection during a given time, but the information relative to source trajectory, which we can infer from the measurements and the (optimised) sequence of decisions. Even if there is no mystery in the algorithm estimating the sequence of source states, the performance of such an algorithm is mainly conditioned by the observer trajectory itself directly depending on the sequence of decisions.

2 Bearings-only TMA general framework

In the bearings-only target motion analysis problem, the states, for the observer and the source, are defined by the position and the velocity:

$$\mathbf{x}_s \triangleq [r_{xs}, r_{ys}, v_{xs}, v_{ys}]^T \quad \text{(for the source)}$$

$$\mathbf{x}_o \triangleq [r_{xo}, r_{yo}, v_{xo}, v_{yo}]^T \quad \text{(for the observer)}$$

where 'T' means transposition. Using the relative source state ($\mathbf{x} = \mathbf{x}_s - \mathbf{x}_0 = [\mathbf{r}, \mathbf{v}]$), the BOT problem is modelled by the following equations:

$$\mathbf{x}(t_k) = \mathbf{\Phi}(t_k, t_{k-1})\mathbf{x}(t_{k-1}) + \mathbf{u}(t_k) \tag{1}$$

with:

$$\mathbf{\Phi}(t_k, t_{k-1}) = \begin{pmatrix} \mathbf{Id}_2 & (t_k - t_{k-1})\mathbf{Id}_2 \\ 0 & \mathbf{Id}_2 \end{pmatrix} \tag{2}$$

and

$$\mathbf{Id}_2 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \tag{3}$$

Usually, the estimated data are the azimuths (defined relatively to the north):

$$\hat{\theta}_k = \theta_k + \nu_k, \quad \theta_k = \arctan\left(\frac{r_x(t_k)}{r_y(t_k)}\right)$$

$$\nu_k \sim \mathcal{N}(0, \sigma_\nu^2) \tag{4}$$

where the variance σ_ν^2 is given by the Woodward formula, in the case of a linear array regularly sampled in space and for a unique source in the array broadside:

$$\sigma_\nu^2 = \frac{3(1 + p\rho)(2N - 1)}{p^2\rho^2(p^2 - 1)\pi^2 d^2 N(N + 1)}$$

with:

ρ = signal to noise ratio

p = number of sensors

N = number of snapshots

d = intersensor distance

If the rectilinear and uniform motion assumption is made, the BOT problem reduces to the estimation of the initial state vector $\mathbf{x}_0$. Assume now that the source can change its velocity. Interesting sources are those which manoeuvre for tactical reasons. The decisive advantage of Markov modelling is that one does not have to define different types of manoeuvre, with the risk that the source does not follow any of them. It is just assumed that the source velocity is not radically changed between consecutive instants. More precisely, a two-level Markov chain is considered in order to model the source trajectory.

Using the elementary lemma:

$$\Pr(A, B|C) = \Pr(A|B, C)\Pr(B|C)$$

it becomes:

$$\Pr(\mathbf{x}_{t+1}|\mathbf{x}_t) = \Pr(\mathbf{r}_{t+1}|\mathbf{v}_{t+1}, \mathbf{r}_t, \mathbf{v}_t)\Pr(\mathbf{v}_{t+1}|\mathbf{r}_t, \mathbf{v}_t)$$

Furthermore, it is quite reasonable to assume that the transition over the velocities is independent of the position at time t, so that:

$$\Pr(\mathbf{x}_{t+1}|\mathbf{x}_t) = \Pr(\mathbf{r}_{t+1}|\mathbf{v}_{t+1}, \mathbf{r}_t, \mathbf{v}_t)\Pr(\mathbf{v}_{t+1}|\mathbf{v}_t) \tag{5}$$

The state space (position and velocity) is then discretised as well as the observation space (bearing measurements). The observations are defined by the Bayes formula:

$$\begin{aligned} \Pr\left(\hat{\theta}_t = \theta_j|\mathbf{x}_t = \mathbf{x}_i\right) &= \frac{\Pr\left(\mathbf{x}_t = \mathbf{x}_i|\hat{\theta}_t = \theta_j\right)\Pr\left(\hat{\theta}_t = \theta_j\right)}{\Pr(\mathbf{x}_t = \mathbf{x}_i)} \\ &= Cst.\Pr\left(\mathbf{x}_t = \mathbf{x}_i|\hat{\theta}_t = \theta_j\right) \end{aligned} \tag{6}$$

The latter equality holds in the absence of any prior information on the state and the observation.

Eqns. 5 and 6 define a hidden Markov model. As we are using the relative state vector, the observer velocity

can be seen as a control of the HMM. Different controls yield different quality of estimation of the Markov chain states. The problem is now to find the set of controls that yields the best source trajectory estimation.

3 Presentation of the POMDP framework

Consider a Markov process X_t that takes its values in a finite space (state space) $\mathcal{S}$. The stochastic process $\{X_t, t \in [0; T]\}$ is known as the central process. It is assumed to be modelled by a stationary Markov chain, which transition matrix is $\mathbf{P}$ ($\mathbf{P} = \{p_{ij}\}_{1 \le i,j \le N}$, Card($\mathcal{S}$) = N). The central process is entirely defined by $\mathbf{P}$ and the initial distribution, noted $\boldsymbol{\pi}(1)$ ($\boldsymbol{\pi}(1) \triangleq (\pi_1(1), ..., \pi_N(1))$) where $\pi_i(1) \triangleq \Pr\{X_1 = i\}$.

The central process is not directly observable, which means that X_t cannot be known exactly at time t. X_t is associated with a stochastic variable θ_t, which takes its values in a finite set of 'messages' or 'observations' $\Theta = \{1, 2, ..., m\}$. The relation between X_t and θ_t is known.

The set of all possible probability distributions on X is the simplex $\Pi(X)$ defined by $\Pi(X) = \{\boldsymbol{\pi} \in \mathbb{R}^n | \forall i \in [1, N]\ \pi_i \ge 0$ and $\Sigma_{i=1}^{N}\ \pi_i = 1\}$. Each vector $\boldsymbol{\pi}$ is a probability distribution on X (probabilistic interpretation), but it can also be geometrically represented as a point of the associated unit simplex $\Pi(X)$.

Let $\mathbf{H}_t = \{\boldsymbol{\pi}_1, d_1, \theta_1, d_2, \theta_2, ..., d_{t-1}, \theta_{t-1}\}$ ($\theta \in \Theta$: observation, $d \in \mathcal{D}$: decision, action or control, $\mathcal{D}$ finite set) be the information vector which is the initial distribution appended with the history of actions and messages received up to time t. At time t, $\mathbf{H}_t$ contains all the information that the decision maker can use to access the state of the system and to choose its decision. The global system evolution is modelled as follows:

(i) Knowing the history of the decisions (controls) and of the observations ($\mathbf{H}_t$) a decision d_t is taken.

(ii) The system transits from state X_t to state X_{t+1} according to the transition probability p_{ij}^d.

$$p_{ij}^d \triangleq \Pr\{X_{t+1} = j | X_t = i, d_t = d\}$$

(iii) An observation $\theta_t \in \Theta$ is received according to the probability:

$$r_{j\theta}^d \triangleq \Pr\{\theta_t = \theta | X_{t+1} = j, d_t = d\}$$

(iv) The information vector is updated with two new data:

$$\mathbf{H}_{t+1} \triangleq \mathbf{H}_t \bigcup \{d_t, \theta_t\}$$

(v) $w_{ij\theta}^d$ is the immediate reward cost conditionally on the following event: under the decision (or action) d the process $\{X_t\}$ transits from state i to state j and generates the observation θ.

It can be noted that, if $r_{j\theta}^d = 1$ for $j = \theta$ and 0 else, we get back to the complete information case. Uncertainities on the observation of the central process $\{X_t\}$ imply uncertainties on the decision.

In this problem the available information at time t on the internal state of the system is given by the probability distribution $\boldsymbol{\pi}_t$, which is computed only with the information vector $\mathbf{H}_t$. $\boldsymbol{\pi}_t$ is an exhaustive resume of H_t, so both of them can be named information vector:

$$\boldsymbol{\pi}_t \triangleq (\pi_1(t), \ldots, \pi_n(t))^t$$

with:

$$\pi_i(t) \triangleq \Pr\{X_t = i | H_t\}$$

The information vector $\boldsymbol{\pi}$ plays a fundamental role in POMDP problems, because it is completely observable, due to its definition. $\{\boldsymbol{\pi}_t\}_t$ is a Markovian process for a finite series of decisions. The updating formula stands as follows [25]:

$$\boldsymbol{\pi}_{t+1} \triangleq T(\boldsymbol{\pi}_t | d_t, \theta_t) = \frac{\boldsymbol{\pi}_t^T \mathbf{P}^{d_t} \mathbf{R}^{d_t}(\theta_t)}{\boldsymbol{\pi}_t^T \mathbf{P}^{d_t} \mathbf{R}^{d_t}(\theta_t) \underline{\mathbf{e}}} \quad (7)$$

In eqn. 7 the matrix $\mathbf{P}^{d_t}$ designates the transition matrix (N, N) of which elements are $\{p_{ij}^{d_t}\}_{i,j \in [1,N]}$, the matrix $\mathbf{R}^d(\theta_t)$ represents the diagonal matrix which (j, j) element is $r_{j\theta_t}^d$. Finally, $\mathbf{e}$ is the N dimension vector with all unit components.

The initial POMDP is equivalent (it gives the same cost function) to a sequential decision problem with state space $\Pi(X)$ and dynamic equation $\boldsymbol{\pi}_{t+1} = T(\boldsymbol{\pi}_t, d_t, \theta_t)$.

One can now consider the problem of the computation of the optimal decisions that can be solved by a dynamic program. In that aim, one can define $V_t(\boldsymbol{\pi})$ as the maximum expected value of the cost function that the system can accrue if the current information vector is $\boldsymbol{\pi}$ and if t iterations remain. The temporal index t is defined from the horizon and not from the origin of the scenarios. In finite horizon dynamic programming problems, temporal indexes always begin at the horizon. This is due to the algorithm itself starting its optimisation from the end.

Then, expanding over all possible next transitions and observations yields the following recursive equation:

$$V_t(\boldsymbol{\pi}) = \max_{d \in \mathcal{D}_t} \left[\sum_{i=1}^{n} \pi_i \sum_{j=1}^{n} p_{ij}^d \times \sum_{\theta} r_{j\theta}^d \left\{ w_{ij\theta}^d + V_{t-1}(T(\boldsymbol{\pi}|d, \theta)) \right\} \right] \quad (8)$$

This equation can be simplified by defining the expected immediate cost for state i if the decision d is taken during the next control interval:

$$q_i^d = \sum_{j,\theta} p_{ij}^d r_{j\theta}^d w_{ij\theta}^d \quad (9)$$

The dynamic programming equation (eqn. 8) then becomes:

$$V_t(\boldsymbol{\pi}) = \max_{d \in \mathcal{D}_t} \left\{ \sum_i \pi_i \left[q_i^d + \sum_{j,\theta} p_{ij}^d r_{j\theta}^d V_{t-1}(T(\boldsymbol{\pi}|d, \theta)) \right] \right\} \quad t \ge 1 \quad (10)$$

It only remains to define the cost associated with the fact that the process terminates on state i, which is the definition of q_i^0:

$$V_0(\boldsymbol{\pi}) = \sum_i \pi_i q_i^0 = \langle \boldsymbol{\pi}, \mathbf{q}^0 \rangle \quad (11)$$

It is possible to try to solve directly the previous dynamic programming equation, but the numerical burden grows rapidly. The following property proved by Smallwood and Sondik appears to be an instrumental answer to solving our problem:

Property 1 [25]: The function $V_t(\boldsymbol{\pi})$ is piecewise linear and convex and so can be written as follows:

$$V_t(\boldsymbol{\pi}) = \max_k \left[\sum_{i=1}^{n} \alpha_i^k(t)\pi_i \right]$$

where the $\{\boldsymbol{\alpha}^k(t)\}_k$ vectors are n-dimensional and are computed by a recursion.

The dynamic programming algorithm consists in finding, for each time index. the set of α-vectors that define exactly the cost function. Once this computation is done, if one knows the probability distribution of the system state, the optimal decision is deduced by considering the decision associated with the α-vector giving the higher scalar product with this probability distribution. The demonstration of this property (that will be omitted for the sake of brevity) gives explicitly the solution:

$$V_t(\pi) = \max_{d\in\mathcal{D}_t} \left\{ \sum_i \pi_i \left(q_i^d + \sum_{\theta,j} p_{ij}^d r_{j\theta}^d \alpha_j^{l(\boldsymbol{\pi},d,\theta)}(t-1) \right) \right\} \quad (12)$$

This formula allows one to find the α-vector associated with the information vector $\boldsymbol{\pi}$. Noting, $\Gamma(t-1) = \{\boldsymbol{\alpha}^k(t-1)\}_{k=1,\ldots,\gamma_{t-1}}$ the set of all α-vectors necessary to describe V_{t-1}, this formula permits one to find all the α-vector of $\Gamma(t)$ (with $\gamma(t)$ elements) from the elements of $\Gamma(t-1)$. If the set of decisions contains N_d elements, and the set of observations N_o elements, the recursion formula gives $N_d\gamma_{t-1}^{N_o}$ different α-vectors, which form a set that will be noted $\mathcal{G}(t)$. Only a subset of $\mathcal{G}(t)$ is enough to describe V_t and to build the set $\Gamma(t)$ ($\Gamma(t) \subseteq \mathcal{G}(t)$). Some of the α-vectors computed are not associated with any probability distribution and are thought not necessary in the description of V_t.

4 Exact algorithms for POMDP problems

One of the possibilities for obtaining all the α-vectors of $\Gamma(t)$ is to discretise the simplex $\Pi(X)$ and to compute the associated α-vector for each sample. The main problem of this approach is that it induces an important numerical burden and that it does not guarantee that all the α-vectors will be computed. Some small regions may be invisible by the grid used to discretise the simplex, regions that will possibly be associated with wrong decisions in the utilisation step. Different algorithms exist to compute the significant subset of α-vectors to build $\Gamma(t)$.

From eqn. 12 three approaches have been developed. The first is from Smallwood and Sondik, another from Monahan and the last from Cheng [27]. See [28] for more details on these algorithms.

In the following descriptions of algorithms, one can suppose that $\Gamma(t-1)$ is known as well as the associated decisions. The problem is then to find a method to compute the elements of $\Gamma(t)$ from this information.

4.1 Cheng algorithm

The idea of Cheng is to build up the set $\Gamma(t)$, adding the α-vectors one by one, instead of considering all the possible α-vectors and paring them down to the appropriate $\Gamma(t)$.

If the set $\Gamma(t-1)$ is assumed to be known, the α-vectors are computed on the vertices of the simplex $\Pi(X)$ to build the set $G_1(t)$ (i.e. the first approximation of $\Gamma(t)$). $G_1(t)$ is the set of the α-vectors associated with the vertices of the simplex. For each $\boldsymbol{\alpha}^0 \in G_1(t)$, the convex region $R_1(\boldsymbol{\alpha}^0)$, associated with $\boldsymbol{\alpha}^0$ is built:

$$R_1(\boldsymbol{\alpha}^0) = \left\{ \boldsymbol{\pi} \in \Pi(X) | \langle \boldsymbol{\pi}, \boldsymbol{\alpha}^0 \rangle \geq \langle \boldsymbol{\pi}, \boldsymbol{\alpha} \rangle, \forall \boldsymbol{\alpha} \in G_1(t) \right\} \quad \forall \boldsymbol{\alpha}^0 \in G_1(t)$$

$G_1(t)$ is used to compute $v_t^1(\boldsymbol{\pi})$ which is the first approximation of $V_t(\boldsymbol{\pi})$:

$$v_t^1(\boldsymbol{\pi}) = \max_{\boldsymbol{\alpha}\in G_1(t)} \langle \boldsymbol{\pi}, \boldsymbol{\alpha} \rangle$$

For each area $R_1(\boldsymbol{\alpha}^0)$ the cost error function is:

$$e_{G_1(t),\boldsymbol{\alpha}^0}(\boldsymbol{\pi}) = V_t(\boldsymbol{\pi}) - \langle \boldsymbol{\pi}, \boldsymbol{\alpha}^0 \rangle \quad \forall \boldsymbol{\pi} \in R_1(\boldsymbol{\alpha}^0)$$

This expression is convex in $\boldsymbol{\pi}$, so its maximum value is attained on one of the vertices of $R_1(\boldsymbol{\alpha}^0)$. If the maximum error is null for all α-vectors in $G_1(t)$, it means that $G_1(t) = \Gamma(t)$. If this is not the case, the vertex where the maximum error occurs belongs to the operative region of a new α-vector. This new α-vector ($\boldsymbol{\alpha}$) is included in $G_1(t)$ to build $G_2(t) = \{G_1(t), \boldsymbol{\alpha}\}$. The maximum error is computed again, and the process is repeated until this error is null.

Two approaches can be found for this problem. The first one consists of computing all the vertices of $R_n(t)$ for all n: this is the method of Cheng. The second one consists of computing the position only of the new vertices. This new approach, that we have explored, although also giving an exact solution, is faster since the computation is optimised.

4.2 New algorithm

The state is supposed to be of dimension N, and the cardinality of the set $G_n(t)$ is M. In the $N-1$-dimensional unit simplex (corresponding to the N-dimensional state), one can sort the $\boldsymbol{\pi}$ vectors by their number of null components (the order of the $\boldsymbol{\pi}$ vector). A $\boldsymbol{\pi}$ vector having only one non-null component is a vertex of the simplex. A $\boldsymbol{\pi}$ vector having two non-null components, is a linear combination of the vertices corresponding to the two non-null components. One can generalise this sorting process, saying that a $\boldsymbol{\pi}$ vector having K non-null components belongs to the $K-1$ dimensional space generated by all the vertices corresponding to the positive components of $\boldsymbol{\pi}$. A $\boldsymbol{\pi}$ vector having all its components strictly positive is inside the unit simplex.

The inclusion of a new α-vector is illustrated in Fig. 1. If all the vertices were computed every iteration, most of them would be searched more than once. Including a new α-vector in $G_n(t)$ does not suppress all the vertices of $R_n(t)$. It eliminates only the vertices for which the new α-vector is dominant compared to the α-vector of $G_n(t)$, and the other vertices are not modified even in their positions, but new vertices appear and attention must be concentrated on them to optimise CPU usage.

It appeared, on the one hand, for this example, that the maximum error on the cost function was attained on $\boldsymbol{\pi}_9$. On the other hand:

$$\langle \boldsymbol{\alpha}^0, \boldsymbol{\pi}_8 \rangle \geq \langle \boldsymbol{\alpha}^i, \boldsymbol{\pi}_8 \rangle \quad i = 1, \ldots, 5$$

The vertices $\boldsymbol{\pi}_8$ and $\boldsymbol{\pi}_9$ have to be suppressed, and the points (v_1, ..., v_5) have to be added as new vertices of the new partition. Here is an algorithm allowing one to update a partition after the inclusion of a new α-vector:

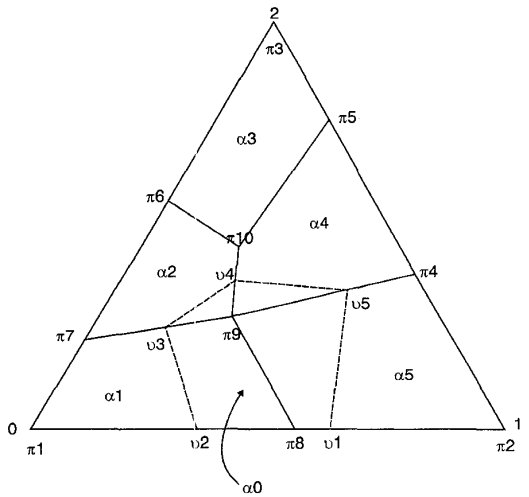

Fig. 1 *Introduction of new α-vector α^0 in set G(t)*

(i) The list of the vertices of the partition of the simplex with the operative regions of the α-vectors of $G_n(t)$ is known and denoted $VER(G_n(t))$. Each π vector is associated with the list of α-vectors for which it is a vertex. It is also associated with $v_t^n(\pi)$ and $V_t(\pi)$ for some of them.

(ii) $V_t(\pi)$ is computed on the vertices that do not have this information, and the maximum error is searched:

$$E = \max_{\pi \in VER(G(t))} V_t(\pi) - v_t(\pi)$$

If $E = 0$, then $G_n(t) = \Gamma(t)$, $v_t^n = V_t$ and the partition is completed. If E is non-null, the operative α-vector (α^0) at the location of the maximum error is included in the α-vector list:

$$G_{n+1}(t) = \{G_n(t), \alpha^0\}$$

(iii) The list of the vertices that has to be deleted is built:

$$DEL(G_n(t)) = \{\pi \in VER(G_n(t)) | \langle \alpha^0, \pi \rangle - v_t(\pi) > 0\}$$

Each π vector is associated with the information related to the subspace on which this π vector is, and to the α-vector for which it is a vertex.

(iv) For each vertex π in the $DEL(G_n(t))$ list:

- K is the dimension of the subspace;
- $K'(\geq K)$ is the number of α-vectors for which π is a vertex, the list of this α-vector is (α^{i_1}, ..., $\alpha^{i_{K'}}$) ($i_1 < ... < i_{K'}$);
- for all the combinations (j_1., ..., j_{K-1}), $1 \leq j_1 < ... < j_{K-1} \leq K'$. the system allowing the vertex (of order K) associated to (α^0, α^{j_1}, ..., $\alpha^{j_{K-1}}$) to be found, is composed. It is solved if, and only if, there exists a vertex of $VER(G_n(t))$, which is not in $DEL(G_n(t))$, that is associated to the α-vectors (α^0, α^{j_1}, ..., $\alpha^{j_{K-1}}$);
- if $K < N$, then for all the combinations (j_1, ..., j_K), $1 \leq j_1 < ... < j_K \leq K'$, the system allowing the vertex (of order $K + 1$) associated to (α^0, α^{j_1}, ..., α^{j_K}) to be found, is composed. It is solved if, and only if, there exists a vertex of $VER(G_n(t))$, which is not in $DEL(G_n(t))$, that is associated to the α-vectors (α^{j_1}, ..., α^{j_K}).

(v) Add the newly found vertices, and delete those of $DEL(G_n(t))$. Go back to Step (ii).

This algorithm may be used on the example illustrated on Fig. 1. In Step (i), the following list (containing the co-ordinates of the vertex, its cost approximation, its real cost, the indices of the non-null components, and the indices of the α-vectors for which this point is really a vertex) is assumed to be known:

$$VER(G(t)) = \left\{ \begin{array}{c} [\pi_1, v_t(\pi_1), V_t(\pi_1), (0)(1)] \\ \vdots \\ [\pi_4, v_t(\pi_4), V_t(\pi_4), (1,2), (4,5)] \\ \vdots \\ [\pi_{10}, v_t(\pi_{10}), V_t(\pi_{10}), (0,1,2), (2,3,4)] \end{array} \right\}$$

The computation of the cost function on the vertices indicates that the maximum error is obtained on π_9, and the associated α-vector (α^0) generates a cost on π_8 that is greater than $v_t(\pi_8)$. The following $DEL(G_n(t))$ list (containing the co-ordinates of the vertex, the indices of the non-null components, and the indices of the α-vectors for which this point is really a vertex) is generated:

$$DEL(G(t)) = \left\{ \begin{array}{c} [\pi_8, (0,1), (1,5)] \\ [\pi_9, (0,1,2), (1,2,4,5)] \end{array} \right\}$$

Here is the list of α-vector combinations associated with π_9, for which intersection search has to be done:

α^0, α^1, α^2 → yes

α^0, α^1, α^4 → no (there is no vertex associated to α^1 and α^4 that belongs to $VER(G_n(t))$ and not to $DEL(G_n(t))$)

α^0, α^1, α^5, → no (there is no vertex associated to α^1 and α^5 that belongs to $VER(G_n(t))$ and not to $DEL(G_n(t))$)

α^0, α^2, α^4, → yes

α^0, α^2, α^5, → no (there is no vertex associated to α^2 and α^5 that belongs to $VER(G_n(t))$ and not to $DEL(G_n(t))$)

α^0, α^4, α^5, → yes

Here is the list of α-vectors combinations associated with π_8, for which an intersection search has to be done:

α^0, α^1 → yes

α^0, α^5 → yes

α^0, α^1, α^5, → no (there is no vertex associated to α^1 and α^5 that belongs to $VER(G_n(t))$ and not to $DEL(G_n(t))$)

There are five systems to solve, which is exactly the number of vertices to add.

In the Cheng approach, it would have been necessary to compute the intersection of all pairs and triples of α-vectors. This means:

(i) 6*5*3 = 90 intersections of pairs (with the three sides of the simplex)

(ii) 6*5*4 = 120 intersections of triples.

One can compare these 210 intersections computations with the five required by the new algorithm. The latter requires more memory because it has to manage lists.

5 Optimal observer trajectory for target motion analysis

In the problem of target motion analysis (TMA), we are dealing with the estimation of the whole target

trajectory. The source movement is modelled by a Markov chain with finite state number (see Section 2), and one can consider the observer trajectory as a command, and then apply the POMDP framework to this problem.

The measurements (bearings) are discretised, and the function to maximise is related to the uncertainty on the estimation of the target trajectory (Fisher information matrix) F.

In [29] it has been proved that the only information functionals (i.e. functionals of F) suitable for dynamic programming is of the form $f(F) = g(tr(AF))$ when F is the FIM, A is a constant matrix and g is a monotonically increasing function (from $\mathbb{R}$ to $\mathbb{R}$). Using this fact, the trace of the FIM will be used as the cost function. The FIM F is the sum of instantaneous elementary FIMs, which depend on the trajectory of the target as well as on the observer trajectory.

A compound vector $(\pi, \mu, \ell) \in \Pi \times D \times C$ will be used where:

(i) $\pi_j(k) = \Pr\{$the target is in cell j at the begining of time period $k\}$;

(ii) μ = the displacement vector of the observer;

(iii) ℓ = the arrival cell of the observer.

As in the target search problem, only a subset of controls are available at time $k-1$ if the control (μ, ℓ) has been conducted at time k. This subset depends only on ℓ and is defined by:

$$B_{(\nu,k)} = B_k = \{(\mu, \ell) \in D \times C | \mu = k \rightarrow \ell\}$$

where $\mu = k \rightarrow \ell$ means that μ is the displacement vector that originates from k and arrives in ℓ. It is necessary to include the displacement vector in B_i because the cost, the FIM trace, depends on it. Using previous notations, $w_{ij\theta}^{(\mu,\ell)}$ is the imediate reward cost conditionally to the following event: the observer arrives in state ℓ and while it was moving according to the displacement vector μ, the target moved from state i to state j and generated the observation θ. This process is described in Fig. 2.

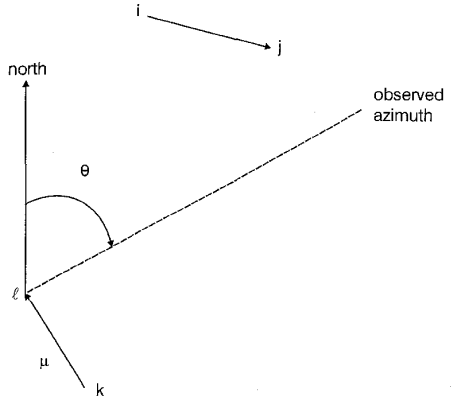

Fig. 2 *Typical scenario of observer going from state k to state ℓ according to displacement vector μ and receiving observation θ from target that went from state i to state j during the same time interval*

It can be noted that the transition matrix is independent of the movement and the position of the observer, i.e.:

$$P^{(\mu,\ell)} = P$$

The function T, which updates the vector π conditionally on the decision and the new observation, depends on the final position of the observer as:

$$T(\pi|\mu, \ell, \theta) = \frac{\pi^t P R^{(\mu,\ell)}(\theta)}{\pi^t P R^{(\mu,\ell)}(\theta)\mathbf{e}}$$

where $R^{(\mu,\ell)}(\theta)$ is a diagonal matrix whose (j, j) element is defined by:

$$r_{j,j}^{(\mu,\ell)}(\theta) = \Pr(\theta|j, \mu, \ell)$$

In other words, $r_{jj}^{(\mu,\ell)}(\theta)$ is the probability of observing θ if the target is at state j, while the observer has been displaced by a vector μ to the state ℓ. The displacement vector is not necessary since the observation occurs at the end of each step. Thus we have:

$$r_{j,j}^{(\mu,\ell)}(\theta) = r_{j,j}^{\ell}(\theta) = \Pr(\theta|j, \ell)$$

This implies that:

$$T(\pi|\mu, \ell, \theta) = T(\pi|\ell, \theta) = \frac{\pi^t P R^{\ell}(\theta)}{\pi^t P R^{\ell}(\theta)\underline{e}}$$

The cost function can now be defined recursively as:

$$V_t(\pi, \nu, k) = \max_{(\mu,\ell)\in B_k} \left[\sum_{i=1}^{N} \pi_i \sum_{j=1}^{N} p_{ij} \times \sum_{\theta} r_{j,\theta}^{\ell} \left\{ w_{ij\theta}^{(\mu,\ell)} + V_{t-1}(T(\pi|\ell, \theta), \mu, \ell) \right\} \right]$$

Using the following simplification:

$$q_i^{(\mu,\ell)} = \sum_{j,\theta} p_{ij} r_{j,\theta}^{\ell} w_{ij\theta}^{(\mu,\ell)}$$

the dynamic programming equation becomes:

$$V_t(\pi, \nu, k) = \max_{(\mu,\ell)\in B_k} \sum_i \pi_i \left[q_i^{(\mu,\ell)} + \sum_{j,\theta} p_{ij} r_{j\theta}^{\ell} V_{t-1}(T(\pi|\ell, \theta), \mu, \ell) \right]$$

As B_k depends only on k, this means that, whatever the displacement ν that was finished on state k (whatever the origin of the displacement), identical probability distributions will yield identical values of the cost function, or:

$$V_t(\pi, \nu, k) = V_t(\pi, k) = \max_{(\mu,\ell)\in B_k} \sum_i \pi_i \left[q_i^{(\mu,\ell)} + \sum_{j,\theta} P_{ij} r_{j\theta}^{\ell} V_{t-1}(T(\pi|\ell, \theta), \ell) \right]$$

with:

$$V_0(\pi, k) = 0 \quad \forall(\pi, k) \in \Pi \times C$$

That cost function can be proved to be piecewise linear and convex.

Property 2: For $t = 0, ..., T$, $V_t(\pi, k)$ is piecewise linear and convex in π. That is:

$$V_t(\pi, k) = \max_{\beta\in A(t,k)} \langle \pi, \beta \rangle \tag{13}$$

where $A(t, k)$ is a finite set of $N_{t,k}$ vectors.

Since this result is crucial for our application, a general proof is given below:

Proof of property 2:

$$t = 0 \ : \ A(0,k) = \mathbf{0} \quad V_0(\pi,k) = 0,$$

$$t = 1 \ : \ V_1(\pi,k) = \max_{(\mu,\ell)\in B_k} \sum_j q_j^{(\mu,\ell)} \pi_j$$

$$= \max_{\beta\in A(1,k)} \langle \pi, \beta \rangle$$

where:

$$A(1,k) = \left\{ q^{(\mu,\ell)} | (\mu,\ell) \in B_k \right\}$$

Assume that eqn. 13 holds at time $t-1$, then:

$$V_{t-1}(\pi,\ell) = \max_{\alpha\in A(t-1,\ell)} \sum_j \alpha_j \pi_j$$

Using the previously defined T function:

$$V_{t-1}(T(\pi|\ell,\theta),\ell) = \max_{\alpha\in A(t-1,\ell)} \sum_j \alpha_j \frac{\sum_i \pi_i p_{ij} r_{j\theta}^{\ell}}{\sum_{ij} \pi_i p_{ij} r_{j\theta}^{\ell}}$$

The recursion hypothesis is that $V_{t-1}(\pi, \ell)$ is piecewise linear and convex. This means that the space of probability distribution can be divided into regions in which the cost function can be computed using only one α-vector. Thus, we can introduce the following notation:

$$V_{t-1}(T(\pi|\ell,\theta),\ell) = \max_{\alpha\in A(t-1,\ell)} \sum_j \alpha_j \frac{\sum_i \pi_i p_{ij} r_{j\theta}^{\ell}}{\sum_{ij} \pi_i p_{ij} r_{j\theta}^{\ell}}$$

$$= \sum_j \alpha_j^{n_{(t-1,\ell)}(\pi,\ell,\theta)} \frac{\sum_i \pi_i p_{ij} r_{j\theta}^{\ell}}{\sum_{ij} \pi_i p_{ij} r_{j\theta}^{\ell}}$$

where $\alpha^{n_{(t-1,\ell)}(\pi,\ell,\theta)}$ designates the α-vector taken from $A(t-1, \ell)$, which is used to compute the cost function $V(., \ell)$ for the distribution vector $T(\pi|\ell, \theta)$.

The dynamic programming equation is:

$$V_t(\pi,k) = \max_{(\mu,\ell)\in B_k} \sum_i \pi_i \left[q_i^{(\mu,\ell)} + \sum_{j,\theta} p_{ij} r_{j\theta}^{\ell} V_{t-1}(T(\pi|\ell,\theta),\ell) \right]$$

This can be rewritten:

$$V_t(\pi,k)$$

$$= \max_{(\mu,\ell)\in B_k} \sum_i \pi_i \left[q_i^{(\mu,\ell)} + \sum_\theta \left(\sum_j p_{ij} r_{j\theta}^{\ell} \right) \times \left(\sum_j \alpha_j^{n_{(t-1,\ell)}(\pi,\ell,\theta)} \frac{\sum_i \pi_i p_{ij} r_{j\theta}^{\ell}}{\sum_{ij} \pi_i p_{ij} r_{j\theta}^{\ell}} \right) \right]$$

$$= \max_{(\mu,\ell)\in B_k} \left\{ \left(\sum_i \pi q_i^{(\mu,\ell)} \right) + \sum_\theta \left[\left(\sum_{ij} \pi_i p_{ij} r_{j\theta}^{\ell} \right) \frac{\sum_{ij} \alpha_j^{n_{(t-1,\ell)}(\pi,\ell,\theta)} \pi_i p_{ij} r_{j\theta}^{\ell}}{\sum_{ij} \pi_i p_{ij} r_{j\theta}^{\ell}} \right] \right\}$$

$$= \max_{(\mu,\ell)\in B_k} \sum_i \pi_i \left[q_i^{(\mu,\ell)} + \sum_{j\theta} \alpha_j^{n_{(t-1,\ell)}(\pi,\ell,\theta)} p_{ij} r_{j\theta}^{\ell} \right]$$

$$= \max_{\beta\in A(t,k)} \sum_i \pi_i \beta_i$$

with:

$$A(t,k)$$

$$= \left\{ \beta \in \mathbb{R}^N | \beta = q^{(\mu,\ell)} + \sum_\theta PR^{\ell}(\theta) \alpha^{n_{(t-1,\ell)}(\theta)} \right.$$

$$\forall (\mu,\ell) \in B_k,$$

$$\left. n_{(t-1,\ell)} \left| \begin{array}{l} \Theta \mapsto \mathbb{N}^*_{\#A(t-1,\ell)} \\ \theta \to n_{(t-1,\ell)}(\theta) \end{array} \right. \right\}$$

So, $V_t(\pi, k)$ is piecewise linear and convex.

6 Simulation results

In the simulation we have conducted, the physical space is a square with side 25km. This area is divided into 25 square regions (square with side 5km), which are labelled. Whenever an object, the source or the observer, is in the region labelled 'i', it is said to be in state number i (cf. Fig. 3).

1	2	3	4	5
6	7	8	9	10
11	12	13	14	15
16	17	18	19	20
21	22	23	24	25

Fig.3 *State space of observer and target*

At each time period, the observer and the target can stay on the current cell but can also move north, east, south and west, if that movement is possible within the grid. The source motion conforming to a Markov process, a transition matrix (of size 25 × 25) has to be defined. It has been chosen to allow the source to stay on the current cell with probabilitv 0.4 and to move to an adjacent cell with probability 0.6/n [30] where n is the number of accessible cells.

The observation process has also to be defined. To limit the numerical burden, only four observations (sectors) are possible: north-east, south-east, south-west, north-west. To simplify even more, a target which is in sector j is detected in that sector with probability 1 (cf. Fig. 4).

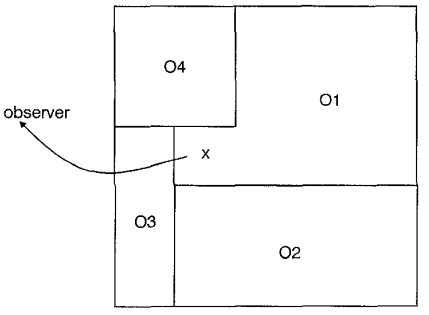

Fig. 4 *Map of observations available from observer that stands at 'x' sign*

The α-vectors were computed for ten iterations, and so can be used for scenarios that contain ten or fewer iterations. The point now is to test whether or not the algorithm provides good estimates of the source's trajectory.

In all the scenarios that will be shown, the trajectory of the source is the continuous line, the observer one is the dashed line and the source trajectory estimation (the highest probability one) is printed with the dot–dashed line. If the source (or its state estimation) or the observer stays on a state more than one iteration, multiple '+' or '*' signs are printed on the diagonal. The cells represent squares of 5km × 5km, and the time duration of an iteration is 500s. This means that the scenarios that are represented in Figs. 5–7 have a total duration of 5000s. (approximately 1h 25 min) and that the target speed is around 10 knots in each case.

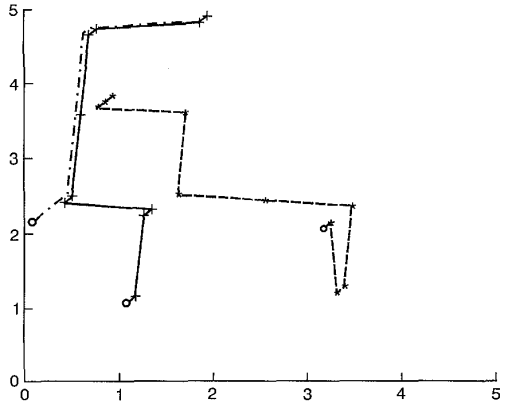

Fig. 5 *Simulation 1*
Graduation 5km

In all the scenarios it may be noticed that the estimation is better at the end than at the beginning. This is because the observer gets closer and closer to the source. This behavioural aspect of the optimal trajectory was pointed out by Olsder [31], Liu [32] and Passerieux *et al.* [33]. When the source is far from the observer, the latter will get the best estimation if it moves orthogonaly to the azimuth of the source. On the other hand, if the source is near to the observer, the latter will have the best estimation if it gets closer to the source. A local coefficient to maximise is $\dot{\beta}/r^2$, where $\dot{\beta}$ is the derivative of the azimuth, and r is the source–observer distance.

The first simulation (Fig. 5) represents a source that is on left side of the grid and that moves upward. The estimation is perfect, except at the beginning where the available information is not sufficient.

The second simulation (Fig. 6), is a source that executes a 'Z' in the middle of the observed surface. The estimation is quite good, even if the last manoeuvre cannot be followed.

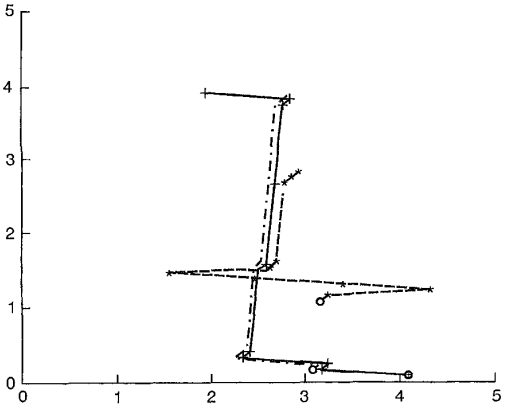

Fig. 6 *Simulation 2*
Graduation 5km

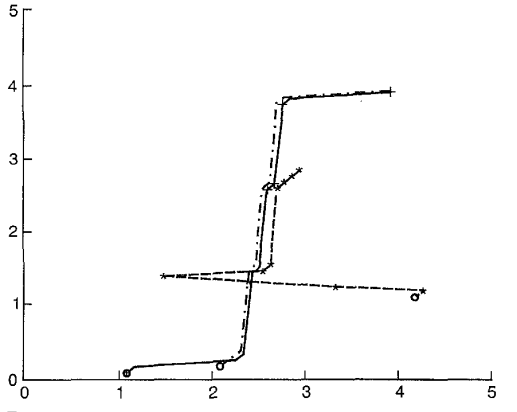

Fig. 7 *Simulation 3*
Graduation 5km

Finally, the third and last simulation (Fig. 7) represents another 'S' trajectory. The initial position of the observer is quite far from the source, but the estimation is perfect except at the beginning.

In all these simulations. almost perfect estimations are obtained with quite coarse information (only four directions). There are numerous reasons for these good results. One reason is that the cells represent squares of 5km × 5km, which means that, if the real source state and its estimation are in the same cell, they can be quite far apart from each other. Another one is that the grid is composed of only 25 cells, which is not enough for real applications.

7 Conclusion

The problem of optimisation of the observer path in the context of manoeuvring target motion analysis has been addressed. Markov chains are used to model the ownship and the target motion. The theory of the partially observable Markov decision process is then utilised to determine an optimal control law for the observer, thus solving a very intricate problem.

Numerous tests have been conducted for a 5×5 state-space grid, giving, in general, good results for the target trajectory estimation. The main difficulty that has been encountered is still a numerical burden, and memory requirements. However, this approach appears to be feasible and useful if the problem is conveniently simplified.

8 Acknowledgment

This work has been supported by DCN/Ing/Sud.

9 References

1 CHANG, C.B., WHITING, R.H., and ATHANS, M.: 'On the state and parameter estimation for maneuvering reentry vehicles', *IEEE Trans. Autom. Control,* 1977, **AC–22**, pp. 99–105

2 HAMPTON, R.L.T., and COOKE, J.R.: 'Unsupervised tracking of maneuvering vehicles', *IEEE Trans. Aerosp. Electron. Syst.,* 1973, **AES–9**, pp. 197--207

3 NEGRON, C.D.: 'Discrimination of motion dynamics for maneuvering target by the application of statistical decision theory'. Proceedings of conferences on *Advance in target tracking*, 1977

4 RICKER, G.C., and WILLIAMS, J.R.: 'Adaptive tracking filter for maneuvering targets', *IEEE Trans. Aerosp. Electron. Syst.,* 1978, **AES–14**, pp. 185–193

5 TENNEY, R.R., BALLARD, T.B., and MILLER, L.E.: 'A dual passive tracking filter for maneuvering targets'. Proceedings of conferences on *Advance in target tracking*, 1977

6 TENNEY, J.S. , HERBERT, R.S., and SANDELL, N.R.: 'A tracking filter for maneuvering sources', *IEEE Trans. Autom. Control,* 1977, **AC–22**, (246–251)

7 THORP, J.S.: 'Optimal tracking of maneuvering targets', *IEEE Trans. Aerosp. Electron. Syst.,* 1973, **AES–9**, pp. 512–519

8 SINGER, R.A.: 'Estimation optimal tracking filter performance for manned maneuvering targets', *IEEE Trans. Aerosp. Electron. Syst.,* 1970, **AES–6**, pp. 473–483

9 GHOLSON, N.H., and MOOSE, R.L.: 'Maneuvering target tracking using adaptive state estimation', *IEEE Trans. Aerosp. Electron. Syst.,* 1977, **AES–13**, (3), pp. 310–317

10 McAULAY, R.J., and DENLINGER, E.J.: 'A decision-directed adaptive tracker', *IEEE Trans. Aerosp. Electron. Syst.,* 1973, **AES–9**, pp. 229–236

11 MOOSE, R.L.: 'An adaptive state estimation solution to the maneuvering target problem', *IEEE Trans. Autom. Control,* 1975, **AC–20**, pp. 359–362

12 MOOSE, R.L., VANLANDINGHAM, H.F., and McCABE, D.H.: 'Modeling and estimation for tracking maneuvering targets', *IEEE Trans. Aerosp. Electron. Syst.,* 1979, **AES–15**, pp. 448–455

13 LIN, H.-J., and ATHERTON, D.P.: 'An investigation of the sfimm algorithm for tracking maneuvering targets'. Proceedings of the 32nd conference on *Decision and control*, 1993, pp. 930–935

14 STREIT, R.L., and BARRETT, R.F.: 'Frequency line tracking using hidden markov models', *IEEE Trans. Acoust. Speech Signal Process.,* 1990, **38**, (4), pp. 586–598

15 XIANYA, X.: 'Multiple frequency line tracking using hidden markov/a models'. Proccedings on the 29th conference on *Decision and control*, 1990

16 XIE, X., and EVANS, R.J.: 'Multiple target tracking and multiple frequency line tracking using hidden markov models', *IEEE Trans. Signal Process,* 1991, **39**, (12), pp. 2659–2676

17 XIE, X., and EVANS, R.J.: 'Multiple frequency line tracking with hidden markov models – further results', *IEEE Trans. Signal Process.,* 1993, **41**, (1), pp. 334–343

18 BARNIV, Y.: 'Dynamic programming solution for detecting dim moving target', *IEEE Trans. Aerosp. Electron. Syst.,* 1985, **AES–21**, (1), pp. 144–156

19 BARNIV, Y., and KELLA, O.: 'Dynamic programming solution for detecting dim moving target. Part 2. Analysis', *IEEE Trans. Aerosp. Electron. Syst.,* 1987, **AES–23**, (6), pp. 776–788

20 MARTINERIE, F.: 'New data fusion and tracking approaches in multiple target/ distributed sensors network context'. Proceedings of IEEE international conference on *Acoustics, speech andsignal processing* (ICASSP), 1993

21 MARTINERIE, F., and FORSTER, P.: 'Data association and tracking from distributed sensors using hidden Markov models and evidential reasoning'. Proceedings of IEEE conference on *Detection and control*, 1992

22 LE CADRE, J.-P., and TRÉMOIS, O.: 'Bearings-only tracking of maneuvering sources', *IEEE Trans. Aerosp. Electron. Syst.,* 1998, **34**, (1), pp. 179–193

23 TRÉMOIS, O.: 'Étude de méthodes de trajectographie pour des sources manoeuvrantes'. PhD thesis, Université de Rennes I, 1995

24 TRÉMOIS, O., and LE CADRE, J.-P.: 'Maneuvering target motion analysis using hidden markov model'. Proceedings of the IEEE international conference on *Acoustics, speech and signal processing*

25 MONAHAN, G.E.: 'A survey of partially observable markov decision processes: theory, models and algorithms', *Managem. Sci.,* 1982, **28**, (1), pp. 1–16

26 SMALLWOOD, R.D., and SONDIK, E.J.: 'The optimal control of partially observable markov processes over a finite horizon', *Operat. Res.,* 1973, **21**, pp. 1071–1088

27 CHENG, H.: 'Algorithms for partially observed Markov decision processes'. PhD thesis, Faculty of commerce and business administration, University of British Columbia, 1988

28 LOVEJOY, W.S.: 'A survey of algorithmic methods for partially observed Markov decision processes', *Ann. Operat. Res.,*, **28**, pp. 47–66

29 LE CADRE, J.-P., and TRÉMOIS, O.: 'The matrix dynamic programming property and its implications', *SIAM J. Matrix Anal. Appl.,* 1997, **18**, (4), pp. 818–826

30 EAGLE, J.N.: 'The optimal search for a moving target when the search path is constrained', *Operat. Res.,* 1985, **32**, pp. 1107–1114

31 OLSDER, G.T.: 'On the optimal maneuvering during bearings-only tracking'. Proceedings of 23rd conference on *Decision and control*, 1984

32 LIU, P.T.: 'An optimum approach in target tracking with bearing measurements', *J. Opt. Theory Appl.,* 1988, **56**, (2), pp. 205–214

33 PASSERIEUX, J.-M., and VAN CAPPEL, D.: 'Manoeuvre optimale en trajectographie passive à partir d'azimuts'. presented at Treizième colloque GRETSI, 1991