ALMA MATER STUDIORUM
UNIVERSITA' DI BOLOGNA

SCUOLA DI INGEGNERIA E ARCHITETTURA
- Sede di Forlì -

CORSO DI LAUREA IN
INGEGNERIA AEROSPAZIALE
Classe: L-9

ELABORATO FINALE DI LAUREA

In **Sistemi di Radiocomunicazione**

Analisi di un sistema di tracking per target multipli tramite approccio PHD

CANDIDATO

Nicola Zavatta

RELATORE

Prof. Ing. Enrico Paolini

CORRELATORI

Dott. Ing. Matteo Mazzotti

Prof. Ing. Andrea Giorgetti

Anno Accademico 2012-2013
Sessione I

Ringraziamenti

Prima di iniziare a parlare della tesi che ho sviluppato, vorrei ringraziare chi, con la sua presenza e il suo aiuto lo ha reso possibile.
Un ringraziamento più che doveroso va al mio relatore, Prof. Enrico Paolini, che ha accettato di prendermi prima come tirocinante e poi come laureando.
Nel corso dei mesi in cui ho lavorato con lui ho avuto modo di apprezzarne le doti umane, oltre che professionali, avendo sempre dato prova di grande disponibilità e precisione. Per questo lo ringrazio.
Lo stesso genere di ringraziamento lo vorrei rivolgere ai suoi colleghi, nonché miei correlatori, Dott. Mazzotti e Prof. Giorgetti che, per quanto non abbia potuto approfondire più di tanto la loro conoscenza, si sono dimostrati molto cordiali nei miei confronti.
Non vorrei dilungarmi troppo, ma credo che loro tre meritino un plauso particolare perché con la loro serietà e competenza contribuiscono a rendere l'università un posto migliore.
Vorrei ringraziare i compagni di corso con cui ho condiviso l'esperienza universitaria tra esami e battute.
Un ringraziamento di cuore ai miei amici più cari e a mio cugino, che non hanno mai smesso di credere in me, e alla mia ragazza, il cui contributo alla stesura della tesi è stato molto importante.
Infine un grazie va ai parenti che mi sono stati vicini, in particolar modo i miei genitori, mia nonna, le mie zie e mio zio.
Potrei continuare e forse sarebbe ripetitivo, mi limito a rivolgermi a tutti coloro che hanno contribuito alla mia formazione con un sincero

GRAZIE

Nicola Zavatta, Rimini 19/07/2013

Indice

Elenco delle figure

Tabelle

Introduzione

In molte applicazioni pratiche in ambito aeronautico, ma non solo, è richiesto di stimare la presenza e la posizione di uno o più oggetti non direttamente identificabili.
Questo problema, che ha trovato proprio in aeronautica una delle sue prime soluzioni con lo sviluppo del radar a partire dagli anni ’30, viene oggi risolto tramite sistemi capaci di fornire un’approssimazione della posizione, ma anche di altre grandezze, dell’oggetto osservato, o *target*. Tali sistemi prendono il nome di *filtri* e, con il grande sviluppo dei calcolatori, le loro prestazioni hanno raggiunto notevoli livelli di precisione. Uno dei principali settori del filtraggio è quello *multi-target*, che analizza cioè situazioni in cui siano coinvolti più target contemporaneamente, come può accadere lungo le aerovie o in numerose applicazioni militari.
In questo elaborato ci proponiamo di introdurre i concetti fondamentali che stanno alla base del filtraggio multi-target, così come le diverse problematiche connesse, con particolare riguardo alle applicazioni di *tracking*, cioè di inseguimento del target. Dopodiché discuteremo un possibile approccio alla realizzazione di un filtro di questo tipo e vedremo come è possibile implementarlo praticamente su un calcolatore.
Nella seconda parte della tesi saranno illustrati i risultati sperimentali che si ottengono dall’implementazione.
Operativamente parlando, nel Capitolo 1 vengono introdotti i filtri multi-target a partire da una prospettiva Bayesiana, nel Capitolo 2 si studia il filtro PHD, che andremo poi a implementare e di cui mostreremo le simulazioni nel Capitolo 3. Chiudono l’elaborato il Capitolo 4 conclusivo e un’Appendice in cui sono presentati alcuni utili strumenti matematici e in cui è presente il codice *Matlab* dell’algoritmo impiegato.

Capitolo 1

Il filtraggio multi-target

Il filtraggio multi-target costituisce essenzialmente una generalizzazione del più semplice filtraggio a singolo target.
Come in quest'ultimo, il risultato che si vuole ottenere è una stima dello stato dei target presenti in una certa regione di spazio, detta regione di sorveglianza, in un certo intervallo temporale.
Rispetto al filtraggio single-target, tuttavia, quello multi-target deve tenere conto di diversi fattori aggiuntivi.
Il primo problema da affrontare è la stima del numero di target presenti, osservando che tale quantità in generale è variabile nel tempo.
Questo pone una seconda questione, vale a dire lo sviluppo di un metodo che tenga conto dei fenomeni di nascita di nuovi target e scomparsa di quelli già presenti.
A questi problemi si aggiunge infine quello di mantenere sempre un corretto aggancio con il target in modo da evitare di scambiare tra loro target vicini.
Queste questioni saranno trattate in dettaglio nei paragrafi seguenti.

1.1 Random Finite Sets

Iniziando la trattazione di problemi di filtraggio multi-target è innanzitutto necessario identificare gli strumenti matematici adeguati allo scopo. Nel caso del filtraggio a target singolo si utilizza normalmente una descrizione nello spazio degli stati, la quale utilizza elementi dell'algebra lineare, modellando lo stato del target e le misure effettuate dai sensori come vettori di variabili aleatorie. Le dimensioni di tali vettori in generale non variano nel tempo e sono note a priori, consentendo un approccio analitico di tipo "tradizionale". Può essere utile domandarsi in che modo sia possibile estendere questa modellizzazione al caso multi-target.
In generale si nota immediatamente che il numero di target, e perciò le dimensioni dello stato e delle misure, non solo non è noto, ma è anche variabile nel tempo, ponendo una prima difficoltà. Per modellare lo scenario in esame si introducono delle entità matematiche chiamate *Random Finite Sets* (RFS).

Nel seguito descriveremo brevemente in cosa consistono tali entità.
Si schematizzino lo stato e le misure dei vari target come insiemi X e Z a dimensione finita, rispettivamente N ed M. Tali insiemi varieranno col tempo e in particolare, considerando un modello tempo-discreto, al k-esimo istante risulterà $X = X_k$ e $Z = Z_k$.
Per quanto riguarda le cardinalità di tali insiemi, per quanto detto sopra saranno anch'esse funzioni del tempo, $N = N_k$ ed $M = M_k$.
Inoltre, a causa dell'incertezza sullo stato e sulle misure, gli elementi $x_{k,1},..x_{k,N_k} \in X_k$ e $z_{k,1},..z_{k,M_k} \in Z_k$ saranno delle variabili aleatorie, analogamente a quanto avviene con target singoli.
In termini matematici più precisi i Random Finite Sets possono essere descritti tramite il formalismo seguente che, seppur pesante, risulterà utile per la trattazione degli argomenti successivi. Sia $Q \subseteq R^n$ uno spazio compatto e $F(Q)$ la famiglia dei sottoinsiemi finiti di Q. Sia inoltre (Ω, A, P) uno spazio di probabilità. Un Random Finite Set Ψ è definito come una funzione misurabile $\Psi : \Omega \to F(Q)$ e la misura di probabilità viene definita tramite la distribuzione $P_\Psi(\mathrm{T}) = P(\{\omega : \Psi(\omega) \in \mathrm{T}\})$. Particolare attenzione va posta nella definizione della densità di probabilità p_Ψ, calcolata come derivata di Radon-Nikodym della distribuzione P_Ψ con un'opportuna misura μ tale che $P_\Psi(\mathrm{T}) = \int_{\mathrm{T}} p_\Psi(X)\mu(dX)$.
Solitamente per tale misura si assume la forma $\mu(\mathrm{T}) = \sum_{i=0}^{\infty} \frac{\lambda^i(\mathrm{T} \cap E^i)}{i!}$, dalla quale discende $P_\Psi(\mathrm{T}) = \sum_{i=0}^{\infty} \frac{1}{i!} \int_{S^i} p_\Psi(\{x_1,..x_i\})\lambda^i(x_1..x_i)$, con $\mathrm{T} = \bigcup_{i=0}^{\infty} S^i$.
Per quanto riguarda le nozioni di derivata su un insieme e di integrale di un insieme, ulteriori dettagli sono forniti in Appendice.

Prima di concludere questa sezione è opportuno fare alcune considerazioni di carattere generale in merito all'argomento trattato.
Come abbiamo visto, abbiamo introdotto i Random Finite Sets a partire da una generalizzazione del caso a singolo target. In questo senso è stata ottenuta una estensione delle strutture matematiche normalmente impiegate nel problema del filtraggio, in grado di rappresentare modelli molto generali e articolati.

Da un punto di vista cronologico, questo
tipo di insiemi di derivazione statistica è stato introdotto nel campo del tracking solo in tempi molto recenti e ha contribuito ad offrire un approccio al caso multi-target che risultasse il più unitario possibile, favorendo inoltre le operazioni di *data-fusion* di cui però non ci occuperemo.
Al termine del prossimo paragrafo saranno discussi anche i principali problemi legati a questo tipo di rappresentazione e verrà esposta una possibile soluzione.

1.2 Filtri Bayesiani

Nel caso a singolo target il problema del filtraggio può essere descritto in termini statistici giungendo alla formulazione di una categoria di filtri che prendono il nome di filtri Bayesiani. In particolare si può dimostrare come il filtro di Kalman, che come noto rappresenta lo stimatore ottimo in situazioni di questo tipo, rappresenti un caso particolare di filtro di Bayes. Naturalmente è possibile offrire una formulazione in termini Bayesiani che è del tutto equivalente a quella del filtro di Kalman classico. L'idea che sta alla base dell'approccio Bayesiano si basa sulla propagazione di una funzione di densità di probabilità associata al target piuttosto che su quella dello stato del target stesso. I passi di predizione e correzione rimangono invariati nei loro ruoli, seppur con ovvie modifiche nelle rispettive espressioni formali.
Introducendo i Random Finite Sets nella teoria dei filtri Bayesiani si ottiene una estensione al caso multi-target che sia la più generale possibile. Naturalmente la funzione che viene propagata sarà ora la densità di probabilità di un Random Finite Set e andrà calcolata secondo le regole esposte in precedenza. Nelle seguito andremo a presentare i termini che compaiono nelle equazioni del filtro di Bayes e cercheremo di spiegarne il significato.
Indichiamo con E_s ed E_o gli spazi degli stati e delle misure rispettivamente e con Ψ_k e Σ_k i Random Finite Sets su tali spazi.
Sia $\Psi_k \in U$ con $U \subseteq F(E_s)$ e $\Sigma_k \in V$ con $V \subseteq F(E_o)$.

Data una realizzazione X_k dello stato all'istante k-esimo, all'istante successivo lo stato del target potrà essere espresso come $\Psi_{k+1} = S_{k+1}(X_k) \cup B_{k+1}(X_k) \cup \Gamma_{k+1}$, dove $S_{k+1}(X_k)$ indica il RFS dei target presenti all'istante k-esimo che sono sopravvissuti, $B_{k+1}(X_k)$ il RFS dei target generati al nuovo istante e Γ_{k+1} quello dei target apparsi spontaneamente. Allo stesso modo all'istante k-esimo l'osservazione potrà essere espressa dal RFS $\Sigma_k = E_k(X_k) \cup C_k(X_k)$, essendo $E_k(X_k)$ il RFS delle misure allo stesso istante e $C_k(X_k)$ quello dovuto a fenomeni di clutter o ad altri disturbi.
Possiamo ora definire le distribuzioni di probabilità su tali insiemi. Chiamiamo $P_{k|k}(U|Z_k) = P(\Psi_k \in U|Z_k)$ la distribuzione di probabilità a posteriori che viene propagata dal filtro, $P_{k|k-1}(U|X_{k-1}) = P(\Psi_k \in U|X_{k-1})$ la distribuzione di Markov, che descrive il moto del target, e $P_{k|k-1}(V|X_k) = P(\Sigma_k \in V|X_k)$ la distribuzione delle misure. Per quanto riguarda le densità, indichiamo con $p_{k|k}(X_k|Z_k)$ la densità a posteriori, con $f_{k|k-1}(X_k|X_{k-1})$ la densità di transizione e con $g_k(Z_k|X_k)$ la funzione di verosimiglianza. Ricordiamo che queste funzioni si ottengono come derivate di Radon-Nikodym delle precedenti, risultando perciò:

$$P_{k|k}(U|Z_k) = \int_U p_{k|k}(X_k|Z_k)\mu_s(dX_k),$$

$$P_{k|k-1}(U|X_{k-1}) = \int_U f_{k|k-1}(X_k|X_{k-1})\mu_s(dX_k),$$

$$P_k(V|X_k) = \int_V g_k(Z_k|X_k)\mu_o(dZ_k),$$

dove con μ_s e μ_o sono state indicate le misure su $F(E_s)$ e su $F(E_o)$.
Vediamo ora le equazioni che costituiscono il filtro Bayesiano vero e proprio, vale a dire gli stadi del predittore e del correttore.
Come anche nel filtraggio a singolo target, le equazioni rappresentano un algoritmo ricorsivo articolato su un primo passo di predizione, basato sul modello di moto del target, e su un secondo di correzione, al seguito delle misure ricevute dai sensori asserviti al sistema.

Di seguito presentiamo le espressioni formali dei due stadi:

$$p_{k|k-1}(X_k|Z_{k-1}) = \int f_{k|k-1}(X_k|X)p_{k-1|k-1}(X|Z_{k-1})\mu_s(dX),$$

$$p_{k|k}(X_k|Z_k) = \frac{g_k(Z_k|X_k)p_{k|k-1}(X_k|Z_{k-1})}{\int g_k(Z_k|X)p_{k|k-1}(X|Z_{k-1})\mu_s(dX)}.$$

Come si può vedere, il filtro di Bayes è retto da equazioni integrali non lineari. Nel caso multi-target in esame, queste richiedono la valutazione di integrali di insiemi, che complicano la situazione rispetto al caso a singolo target. La soluzione di tali equazioni non è in generale ottenibile in forma chiusa, obbligando ad impostare dei metodi risolutivi per via numerica.
Un approccio possibile in questa direzione consiste nel discretizzare il problema tramite un insieme di particelle, da cui il nome di filtro particellare. Ad ognuna di queste particelle viene poi associato un peso. Le particelle e i pesi vengono aggiornati tramite un opportuno algoritmo per seguire l'evoluzione del moto dei target. In generale il numero di particelle possono variare col tempo e al termine di ogni scansione temporale viene eseguita un'operazione di ricampionamento. In questo algoritmo l'operazione di integrazione viene eseguita sommando i pesi complessivi delle varie particelle. Questo metodo consente di propagare interamente la densità a posteriori, superando i problemi posti dalla risoluzione analitica. Il filtro che si ricava è molto generale e permette di trattare problemi non lineari e con target il cui numero vari dinamicamente nel tempo. Tuttavia, affinché i risultati siano corretti, è necessario effettuare un campionamento adeguato su spazi che possono assumere dimensioni molto elevate. Tipicamente la scelta di una distribuzione di campionamento adatta può risultare molto difficile.
Peggio ancora, per le ragioni appena citate, l'efficienza di algoritmi di questo tipo è fortemente legata al numero di target presenti e può arrivare a decrescere esponenzialmente con il loro aumento.
Per tutti questi motivi si preferisce ricercare algoritmi più semplici che permettano una risoluzione meglio trattabile dal punto di vista numerico. Uno di questi metodi è l'approccio PHD di cui parleremo nel prossimo capitolo.

Capitolo 2

Il filtro PHD

2.1 Probability Hypothesis Density

Nel Capitolo 1 è stato introdotto il problema del filtraggio multi-target. Sono stati presentati strumenti matematici idonei per la sua trattazione; inoltre, partendo da alcune analogie con il caso single target, sono stati descritti i filtri di Bayes, che rappresentano l'ottimo teorico in termini di stima dello stato. Purtroppo si è visto come, nella sua formulazione completa, questa categoria di filtri sia difficilmente applicabile a situazioni reali per via della sua complessità computazionale. Per questo motivo ci siamo orientati verso lo studio di soluzioni semplificate al problema del filtraggio multi-target. In questo elaborato ci occuperemo di una di queste, che prende il nome di Probability Hypothesis Density o PHD. In questo approccio la quantità che viene propagata è la PHD del target, cioè un'approssimazione al primo ordine della sua densità di probabilità a posteriori. La PHD è quindi un momento del primo ordine e si può considerare come l'analogo del valore atteso dello stato del target nel caso di tracking per singolo target.
Per quanto detto, i risultati ottenuti con questo metodo saranno anch'essi delle approssimazioni di quelli teorici Bayesiani, tanto migliori quanto più accurato risulterà l'algoritmo implementato.
È possibile dimostrare che, sotto l'ipotesi che la densità multi-target assuma la forma di una distribuzione di Poisson, il filtro PHD fornisce la soluzione esatta al problema e definisce completamente la dinamica del processo di moto. Per quanto riguarda le proprietà della funzione PHD, possiamo notare che essa è definita sullo spazio degli stati del target, ma non rappresenta una densità di probabilità propriamente detta. Infatti, pur essendo una funzione a valori positivi e integrabile, il suo integrale non assume necessariamente valore unitario. Più in generale, l'integrale della PHD su una determinata regione spaziale indica il numero stimato di target presenti nella suddetta regione.

A titolo di esempio, una PHD su uno spazio S, tale che il suo integrale su S dia come risultato 2.634, rappresenta, arrotondando all'intero più vicino, 3 target diversi.
Naturalmente il valore dell'integrale della PHD su una stessa regione varia dinamicamente al passare del tempo, a seconda che sulla scena compaiano nuovi target e che quelli già presenti permangano oppure spariscano. Inoltre in corrispondenza dei target la PHD assumerà i valori più elevati.
Pertanto, stimando i picchi della funzione PHD a variare del tempo, è possibile stimare la posizione dei target istante per istante.
Di seguito forniremo una descrizione formale del filtro PHD, dopodiché vedremo come è possibile implementarlo in applicazioni pratiche.
Indichiamo con S la regione di sorveglianza su cui effettuare l'operazione di tracking. Assumendo che lo stato x del target abbia dimensione r, risulterà $S \subseteq R^r$. Chiamiamo $\pi_n(x_1,..x_n)$ la densità di probabilità congiunta degli stati condizionata alla presenza di n target e p_n la probabilità che il loro numero sia esattamente pari a n.
La probabilità che in un intorno di x sia presente un target, è data dalla somma delle densità degli n target distinti:

$$PHD(x|n) = \int \pi_n(x, y_1,...y_{n-1})dy + \int \pi_n(y_1, x, y_2,...y_{n-1})dy + ...\int \pi_n(y_1,...y_{n-1}, x)dy,$$

dove $dy = dy_1 dy_2..dy_{n-1}$. Dato che la densità $\pi_n(x_1,..x_n)$ è invariante rispetto alla permutazione dei suoi argomenti si può scrivere anche

$$PHD(x|n) = n\int \pi_n(x, y_1,...y_{n-1})dy.$$

Detta $f_{k|k}$ la funzione di verosimiglianza a posteriori e Z_k l'insieme delle misure, la PHD può essere espressa nella forma:

$$PHD(x) = \sum_{n=0}^{\infty}(n+1)p_{n+1}\int \pi_{n+1}(x, y_1,...y_n)dy = \sum_{n=0}^{\infty}\frac{1}{n!}\int f_{k|k}(\{x, y_1,...y_n\}|Z_k)dy,$$

dove $dy = dy_1 dy_2..dy_n$. In termini di integrali di insiemi la formula precedente può anche essere riscritta in modo più compatto come

$$PHD(x) = \int f_{k|k}(\{x\} \cup Y)\delta Y = \sum_{n=0}^{\infty}\frac{1}{n!}\int f_{k|k}(\{x\} \cup \{y_1,...y_n\})dy.$$

Chiamiamo $b_{k|k-1}(y)$ la PHD di nascita spontanea di un target, $d_{k|k-1}(x)$ la probabilità che il target scompaia passando dalla scansione $k-1$ alla scansione k, $f_{k|k-1}(y|x)$ la densità di probabilità di transizione e $b_{k|k-1}(y|x)$ la PHD di nascita di un nuovo target da uno già esistente.
Lo stadio di predizione del filtro PHD è costituito dalla seguente equazione:

$$D_{k|k-1}\left(y\middle|Z^{k-1}\right)=b_{k|k-1}(y)+\int D_{k|k-1}(y|x)D_{k-1|k-1}\left(x\middle|Z^{k-1}\right)dx,$$

con

$$D_{k|k-1}(y|x)=\left(1-d_{k|k-1}(x)\right)f_{k|k-1}(y|x)+b_{k|k-1}(y|x).$$

Per modellare le misure $Z^k=\{z_1,..z_m\}$ ricevute all'istante k-esimo e inserirle nel passo di correzione, introduciamo la probabilità di rilevamento P_d, il numero medio di falsi allarmi per scansione λ_k, che assumiamo distribuito secondo Poisson, la loro densità spaziale $c_k(z)$ e infine la funzione di verosimiglianza dei sensori $f_k(z|x)$. Possiamo ora introdurre anche lo stadio correttore, descritto dall'espressione:

$$D_{k|k}\left(x\middle|Z_k\right)=\sum_{z\in Z_k}\frac{P_d D_k(z)}{\lambda_k c_k(z)+P_d D_k(z)}D_k\left(x\middle|z\right)+(1-P_d)D_{k|k-1}\left(x\middle|Z_{k-1}\right),$$

con

$$D_k(z)=\int f_k(z|x)D_{k|k-1}\left(x\middle|Z_{k-1}\right)dx \text{ e } D_k\left(x\middle|z\right)=\frac{f_k(z|x)D_{k|k-1}\left(x\middle|Z_{k-1}\right)}{D_k(z)}.$$

Il numero stimato di target, stando a quanto detto prima, è dato da:

$$T_{k|k}=\int_S D_{k|k}(x)dx.$$

Inoltre, poiché in corrispondenza dei picchi di $D_{k|k}\left(x\middle|Z_k\right)$ si ha la maggior "densità di presenza" dei target, ricercando i $[T_{k|k}]$ massimi di tale funzione, dove con $[x]$ si è indicata la funzione parte intera di x, è possibile localizzare le posizioni dei vari target presenti sulla scena.

Come si può vedere, le equazioni che descrivono la propagazione della PHD, e in particolare quella del correttore, sono decisamente non lineari e come tali non ammettono soluzione in forma chiusa neppure nell'ipotesi di moto lineare Gaussiano. Per questo motivo nella pratica si introducono degli algoritmi che permettono la risoluzione per via numerica. Un esempio è costituito dal metodo Monte Carlo a particelle che descriveremo in dettaglio nel prossimo paragrafo.

2.2 Implementazione del filtro PHD

Passiamo ora a presentare l'implementazione Monte Carlo del filtro PHD che abbiamo impiegato e che costituisce il principale contributo di questo elaborato.
Tale metodo si basa sostanzialmente sulla rappresentazione discreta della PHD tramite un insieme di particelle pesate, con un approccio concettualmente simile a quello visto brevemente per i filtri di Bayes.
Trattandosi di un metodo Monte Carlo, viene fatto uso di tecniche ricorsive basate su processi aleatori per poter giungere alla stima della soluzione. Il risultato che si ottiene è un algoritmo molto generale, capace di trattare problemi non lineari e non Gaussiani.
Nel lavoro qui presentato, tuttavia, ci limitiamo ad analizzare un modello di moto lineare Gaussiano.
Innanzitutto ipotizziamo che i target si muovano di moto piano su uno spazio di coordinate cartesiane x-y. La valutazione della dinamica dei target viene effettuata a istanti di tempo discreti.
Lo stato di ciascun target è costituito da coordinate di posizione e velocità, mentre i sensori forniscono solo misure relative alla posizione.
Indichiamo con $\mathrm{x}(k)$ lo stato reale del target ordinato come vettore colonna:

$$\mathrm{x}(k) = [x(k), \dot{x}(k), y(k), \dot{y}(k)]^T .$$

Lo stato all'istante successivo è descritto dall'espressione:

$$\mathrm{x}(k+1) = F\,\mathrm{x}(k) + \nu(k),$$

dove $\nu(k)$ è un vettore di variabili aleatorie Gaussiane la cui matrice di covarianza viene indicata con $Q(k)$ e F è una matrice che descrive il moto rettilineo uniforme e ha forma

$$F = \begin{bmatrix} 1 & T & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T \\ 0 & 0 & 0 & 1 \end{bmatrix},$$

in cui T è il tempo di campionamento.
Per quanto riguarda il sensore asservito al sistema di tracking, possiamo ipotizzare che sia costituito da un'antenna radar.
Come avviene solitamente in questi casi, di spiccato interesse aeronautico, le rilevazioni ottenute consistono della distanza radiale sorgente-ricevitore e di una misura angolare della posizione. Dal punto di vista della modellizzazione, possiamo pertanto schematizzare le misure come un vettore di coordinate polari

$$z = [r_m, \theta_m]^T = [r + w_r, \theta + w_\theta]^T,$$

in cui r è la distanza radiale, θ è l'angolo di azimuth e w_r e w_θ sono i relativi rumori, a media nulla e varianze $E[w_r^2] = \sigma_r^2$ e $E[w_\theta^2] = \sigma_\theta^2$.
Tali misure possono essere convertite in coordinate cartesiane tramite le note relazioni:

$$\begin{cases} x_m = r_m \cos(\theta_m) \\ y_m = r_m \sin(\theta_m) \end{cases}.$$

La matrice di covarianza delle misure $R(z)$ può invece essere ottenuta per linearizzazione dei singoli termini, da cui risulta

$$R(z) = \begin{bmatrix} \mathrm{var}(x_m) & \mathrm{cov}(x_m, y_m) \\ \mathrm{cov}(x_m, y_m) & \mathrm{var}(y_m) \end{bmatrix},$$

dove

$$\mathrm{var}(x_m) = r_m^2\sigma_\theta^2 \sin^2(\theta_m) + \sigma_r^2 \cos^2(\theta_m),$$

$$\mathrm{var}(y_m) = r_m^2\sigma_\theta^2 \cos^2(\theta_m) + \sigma_r^2 \sin^2(\theta_m),$$

$$\mathrm{cov}(x_m, y_m) = (\sigma_r^2 - r_m^2\sigma_\theta^2)\sin(\theta_m)\cos(\theta_m).$$

Iniziamo ora a descrivere nel dettaglio la procedura iterativa che costituisce il metodo Monte Carlo. Come prima cosa è necessario provvedere all'inizializzazione di un certo numero di particelle.
Per fare questo, all'istante $k = 1$ generiamo N_z particelle attorno ad ogni misura. Le componenti di posizione di tali particelle sono estratte da una distribuzione normale $N(z, R(z))$ di media z e covarianza $R(z)$, mentre le componenti di velocità sono ricavate da una distribuzione uniforme $U[-v_{\max}, v_{\max}]$, in cui $v_{\max}$ è un parametro di progetto.
Ogni particella è costituita da due elementi, un campione $\xi^{(i)}$ dallo spazio degli stati e un peso $\omega^{(i)}$, per $i = 1,2,...N$.
Il numero totale di particelle alla prima scansione è dato, detto Z_1 l'insieme delle misure ricevute e $|Z_1|$ la sua cardinalità, da $N = N_z|Z_1|$.
In questo primo passo a tutte le particelle viene attribuito lo stesso peso. In particolare, poiché la somma dei pesi indica il numero totale dei target, dovrà risultare $\omega^{(i)} = \frac{|Z_1|}{N} = \frac{1}{N_z}$.
A partire dalla seconda scansione inizia la procedura di aggiornamento delle particelle e dei pesi. Durante la fase di predizione le particelle vengono propagate secondo il modello di moto che abbiamo definito in precedenza

$$\xi_{k|k-1}^{(i)} = F\xi_{k-1}^{(i)} + \nu(k)$$

mentre i pesi vengono aggiornati secondo $\omega_{k|k-1}^{(i)} = \omega_{k-1}^{(i)} P_s$, dove P_s è la probabilità di sopravvivenza dei target.

A questo punto vengono inizializzate delle nuove particelle per tenere conto dell'eventuale nascita di nuovi target. In particolare generiamo N_z particelle per ogni misura $z \in Z_k$. Come nella fase di inizializzazione, le componenti dello stato sono tratte dalle distribuzioni $N(z, R(z))$ e $U[-v_{\max}, v_{\max}]$. Inoltre a tutte le nuove particelle vengono dati pesi uguali, la cui somma è pari alla PHD di nascita di nuovi target su tutto lo spazio di sorveglianza S, che chiamiamo $b_{k|k-1}(S)$. Detto $N_{new} = N_z|Z_k|$ il numero delle nuove particelle, l'espressione dei pesi è data da $\omega_{k|k-1}^{(i)} = \frac{b_{k|k-1}(S)}{N_{new}}$ per $i = 1,2,...N_{new}$.

Il numero totale di particelle presenti è ora $N + N_{new}$, pari cioè al numero di particelle al termine della scansione precedente più quelle proposte per esplorare nuovi target.

A questo punto interviene il passo di correzione, il cui scopo è sostanzialmente quello di "sbilanciare" i pesi delle singole particelle in modo da scegliere quelle più importanti. In altri termini, i pesi vengono modificati affinché le particelle che meglio approssimano la PHD dei target presentino valori più elevati. Ovviamente la somma totale dei pesi, cioè il numero di target, rimane invariato.

Per ogni $z \in Z_k$ calcoliamo dapprima la formula seguente:

$$C(z) = \sum_{j=1}^{N+N_{new}} P_d \, f\left(z|\xi^{(j)}\right) \omega_{k|k-1}^{(j)},$$

dopodiché aggiorniamo i pesi tramite:

$$\omega_k^{(i)} = \left[1 - P_d + \sum_{z \in Z_k} \frac{P_d \, f\left(z|\xi^{(i)}\right)}{\lambda_k c_k(z) + C(z)}\right] \omega_{k|k-1}^{(i)} \qquad \text{per } i = 1,2,...N + N_{new},$$

in cui i termini P_d, λ_k e $c_k(z)$ rappresentano, come descritto in precedenza, la probabilità di rilevamento, il numero medio di falsi allarmi e la loro densità, mentre $f\left(z|\xi^{(i)}\right)$ è la funzione di verosimiglianza calcolata sulle particelle.

Una questione molto interessante è rappresentata dalla ricerca dei picchi della PHD, dal momento che la loro conoscenza implica la localizzazione dei target. Il problema della ricerca dei massimi non è banale, in quanto possono insorgere difficoltà legate alla "clusterizzazione", cioè al raggruppamento delle particelle.
Per esempio si potrebbe pensare di stimare gli M picchi maggiori limitandosi a considerare le M particelle di peso più elevato.
Sebbene questo metodo possa sembrare corretto e di semplice implementazione, è sufficiente una semplice osservazione per rendersi conto che esso può determinare errori grossolani.
Infatti abbiamo sì visto che i pesi forniscono una misura della bontà di approssimazione delle relative particelle, tuttavia non è dato alcun collegamento tra i target reali e le particelle che li rappresentano.
Pertanto può benissimo capitare che un target venga risolto meglio degli altri dall'algoritmo di tracking e dunque le particelle che lo descrivono abbiano pesi molto elevati.
Con riferimento all'esempio precedente, le M particelle più pesanti apparterrebbero probabilmente a questo target e sarebbero molto vicine tra loro. Come risultato, la ricerca dei picchi si risolverebbe immancabilmente in un fallimento.
Per evitare di incorrere nei problemi sopra descritti, si cerca di effettuare una ricostruzione della funzione PHD, problema che può essere affrontato in vari modi. La strada da noi percorsa consiste in un metodo che prende il nome di algoritmo *expectation-maximization* (EM).
Sostanzialmente, approssimiamo la PHD con una *Mistura Gaussiana,* vale a dire una combinazione lineare di funzioni Gaussiane, tenendo conto dei pesi all'interno dei coefficienti della combinazione.
Il numero di funzioni di densità di probabilità normali impiegate alla k-esima scansione è pari a $M = |Z_k| + [T_{k-1|k-1}]$, in cui $[T_{k-1|k-1}]$ è il numero stimato di target alla scansione precedente. I valori di media e covarianza delle $[T_{k|k}]$ Gaussiane più pesanti, cioè con i coefficienti più alti, sono assunti come picchi della PHD all'istante k.
I passi della procedura sono presentati di seguito:

$$\delta_{im} = \kappa_m \frac{1}{\sqrt{\det(2\pi\Omega_m)}} e^{-\left(\frac{1}{2}\left(\xi_k^{(i)}-\mu_m\right)^T \Omega_m^{-1}\left(\xi_k^{(i)}-\mu_m\right)\right)} \quad (1)$$

$$\delta_{im} = \frac{\omega_k^{(i)} \delta_{im}}{\sum_{l=1}^{M} \delta_{il}} \quad (2)$$

$$\mu_m = \frac{\sum_{i=1}^{N+N_{new}} \xi_k^{(i)} \delta_{im}}{\sum_{i=1}^{N+N_{new}} \delta_{im}} \quad (3)$$

$$\kappa_m = \frac{\sum_{i=1}^{N+N_{new}} \delta_{im}}{\sum_{i=1}^{N+N_{new}} \sum_{m=1}^{M} \delta_{im}} \quad (4).$$

Come si può vedere, le formule sono state scritte con una notazione di tipo informatico, direttamente implementabile su un calcolatore. Per quanto riguarda i vari termini, abbiamo indicato con δ_{im} i campioni delle M densità di probabilità Gaussiane (pesate), con κ_m i loro coefficienti, con μ_m i valori medi e con Ω_m le matrici di covarianza. Queste ultime sono calcolate come:

$$\Omega_m = \frac{\sum_{i=1}^{N+N_{new}} \delta_{im} \left(\xi_k^{(i)} - \mu_m\right)\left(\xi_k^{(i)} - \mu_m\right)^T}{\sum_{i=1}^{N+N_{new}} \delta_{im}} \quad (5).$$

In alternativa è possibile scegliere una matrice di covarianza comune per tutte le Gaussiane, nel qual caso si parla di versione *omoscedastica* dell'algoritmo:

$$\Omega = \frac{\sum_{i=1}^{N+N_{new}} \sum_{m=1}^{M} \delta_{im} \left(\xi_k^{(i)} - \mu_m\right)\left(\xi_k^{(i)} - \mu_m\right)^T}{\sum_{m=1}^{M} \sum_{i=1}^{N+N_{new}} \delta_{im}} \quad (5 \text{ bis}).$$

Effettuando questa seconda scelta si ottiene un algoritmo più stabile rispetto a quello con covarianze diverse (versione *eteroscedastica*). Per questo motivo, nelle implementazioni numeriche di cui parleremo nel capitolo successivo abbiamo utilizzato sempre un'unica matrice di covarianza data dall'espressione (5 bis).
La procedura viene eseguita scegliendo opportunamente i valori iniziali di μ_m, κ_m e Ω, e iterando (1)-(4) finché non si giunge a convergenza. Come valori iniziali della media μ_m alla scansione k vengono prese le misure Z_k e i picchi della mistura all'istante $k-1$.

La scelta dei valori di κ_m e Ω è invece effettuata in seguito ad una operazione di "tuning" svolta direttamente sull'applicazione numerica. Soffermiamoci un momento su un problema che si riscontra frequentemente nell'implementazione dell'algoritmo appena visto. I valori dei campioni δ_{im} sono spesso molto piccoli e possono eccedere la precisione macchina, creando problemi di underflow.

Inoltre, iterando i passi come richiesto, non si va che ad aggravare la situazione. Nelle esperienze compiute non siamo mai riusciti a raggiungere la terza iterazione senza incorrere in problemi di questo tipo. Una soluzione consiste nell'utilizzare le proprietà dei logaritmi come segue.
Si calcolino $\ln(\delta_{im})$ e $\max_{im}(\ln(\delta_{im}))$. Dopodiché si valuti

$$\hat{\delta}_{im} = \delta_{im} e^{-\max_{im}(\ln(\delta_{im}))}$$

e si utilizzi tale quantità al posto di δ_{im} nei calcoli successivi.
Si può osservare che questo è lecito poiché $e^{-\max_{im}(\ln(\delta_{im}))}$ è un valore costante e la sua aggiunta come fattore moltiplicativo sia al numeratore che al denominatore nelle frazioni in gioco non altera i risultati.
Con l'accorgimento descritto si ottiene un algoritmo ben funzionante.

Passiamo ora all'ultima fase del metodo Monte Carlo, vale a dire quella di *ricampionamento*. Dai vari passi svolti finora abbiamo ottenuto il numero dei target e le loro posizioni, tuttavia, poiché il numero di particelle aumenta ad ogni scansione, si rende necessario un intervento, onde evitare di eccedere le capacità di calcolo della macchina su cui l'algoritmo è implementato. Il ricampionamento consiste nella redistribuzione delle particelle che vengono propagate alla scansione successiva, in modo che le particelle più importanti siano più rappresentate e che il loro numero complessivo resti limitato entro un certo valore. Naturalmente questa operazione non deve variare la somma totale dei pesi, che alla scansione k sarà pari a:

$$T_{k|k} = \sum_{j=1}^{N+N_{new}} \omega_k^{(j)} .$$

Questo può essere fatto in molti modi diversi. In particolare noi abbiamo seguito due diversi approcci, che ora presenteremo.

Il primo metodo segue un procedimento Monte Carlo e si svolge nel modo seguente. Supponiamo che il numero di target alla k-esima scansione sia pari a $[T_{k|k}]$ e che per ogni target si vogliano allocare N_z particelle, in modo da ottenerne un numero complessivo $N_k = N_z[T_{k|k}]$. Sia inoltre $N_{k-1} = N_z[T_{k-1|k-1}]$ il numero di particelle presenti al termine della scansione $k-1$. Normalizziamo i pesi di queste N_{k-1} particelle dividendoli per la loro somma $T_{k-1|k-1}$ ed indichiamoli con $\hat{\omega}_{k-1}^{(1)},..\hat{\omega}_{k-1}^{(N_{k-1})}$. Costruiamo ora un vettore $v = (v_1,..v_{N_{k-1}})$, il cui generico elemento abbia forma:

$$v_i = \sum_{j=1}^{i} \hat{\omega}_{k-1}^{(j)}.$$

Possiamo notare che l'ultimo elemento di tale vettore è sempre uguale a 1. Generiamo una istanza di variabile aleatoria uniforme $h \in [0,1]$. Consideriamo il primo elemento di v più grande di h e chiamiamo i^* il suo indice. Ora generiamo una particella uguale a quella cui corrisponde l'indice i^*, cioè $\xi_{k-1}^{(i^*)}$. Ripetiamo gli ultimi passi per N_k volte in modo da estrarre le N_k particelle volute a partire dalle N_{k-1} originarie. Assegniamo infine pesi uguali a tutte le particelle, pari a $\omega_k^{(i)} = \frac{T_{k|k}}{N_k} = \frac{1}{N_z}$.

La bontà di questo metodo risiede nel fatto che non richiede un'alta complessità di calcolo e funziona bene anche con grandi quantità di particelle. In base alle prove sperimentali effettuate, inoltre, esso fornisce buoni risultati.

Il principale svantaggio consiste nel fatto che, non essendo un metodo deterministico, può portare alla propagazione di particelle mal distribuite, con conseguente possibilità di generare echi spuri, come risulta dalle simulazioni effettuate.

Siccome il numero di particelle utilizzato nelle prove sperimentali non è particolarmente elevato, e dunque il costo computazionale non risulta eccessivo, abbiamo sviluppato anche un metodo di ricampionamento deterministico, che in queste situazioni dovrebbe offrire precisioni superiori. Questo sistema è in realtà molto semplice e consiste nel ricercare le N_k particelle con pesi maggiori al termine della scansione $k-1$ e propagarle nella scansione successiva. In questo modo si ottiene la distribuzione ottimale di particelle all'istante k.
Il metodo descritto può essere implementato senza problemi al calcolatore, ma potrebbe risultare troppo dispendioso qualora il numero di particelle diventi molto grande. In questi casi si preferisce solitamente un approccio Monte Carlo come quello descritto sopra.
Nel seguente capitolo saranno presentate prove con entrambi i metodi e si lascerà spazio ad un breve confronto.

Capitolo 3

Risultati sperimentali

Presentiamo ora i risultati sperimentali ottenuti dalle prove effettuate. Dapprima illustreremo le varie simulazioni, con particolare riguardo ai relativi parametri numerici, dopodiché commenteremo i risultati e ne forniremo un'interpretazione.

3.1 Simulazioni

Le simulazioni effettuate mirano a esplorare in maniera il più completa possibile le capacità dell'algoritmo di filtraggio implementato e dare una stima delle sue prestazioni. Per fare questo mostreremo scenari diversi per estensione spaziale, numero di target presenti e dinamica dei target stessi.
Tutte le prove sono state effettuate implementando l'algoritmo in un codice *Matlab*.
In tutte le simulazioni effettuiamo 200 scansioni da 1 secondo ciascuna, per un tempo totale di osservazione pari a 200 secondi.
Il numero medio di falsi allarmi su tutta la regione di sorveglianza sia $\lambda_k = 2$ e la loro densità $c_k(z) = 1$.
La probabilità di rilevamento è $P_d = 1$, equivalente alla certezza.
Consideriamo una probabilità di sopravvivenza dei target $P_s = 0.99$ e una PHD di nascita di nuovi target $b_{k|k-1}(S)$ costante e pari a 10^{-7}.
Per tutti i target assumiamo che il modello di moto presenti un tempo di campionamento $T = 1\ [s]$ e una matrice di covarianza $Q(k) = 10^6 I_k\ [m^2]$, dove I_k è la matrice identità avente numero di righe pari al numero di target presenti alla scansione k-esima. Inoltre $v_{\max} = 500\, m/s$. I disturbi sulle misure sono caratterizzati da $\sigma_r = 60\, m$ e $\sigma_\theta = 0.5°$. In tutti i casi, salvo dove specificato diversamente, si è scelto $N_z = 30$.
Per quanto riguarda i restanti parametri, i valori sono stati scelti tramite un'operazione di "tuning" effettuata sulle prove sperimentali.

Facciamo notare che, per non ottenere numeri troppo piccoli che possano causare problemi di underflow, i valori numerici nell'algoritmo sono scalati di potenze di 10 e pertanto anche i coefficienti "tunati" risentono di questa operazione.

Abbiamo ottenuto buoni risultati inizializzando $\kappa_m = \frac{1}{M}$ e Ω a una matrice diagonale 4×4 di elementi $(2, 0.04, 2, 0.04)$ nell'algoritmo EM.

L'ultimo parametro che è necessario "tunare" è la funzione di verosimiglianza $f\left(z|\xi^{(i)}\right)$. Da questa funzione dipende buona parte della precisione del filtro, perché tramite essa le particelle vengono distribuite attorno alle misure.

Nella nostra implementazione, la funzione di verosimiglianza è costituita da una densità di probabilità Gaussiana a media z.

Per regolare $f\left(z|\xi^{(i)}\right)$ abbiamo agito sulla matrice di covarianza.

Detta K tale matrice, abbiamo ipotizzato che fosse esprimibile nella forma $K = \alpha I$, dove I è la matrice identità 4×4 e α un opportuno coefficiente.

Iniziamo a presentare le prove sperimentali a partire da quelle che ci hanno permesso di impostare i valori ottimali di tale coefficiente.

Consideriamo uno scenario con 2 target, senza fenomeni di nascita o scomparsa degli stessi. I target hanno coordinate iniziali sul piano x-y $(x_1, y_1) = (80, 80)\ [km]$ e $(x_2, y_2) = (80, 20)\ [km]$ e velocità costanti $(v_{x1}, v_{y1}) = (350, -50)\ [m/s]$ e $(v_{x2}, v_{y2}) = (350, 50)\ [m/s]$.

Per il ricampionamento utilizziamo il metodo Monte Carlo.

Le traiettorie reali dei 2 target sono indicate con i colori blu e verde, mentre le particelle del filtro PHD hanno colore rosso.

Consideriamo tre diversi valori di α, rispettivamente 10^{-4}, 10^{-5} e 10^{-6}:

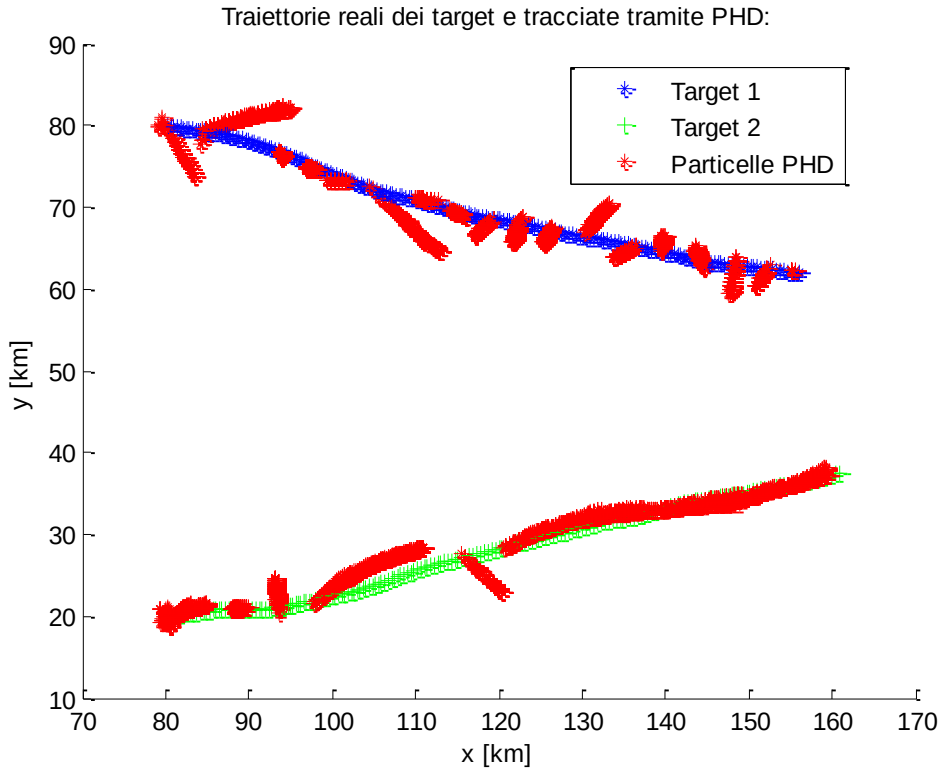

Figura 1 Simulazione nel caso di due target e $\alpha = 10^{-4}$.

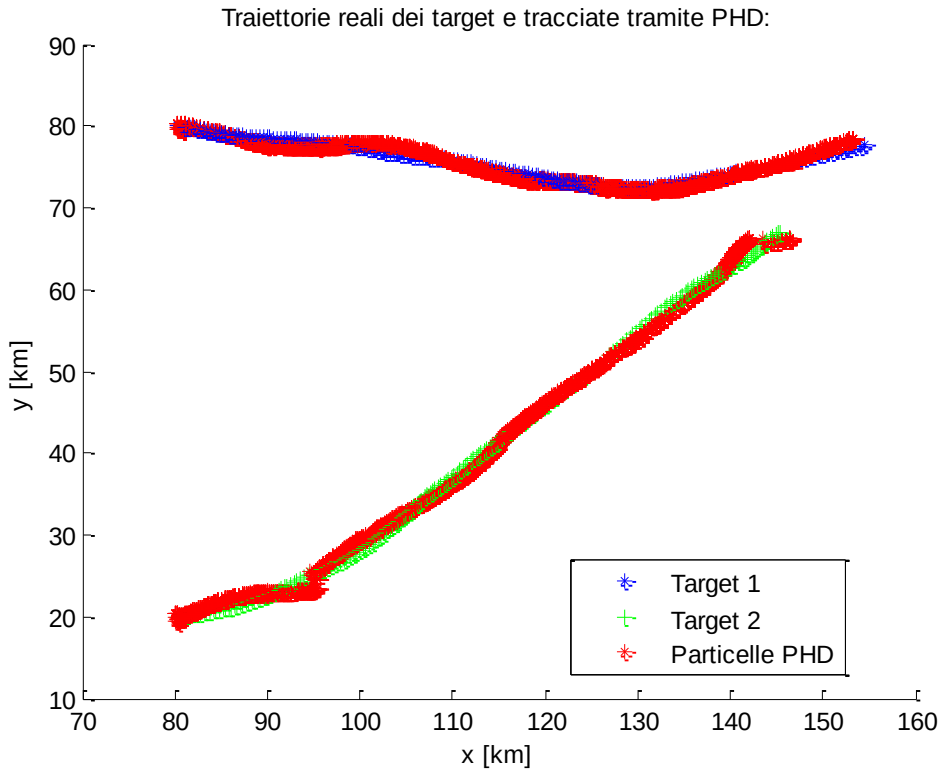

Figura 2 Simulazione nel caso di due target e $\alpha = 10^{-5}$.

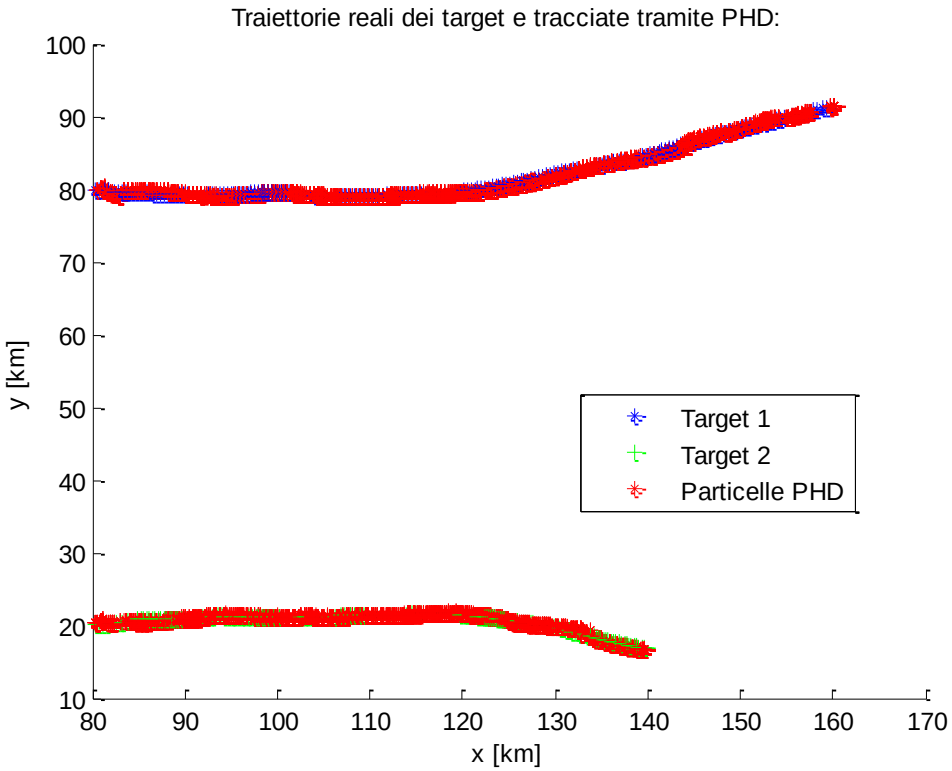

Figura 3 Simulazione nel caso di due target e $\alpha = 10^{-6}$.

Dai grafici mostrati si vede chiaramente la differenza tra le diverse scelte di $f\left(z\middle|\xi^{(i)}\right)$. Se la matrice di covarianza della funzione Gaussiana assume valori troppo elevati le particelle non vengono distribuite adeguatamente attorno alle misure e tendono a divergere (Fig.1). Abbassando il valore di α si possono invece ottenere dei risultati molto migliori. Naturalmente non si può scegliere α arbitrariamente piccolo, in quanto si rischierebbe di oltrepassare la precisione macchina. Dalle prove si è visto che questo comportamento risulta accentuato quando i target seguono traiettorie più irregolari. Il compromesso migliore che abbiamo trovato per il valore della matrice di covarianza è $\alpha = 10^{-6}$. Nel seguito adotteremo sempre questo valore.

Passiamo ora a un confronto tra l'algoritmo di ricampionamento che utilizza il metodo Monte Carlo e quello deterministico.
Utilizziamo lo stesso scenario iniziale delle simulazioni precedenti, tuttavia modifichiamo la matrice di covarianza $Q(k)$ che compare nel modello del moto dei target alzandone il valore: $Q(k) = 5 \cdot 10^{7} I_k$.
In questo modo i target dovrebbero seguire delle traiettorie più irregolari.

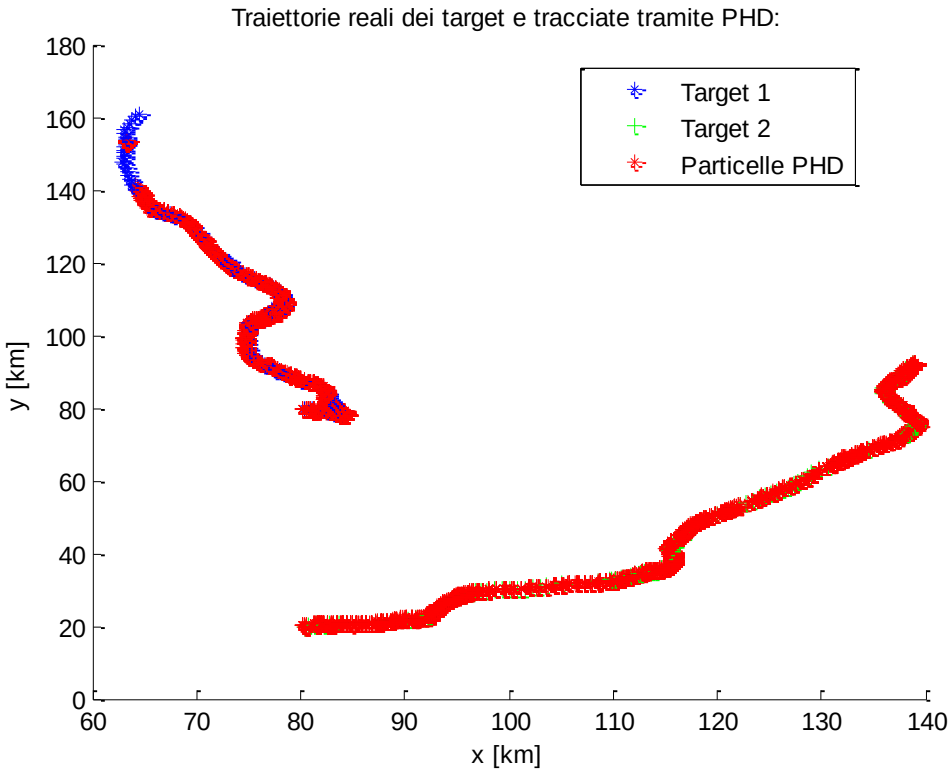

Figura 4 Simulazione con ricampionamento Monte Carlo.

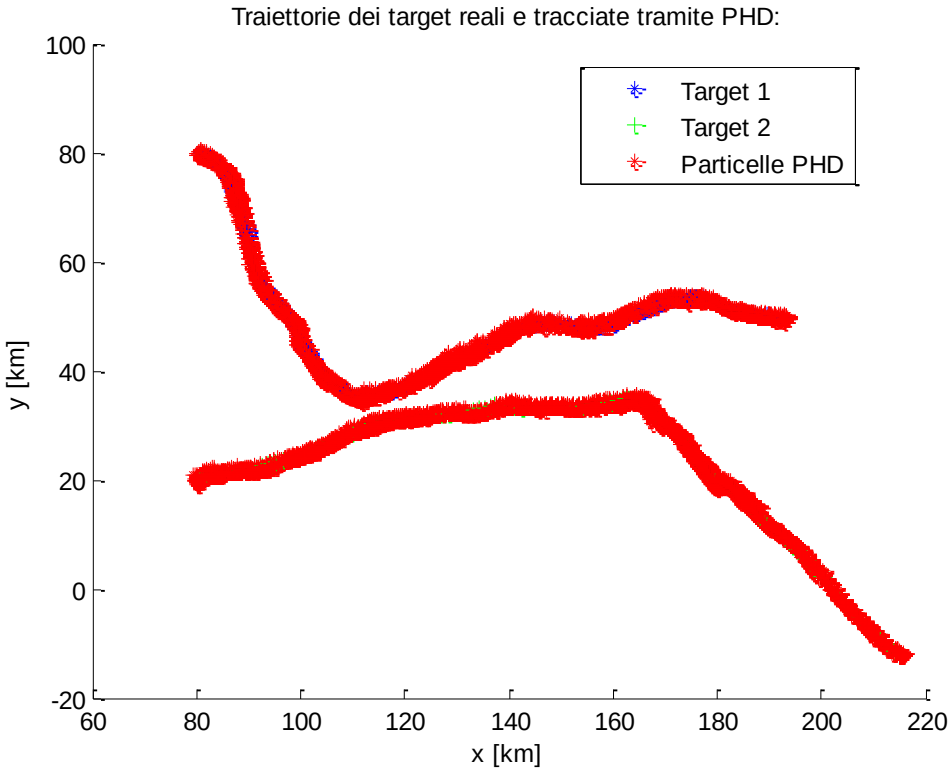

Figura 5 Simulazione con ricampionamento deterministico.

Notiamo innanzitutto che le traiettorie hanno effettivamente assunto una forma meno lineare. Questo può essere il caso di velivoli che stiano compiendo delle manovre non troppo spinte.
Il filtro che adotta il ricampionamento con il metodo Monte Carlo rischia di perdere la traccia. In Fig. 4 per esempio il target 2 viene risolto senza problemi, mentre alcune particelle si distribuiscono in maniera errata e fanno sì che dopo un certo istante il target 1 non venga più tracciato. Il filtro con il ricampionamento deterministico non sembra presentare questo problema e sembra più adatto a seguire velivoli ad alta dinamica.
Finora ci siamo limitati ad osservare situazioni statiche in cui il numero di target non varia nel tempo. Tuttavia, come abbiamo visto, il filtro che abbiamo implementato è stato sviluppato anche per situazioni più generali. È molto interessante studiare il caso in cui un nuovo target si aggiunga a quelli già presenti. In questo caso bisogna verificare che il filtro inizi a tracciare questo nuovo target da quando appare sulla scena e mantenga l'aggancio su tutti i target presenti.

Consideriamo ancora la stessa situazione di partenza, con la differenza che un terzo target compare dopo 100 $[s]$.
La posizione iniziale a cui compare il nuovo target ha coordinate $(160,160)$ $[km]$.
La velocità a cui si muove è $(-300,-100)$ $[m/s]$. Per questo caso impiegheremo unicamente il filtro con ricampionamento deterministico.

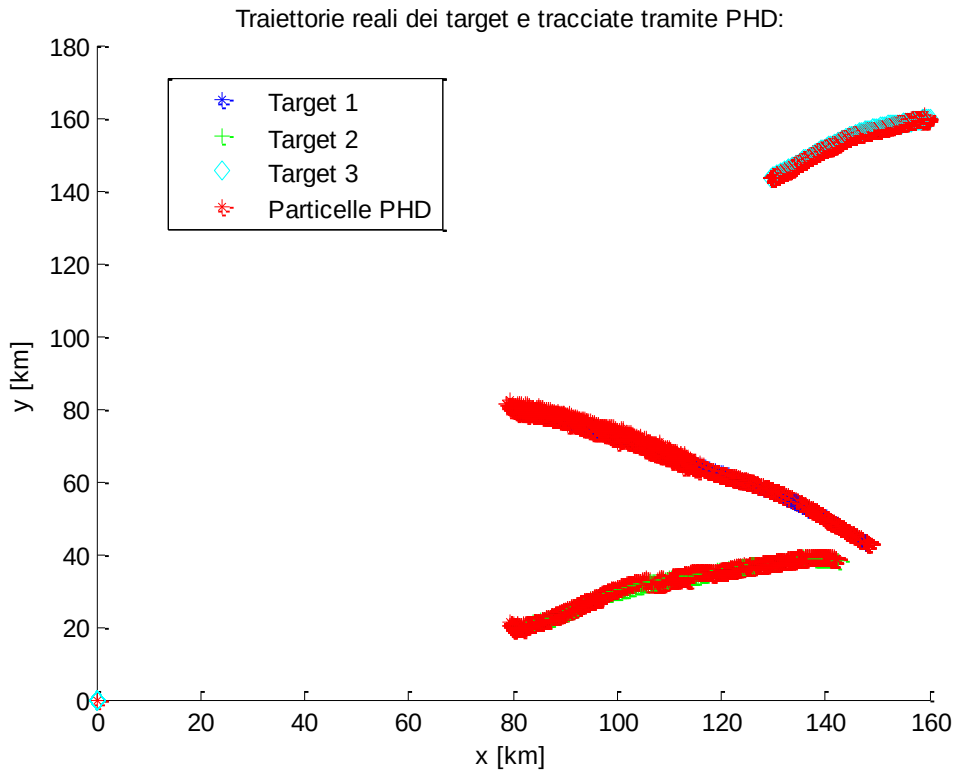

Figura 6 Simulazione con nascita di un nuovo target.

Si può vedere che il nuovo target viene agganciato immediatamente dall'algoritmo e poi tracciato correttamente. Allo stesso modo anche gli altri due target vengono inseguiti con precisione per tutta la durata dell'osservazione.
Appurate le capacità dell'algoritmo di tracciare target multipli sia statici che dinamici in condizioni di manovra moderata, effettuiamo delle simulazioni con un elevato numero di target per testare le prestazioni sotto sforzo. Il numero totale di target è ora pari a 10. Simuliamo uno scenario realistico in cui compaiono target con caratteristiche molto diverse tra loro. In particolare possiamo individuare degli slow target, quali navi, veicoli, UAV e aerei in

volo lento, con numero di Mach $M < 0.3$, e dei fast target, come aerei in volo alto-subsonico o supersonico. Nei grafici indichiamo i target lenti in colore verde e quelli veloci in blu.
Effettuiamo una prova con ciascuno dei due metodi di ricampionamento.

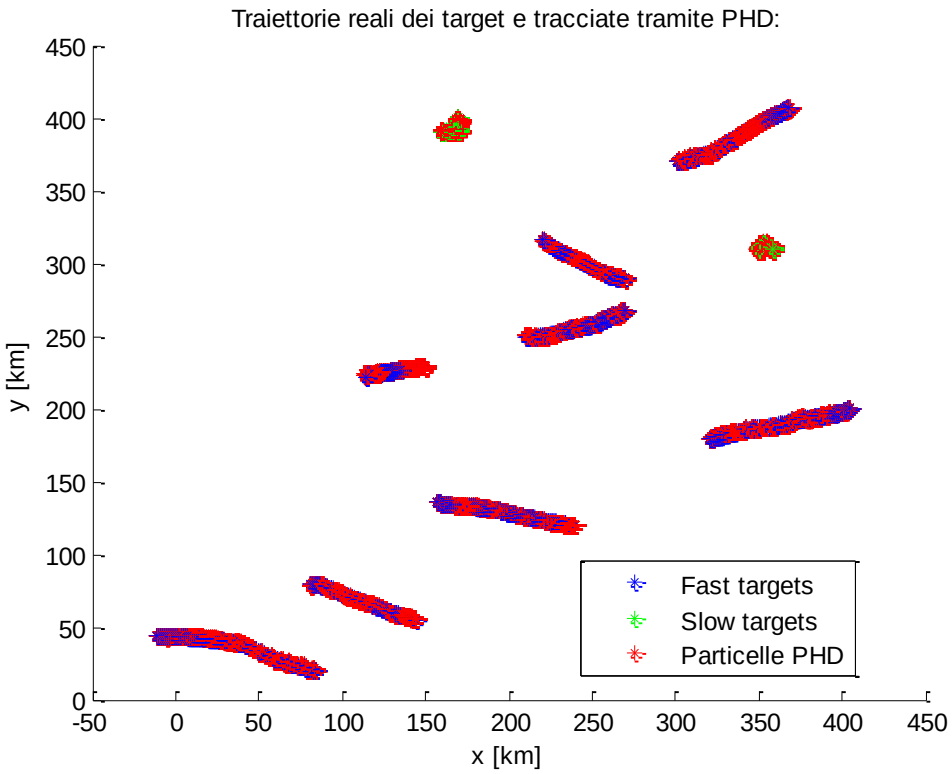

Figura 7 Simulazione con 10 target e ricampionamento con metodo Monte Carlo.

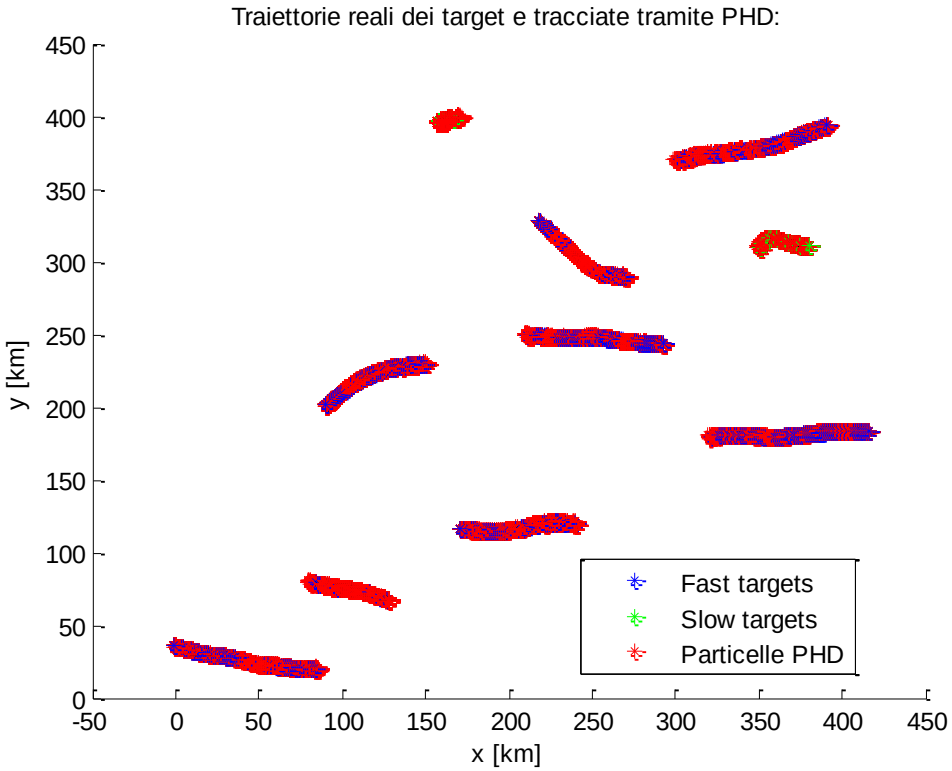

Figura 8 Simulazione con 10 target e ricampionamento con metodo deterministico.

Dalle Figure 7 e 8 di vede che entrambi i metodi di ricampionamento riescono a tracciare in modo corretto i vari target.
Per quanto riguarda i tempi di calcolo i due metodi risultano pressoché equivalenti e impiegano un tempo $260 < \tau < 280\ [s]$.
A titolo di esempio il tempo necessario per risolvere il problema con 3 target è $\tau \cong 19{,}5\ [s]$ e quello per il caso con 2 target raggiunge appena $\tau \cong 16{,}5\ [s]$.
Nel prossimo paragrafo cercheremo di inquadrare in maniera più precisa questo problema.

3.2 Interpretazione dei Risultati

Osservando i risultati delle prove sperimentali e commentandoli, abbiamo avuto modo di affrontare e chiarire alcuni punti chiave dell'algoritmo.

Innanzitutto ci siamo soffermati sull'importanza che una scelta opportuna dei coefficienti numerici riveste ai fini del conseguimento di un risultato corretto.
Come prevedibile, siamo giunti alla configurazione definitiva di parametri operando un compromesso tra varie esigenze progettuali.
In particolare, nel caso in esame abbiamo cercato di conciliare una buona accuratezza del tracking con una versatilità sufficiente da consentire l'impiego dell'algoritmo anche per l'inseguimento di traiettorie poco regolari.
Per quanto riguarda la precisione della stima, possiamo darne una misura tramite gli scarti medi σ tra posizione reale e stimata.
Ricavando tali scarti dalle prove sperimentali, si osservano valori $100 < \sigma < 500\ [m]$.
L'algoritmo si è dimostrato capace di tracciare dinamicamente i target e di superare i problemi legati alla loro nascita o scomparsa.
Una questione molto importante rimane tuttavia ancora aperta.
Abbiamo visto che il filtro è in grado di seguire numerosi target, a fronte tuttavia di un aumento del costo computazionale.
Un aumento del tempo di risoluzione si verifica anche aumentando il numero di particelle per target N_z.
Vogliamo quindi ottenere una relazione tra il numero di particelle impiegate e il tempo di elaborazione impiegato dall'algoritmo.
Per quantificare questa relazione riportiamo i risultati ottenuti da varie simulazioni in una tabella.
I valori del tempo, misurati in secondi, sono presentati divisi a seconda del tipo di ricampionamento effettuato.

N_z	Metodo Monte Carlo	Metodo deterministico
10	8.7196	8.7669
30	21.6802	21.7478
50	34.8143	34.4782
100	68.0710	67.5525
300	208.3780	204.4912
500	362.2202	360.7728

Tabella 1 Tempo di risoluzione in funzione del numero di particelle per target.

I valori sono molto simili con entrambi i metodi di ricampionamento.
Possiamo trovare un'espressione matematica della relazione descritta dai dati sperimentali utilizzando un'approssimazione ai minimi quadrati.

I valori risultano approssimati molto bene da una funzione lineare sia nel caso del ricampionamento con metodo Monte Carlo che con approccio deterministico.

Il tempo sarà dunque espresso dalle seguenti

$\tau = 0.7193 N_z - 1.365$ per il metodo Monte Carlo,

$\tau = 0.7142 N_z - 1.55$ per il metodo deterministico.

Riportiamo in Figura 9 il grafico del tempo di risoluzione in funzione del numero di particelle nell'approccio con metodo Monte Carlo:

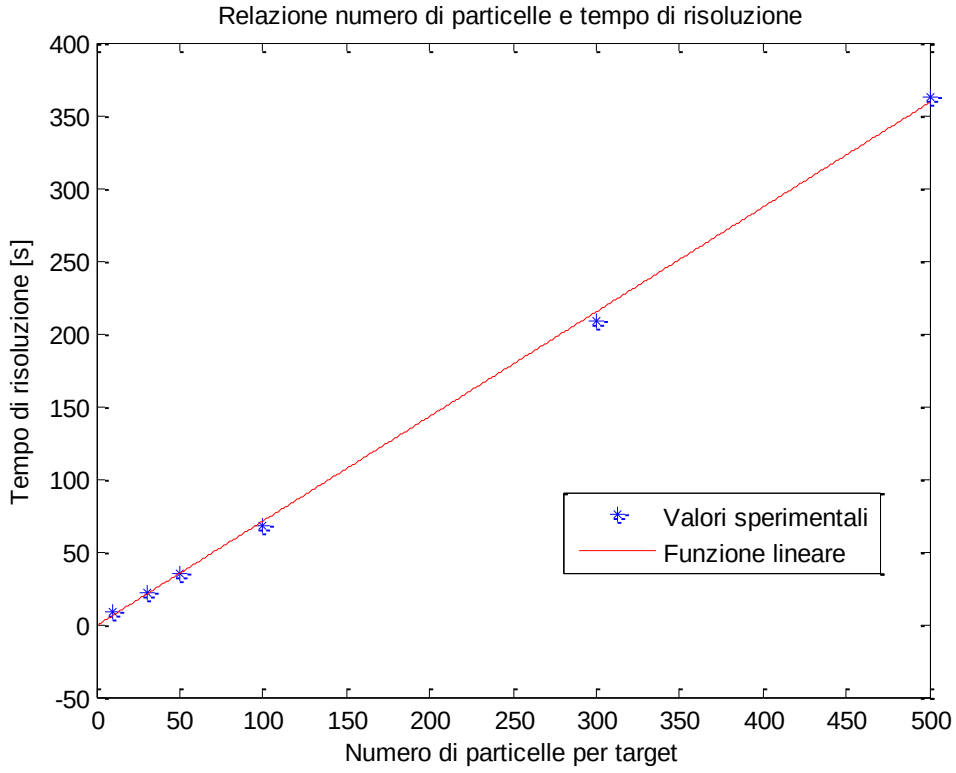

Figura 9 Espressione del tempo di risoluzione in funzione del numero di particelle.

Capitolo 4

Conclusioni

In questo elaborato è stato affrontato il problema del tracking multi-target facendo riferimento all'approccio di filtraggio PHD.
Siamo partiti introducendo la matematica necessaria alla trattazione, quindi abbiamo discusso la teoria dei filtri di Bayes.
Da questa teoria, difficilmente applicabile a livello pratico, è stato estrapolato il metodo PHD, che rappresenta una via percorribile al filtraggio multi-target. È stato poi descritto un algoritmo in grado di implementare il filtro PHD.
Tale algoritmo è stato implementato al calcolatore e, facendo uso di simulazioni numeriche, se ne sono valutate le prestazioni.
Per cercare di esplorare le sue capacità da più punti di vista, sono stati riprodotti svariati scenari, con numeri di target e dinamiche degli stessi diverse. Dell'algoritmo stesso si sono divise due versioni, differenti per il metodo di ricampionamento.
Queste sono state messe a confronto nel corso delle prove numeriche.
Il filtro ha mostrato generalmente ottime prestazioni, riuscendo a risolvere senza particolari problemi tutti i target per cui era stato progettato.
Poiché la validità del sistema è strettamente legata al modello che si utilizza per schematizzare i target e i sensori, sarebbe molto interessante approfondire ulteriormente questo aspetto. In particolare rivestirebbe grande interesse lo sviluppo di un modello non lineare.
Sarebbe infine di grande attrattiva l'implementazione su un supporto fisico e la verifica delle prestazioni in situazioni pratiche.

Appendice A

A.1 Cenni di derivazione su un insieme

In questa sezione descriviamo brevemente i termini matematici che compaiono nella trattazione dei Random Finite Sets.
Dato uno spazio compatto Q e la famiglia dei suoi sottoinsiemi finiti $F(Q)$, indichiamo con $B(F)$la famiglia dei sottoinsiemi di Borel di $F(Q)$.
Per una funzione $\mathrm{E}: B(Q) \to [0,\infty]$ definiamo densità η di E la funzione $\eta: F(Q) \to [0,\infty]$ tale che:

$$\mathrm{E}(Q) = \int_Q \eta(X)\delta X = \sum_{i=0}^{\infty} \frac{1}{i!} \int_{Q^i} f(\{x_1,...x_i\})\lambda^i(dx_1...dx_i),$$

dove λ^i indica la misura di Lebesgue su Q^i.
La derivata di una funzione $\mathrm{E}: B(Q) \to [0,\infty]$ nel punto x è un'applicazione $(d\mathrm{E})_x : B(Q) \to [0,\infty]$ definita come:

$$(d\mathrm{E})_x(S) = \frac{\mathrm{E}(S \cup \Delta_x) - \mathrm{E}(S)}{\lambda(\Delta_x)}$$

con $S \subseteq Q$ e Δ_x intorno di x.

Bibliografia

[1] R. Mahler, “An Introduction to Multisource-Multitarget Statistics and Applications”, Lockheed Martin Technical Monograph. March 15, 2000.
[2] B. Vo, S. Singh and A. Doucet, “Sequential Monte Carlo Implementation of the PHD Filter for Multi-target Tracking”, *Proc. Int. Conf. on Information Fusion*, Cairns, Australia, July 2003.
[3] O. Erdinc, P. Willett and Y. Bar-Shalom, “Probability Hypothesis Density Filter for Multitarget Multisensor Tracking”, 7th International Conference on Information Fusion (FUSION), 2007.