UNIVERSITÀ DEGLI STUDI DI PADOVA

Facoltà di Ingegneria
Corso di Laurea Magistrale in Ingegneria dell'Automazione

Progettazione di Sistemi di Controllo a.a. 2009/2010

CREAZIONE DI UN GRAFO FISICO-LOGICO PER UNA RETE DI VIDEOCAMERE E NON SOLO

Riccardo Ghirardello
607377/IAM

Roberto Guiotto
601527/IAM

Fabio Paggiaro
603370/IAM

Padova, 15 Febbraio 2010

Sommario

Tenere attivamente sotto controllo molteplici schermi in una rete di videosorveglianza non è semplice. In particolare, nel caso in cui ci siano eventi degni di attenzione sarebbe utile un sistema che provveda autonomamente ad individuarli e a visualizzare nei monitor soltanto le aree dove sono localizzati e quelle limitrofe, in cui è più probabile che questi si spostino. Per fare questo, è necessario trovare la relazione tra i campi visivi delle telecamere. Tramite opportune traiettorie (che modellizzano andamenti reali che meglio coprono l'area videosorvegliata) si identificano le aree visuali e le probabilità di transitare da una all'altra. Si decide di affrontare il problema utilizzando una struttura a grafo, associato ad una catena di Markov e, successivamente, aggiornarlo includendo anche quei nodi, detti "nascosti", che non sono direttamente individuabili dai dati forniti dalle telecamere. Per questa parte è stata ideata dagli autori una procedura detta *split* che cerca, tramite la scissione di alcuni nodi del grafo, di individuare quelli "nascosti". Al fine di quantificare la bontà del modello è stato introdotto un indice di prestazione. Esso, nello specifico problema considerato, permette di stabilire se il modello aggiornato tramite tale metodo consente di fare predizione più o meno correttamente rispetto a quello iniziale. In entrambi gli scenari qui considerati risulta che le prestazioni migliorano.

1 Introduzione

Motivazioni

Si immagini un addetto alla videosorveglianza alle prese con una decina di monotor o più. Si immagini inoltre che non abbia sempre un collega al suo fianco e che ciò che sta controllando sia alquanto rilevante. Da diversi studi si è constatato che di fronte a un numero anche limitato di schermi l'attenzione cala significativamente già dopo breve tempo, rendendo l'operatore incapace di riconoscere eventuali eventi anomali che avvengono nell'ambiente sorvegliato. Per questo motivo invece di delegargli il compito di monitorare i video senza alcun particolare criterio, risulterebbe più produttivo se avesse a disposizione un sistema in grado di individuare questi eventi e di fornire sugli schermi sia i segnali video provenienti dalle videocamere che inquadrano l'area in cui si verificano, sia quelli che inquadrano aree adiacenti, in cui è più probabile che l'evento si sposti.

Il problema considerato consiste nell'analizzare la topologia dell'ambiente da monitorare, intesa come insieme di aree di visibilità individuate dall'intersezione dei campi ottici delle videocamere (abbr. VCC). In particolare, si vuole associare una probabilità di transizione che descriva la possibilità di un soggetto di spostarsi da una regione all'altra. La valutazione di tali aree e probabilità sarà effettuata sulla base dei dati provenienti dalle videocamere, nella fase di calibrazione del sistema. Durante questa fase, successiva all'installazione del sistema di videosorveglianza, si ipotizza di avere un *target* facilmente riconoscibile dalle videocamere tramite programmi di visione computazionale, che si muove all'interno dello spazio da monitorare. Lo scopo è fornire al sistema dati che permettano di identificare un primo modello che descriva le aree visuali e le relazioni fra esse, cioè le probabilità di transizione dall'una all'altra. Il sistema dovrà poi essere in grado di correggere in maniera autonoma il modello rendendolo il più possibile affine alla topologia dell'ambiente considerato. Il fine ultimo è quello di supportare un sistema di predizione intelligente. Un modo per verificare se le soluzioni proposte sono valide o meno è proprio quello di valutare se consentono una predizione in qualche senso "più efficiente".

Problematiche

L'idea generale è quella di monitorare una regione spaziale estesa (ad esempio 1000x500m), con molte videocamere (anche centinaia) del tipo PTZ (pan - tilt - zoom). Si suppone di lavorare in un ambiente volumetrico interno (ad esempio un *terminal* aeroportuale), in modo che i loro coni di visibilità possano intersecarsi. E' proprio a causa di queste interserezioni che una descrizione accurata del modello risulta difficoltosa. Entrando più nel dettaglio di ciò che si andrà a fare, l'insieme dei coni visuale e della correlazione tra essi sarà rappresentato da una struttura a grafo, in cui i nodi rappresentano le varie partizioni dello spazio date da tali incroci e gli archi rappresentano le probabilità di transizione. Di fatto però, nella determinazione degli stati, è possibile che aree distinte viste dalle stesse VCC vengano interpretate come un singolo stato quando invece, nella realtà, ciascuna di esse dovrebbe essere descritta da uno stato diverso (ad esempio, se metà delle volte che ci si trova nello stato S_2 si proviene da S_1 e si va nello stato S_3, mentre l'altra metà si proviene da S_7 e si va nello stato S_8, allora si può ipotizzare che lo stato S_2 corrisponda, in realtà, a due partizioni distinte). In questo caso si parla di "stati nascosti" e il problema sarà realizzare un algoritmo che, per quanto possibile, cerchi di identificarli. Naturalmente, se ciò accade, il numero di stati aumenta, e così anche la complessità del modello.

Ci si aspettano inoltre problematiche a livello domputazionale notevoli, data l'elevata mole di dati da elaborare.

Stato dell'arte e contributo originale

Attualmente esiste molto in letteratura sul *tracking* mediante videocamere, cioè seguire un *target* mentre si sposta nello spazio e fornirne un'immagine in tempo reale. Nel conseguire questo, il sistema preposto allo scopo solitamente fa predizione sulla sua posizione e corregge la stima in base alle misure effettive. In una rete di VCC può poi facilmente capitare che il *target* esca dal cono di una telecamere ed entri nel cono di un'altra. In questo caso sarebbe utile sapere in anticipo quali telecamere allertare di volta in volta. Per fare ciò è necessario avere un modello che descrive la relazione esistente tra le varie aree visuali, che sia ad esempio generato come "*train*" di traiettorie possibili e/o probabili, per avere così un'idea di cosa sia un "evento anomalo".

Ciò che di originale viene proposto è la gestione dei sopra citati stati nascosti, servendosi di una certa quantità di informazione riguardo la traiettoria usata per generare il grafo, e un indice che consenta di valutare la prestazione di predizione col nuovo modello più accurato.

Risultati

Si sono considerati degli scenari ad hoc: uno in cui la traiettoria cerca di coprire la maggior area possibile e le telecamere hanno coni di visuale sovrapposti, un altro in cui la traiettoria simula un corridoio e le telecamere lo inquadrano ortogonalmente. In entrambii casi si è visto che cercando e "isolando" gli stati nascosti, l'indice di prestazione migliora, segno che il modello risultante descrive la situazione (topologica e di traiettoria) più accuratamente di quello iniziale.

Descrizione delle sezioni

Nel capitolo 2 si formalizza il problema trattato, introducendo le ipotesi pratiche introdotte, gli obiettivi sostanziali, le catene di Markov nascoste e un grafo associato come

modello scelto. Nel 3 sono presentati gli algoritmi e gli strumenti utilizzati per le elaborazioni: i simulatori per la generazione di dati sintetici, l'algoritmo di Baum e la procedura di *split* inventata dagli autori, nonchè l'indice di prestazione. Il capitolo 4 riporta i risultati delle simulazioni nei due casi considerati e valutazioni a riguardo. Infine nel 5 si hanno le conclusioni generali e proposte per sviluppi futuri.

2 Formalizzazione del problema

2.1 Ipotesi e assunzioni

Poichè, in principio, risulta impensabile gestire una rete composta da un centinaio di videocamere, è stato deciso di studiare il problema in due dimensioni, restringendosi ad analizzare un'area rettangolare di 25x50m con $K = 11$ telecamere posizionate sul perimetro, rivolte verso l'interno e fisse, come mostrato in Figura 1. Il modello di telecamera

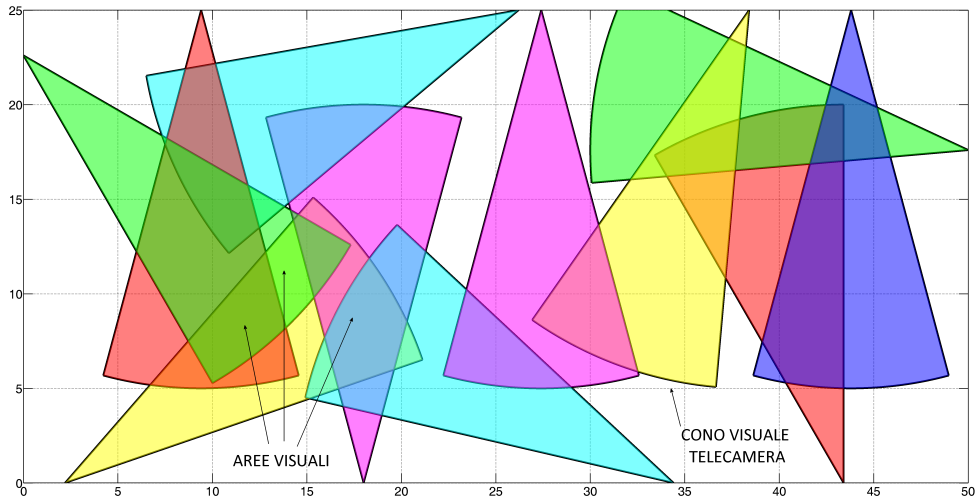

Figura 1: Schema di posizionamento scelto per le telecamere.

scelto, raffigurato nella successiva figura, possiede come parametri intrinseci raggio di visibilità pari a 20m ed angolo di campo di 30°, con capacità di catturare fotogrammi di 25frame/s.
Si suppone che tramite strumenti di visione computazionale si riesca a discriminare esattamente la presenza dell'oggetto (noto) che serve per la taratura del sistema. Delle VCC non sono note nè posizione nè orientamento, quindi neppure la forma delle partizioni dello spazio date dall'intersezione dei coni di visibilità.
I dati che le VCC forniscono sono treni di impulsi binari (a tempo discreto, ogni 1/25 di secondo): 1 quando la telecamera vede l'oggetto e 0 altrimenti. Sia T il numero di campioni considerati. Si considera un contesto di controllo centralizzato, il quale ad ogni istante ha a disposizione un vettore di K booleani (dominio $\{0,1\}^K$) rappresentante uno stato del sistema. Potenzialmente si possono avere 2^K stati S possibili, ma non tutti saranno verificati: alcuni perché quasi impossibili (esempio 1 1 ... 1, ossia tutte le VCC vedono contemporaneamente il *target*), altri perché non sono generabili dai dati raccolti (ad esempio se il *target* per la calibrazione non passa in una certa zona). Saranno quindi di numerosità

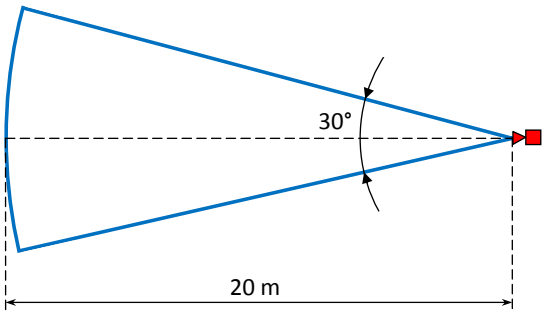

Figura 2: Parametri intrinseci ipotizzati per le videocamere.

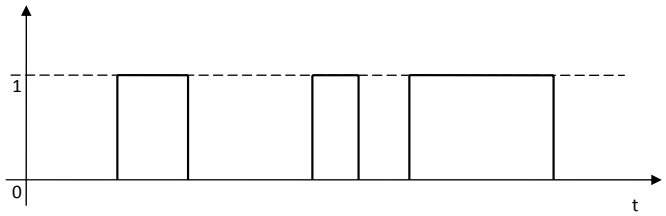

Figura 3: Tipologia dati in ingresso.

N, dove $N < 2^K$ e si potrà così definire $\mathcal{S} = \{S_1, \ldots, S_N\}$ come insieme degli stati. N inoltre è in modo plausibile legato a K da legge non esponenziale ma polinomiale, infatti in una rete di VCC il numero di coni che realmente si intersecano sarà di qualche unità per volta, al più una decina.

2.2 Obiettivi preposti

Fissata una configurazione di telecamere e scelta una traiettoria nello spazio considerato, il primo passo sarà costruire un grafo topologico direttamente dall'analisi dei dati. Si implementerà dunque, con l'ausilio delle catene di Markov (normali prima e nascoste poi) un algoritmo per determinare stati, archi ed etichette.

Il secondo passo sarà realizzare un algoritmo che individui gli stati nascosti ed aggiorni di conseguenza la struttura del grafo. A tal proposito si prenderà in considerazione l'algoritmo di Baum, un metodo iterativo che ri-stima i parametri del modello considerato.

A tal punto è importante avere uno strumento che misuri la bontà del modello finale, ossia quello comprensivo degli stati non più nascosti ma "rivelati", al fine di permettettere una descrizione migliore (in senso di predizione) della traiettoria usata per la costruzione di esso. Dette y le osservazioni effettive e $\hat{y}$ le loro stime, si è utilizzata la probabilità che la stima corrisponda all'osservazione effettiva, date le misure fino al passo precedente

$$\mathcal{P}[\hat{y}_{t+1} = y_{t+1} | y_{0:t}] \,. \tag{2.1}$$

Sono poi state sommare su t per avere un indice di qualità. Dato che però le misure sono disponibili tutte insieme, è possibile tener conto di questo e considerare al posto della precedente

$$\mathcal{P}[\hat{y}_{t+1} = y_{t+1} | y_{0:T-1}] \,. \tag{2.2}$$

2.3 *Hidden Markov Models*

Il modello matematico che si presta meglio ad essere utilizzato per descrivere il problema affrontato è quello delle catene di Markov nascoste (HMM: *Hidden Markov Models*) a stati finiti. La sostanziale differenza con le catene di Markov usuali è che non si conosce lo stato nell'istante di interesse ma si cerca di desumerlo dall'osservazione dei simboli emessi dal sistema, i quali si verificano con probabilità che dipendono non solo dal simbolo stesso, ma anche dallo stato in cui questo viene emesso. In realtà, nel problema trattato, la corrispondenza tra stati e osservazioni è molto stretta: dato che gli stati non sono altro che una codifica univoca dell'osservazione, si ha che ad ogni stato corrisponde sempre una sola osservazione e viceversa, quindi la matrice B definita poco più avanti è l'identità (il "viceversa" è però vero solo all'inizio della trattazione, si rimanda infatti al capitolo 3.3). Sembra dunque che la caratteristica fondamentale di un HMM venga meno; di fatto però si affronterà il problema che alcuni stati possono essere spezzati in due, e quindi, all'inizio, si ha che una di queste parti non è nota.

Si elencano di seguito gli elementi che caratterizzano un *Hidden Markov Model* (molti dei quali sono gli stessi delle catene di Markov semplici).

- N: numero di stati del modello (sebbene nascosti, di solito è comunque possibile dire qual è la loro numerosità). Essi sono estratti da un insieme $\mathcal{S} = \{S_1, S_2, \ldots, S_N\}$ e un singolo stato all'istante t è indicato con q_t.
- M: numero di osservazioni distinte che possono verificarsi. Esse sono estratte da un insieme $\mathcal{V} = \{v_1, v_2, \ldots, v_M\}$ e una sequenza è indicata col simbolo O.
- A: matrice delle probabilità di transizione, i cui elementi sono

$$a_{ij} = \mathcal{P}[q_{t+1} = S_j | q_t = S_i] \qquad 1 \leq i, j \leq N \tag{2.3}$$

  e hanno le proprietà $a_{ij} \geq 0$ e $\sum_{j=1}^{N} = 1$.
- B: matrice delle probabilità di emissione, i cui elementi sono

$$b_{ij} = \mathcal{P}[v_i \text{ al tempo } t | q_t = S_j] \qquad 1 \leq i \leq M\,, \quad 1 \leq j \leq N\,. \tag{2.4}$$

- π : distribuzione iniziale di probabilità di essere nei vari stati, di componenti

$$\pi_i = \mathcal{P}[q_1 = S_i] \qquad 1 \leq i \leq N\,. \tag{2.5}$$

2.4 Grafo delle transizioni

Spesso si è soliti interpretare una catena di Markov del primo ordine con un grafo orientato. In quest'ottica, l'evento $[q_t = S_i]$ rappresenta il fatto che la catena si trova nello stato S_i al tempo t, mentre l'evento $[q_t = S_j | q_{t-1} = S_i]$ è interpretato come la transizione della catena dallo stato S_i allo stato S_j. La costruzione del grafo avviene quindi seguendo le seguenti semplici istruzioni:

1. ad ogni stato della catena si associa un nodo nel grafo;
2. per ogni coppia di nodi nel grafo, l'arco orientato tra il nodo corrispondente all'evento $[q_{t-1} = S_i]$ e quello corrispondente all'evento $[q_t = S_j]$ viene etichettato con il valore di probabilità $a_{ij} = \mathcal{P}[q_{t+1} = S_j | q_t = S_i]$.

L'intero problema considerato può essere pertanto associato alla creazione di un grafo, in cui ogni nodo, come si è detto nell'introduzione, rappresenta una regione dello spazio, che per qualche motivo è da considerare distinta dalle altre. Essi sono determinati dall'analisi delle osservazioni, cioè dai dati provenienti dalle VCC, e sono raccolti in una opportuna struttura dati che è in grado di contenere le informazioni necessarie a descriverne il legame con gli altri. In particolare, per i fini voluti, è necessario tenere conto di quali sono gli stati "futuri" legati a uno stato "presente", cioè verso quali è possibile transitare da un certo nodo in analisi. Per avere una descrizione più completa dei dati, e quindi del comportamento del *target* in movimento, si è deciso di tener conto anche degli stati "passati" legati ad ogni nodo, nonchè dell'informazione di permanenza nello stesso stato.

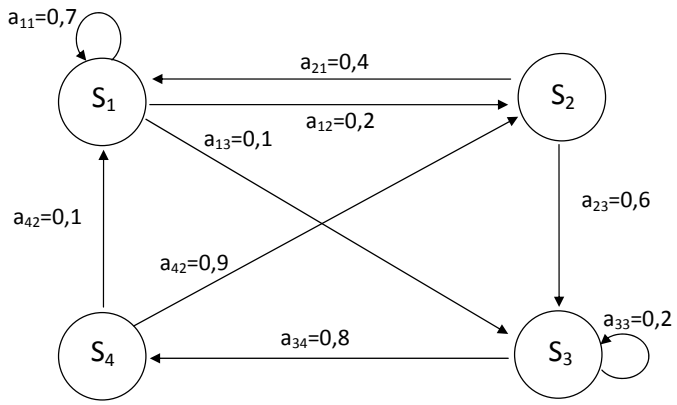

Figura 4: Esempio di un grafo descrittivo di una catena di Markov, si notino come gli archi uscenti da ciascun nodo sommi a 1.

Una volta creato il grafo, quindi la catena di Markov senza memoria (intendendo che per le transizioni non conta il passato ma solo lo stato presente) a partire dalle osservazioni, sarà eseguita un'ulteriore elaborazione considerando un po' di "storia" legata a ciascuno stato. In sostanza si guarda quali percorsi passato-presente-futuro lo interessano e si decide, secondo un qualche criterio da definirsi, se ha senso scindere tale stato in due sottostati per dare una descrizione più verosimile del comportamento dell'oggetto in movimento e della topologia della rete analizzata.

3 Algoritmi e strumenti utilizzati

3.1 Simulatori per la generazione dei dati

Non avendo dati reali da analizzare, è stato necessario implementare due simulatori allo scopo di generarne di sintetici. Essendo le condizioni di simulazione note, si è potuto anche valutare se i risultati ottenuti erano verosimili.

Un primo simulatore serve per la generazione delle traiettorie. Inizialmente si è scelto una spezzata che insegue dei punti random e cerca, allo stesso tempo, di coprire la maggior superficie di interesse possibile. Tale traiettoria è composta da punti che rappresentano gli istanti di campionamento legati ai frame delle VCC. La distanza tra essi è stata scelta ipotizzando che il target si muova a passo d'uomo (circa 4km/h, quindi circa 5cm tra ogni frame). Oltre al posizionemento delle VCC proposto in Figura 1, si è scelto una configurazione che rappresentasse una sorta di corridoio e, in corrispondenza, un'opportuna

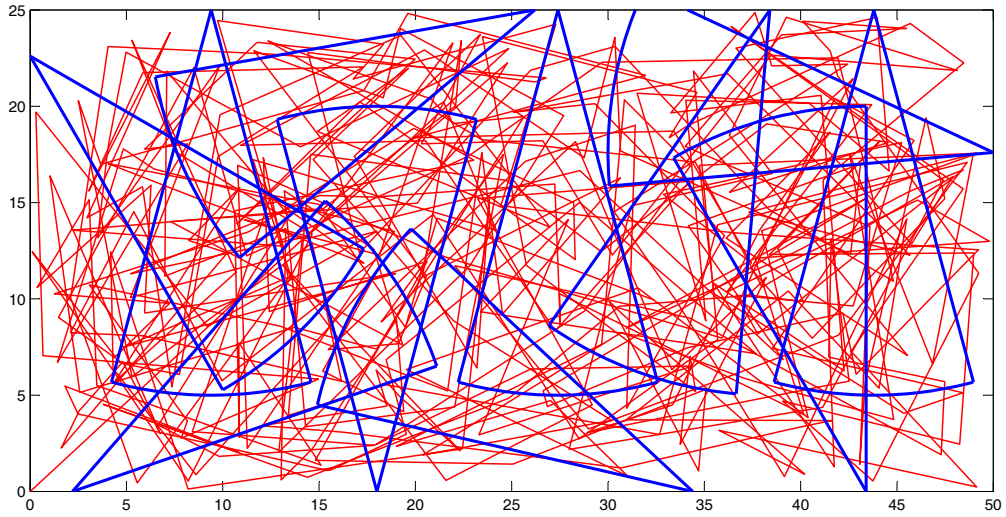

Figura 5: Esempio di traiettoria random utilizzata nelle simulazioni.

traiettoria che lo percorresse interamente da un capo all'altro. Le illustrazioni esplicative sono la 5 e la 6.

Il secondo simulatore riceve in ingresso la traiettoria generata dal primo e la collocazione delle telecamere ed è utilizzato per generare i dati descritti nelle ipotesi come "treni di impulsi". L'output è una tabella D di K colonne e T righe in cui l'elemento $d_{t,j}$ vale 1 se all'istante t la telecamera j vede il target e 0 altrimenti. Interpretata per righe, si ha un modo per assegnare un *tag* agli stati: basta semplicemente convertire la stringa di booleani in un numero intero.

Come prova della correttezza dall'implemetazione eseguita si sono associate le righe a dei colori, e con una simulazione con innumerevoli campioni si è prodotta l'immagine in Figura 7. Da notare come ricalchi la disposizione delle telecamere di Figura 1.

Globalmente, questi simulatori hanno permesso di procurarsi i dati necesari per la costruzione del primo modello di grafo.

3.2 Algoritmo di Baum

Prendendo spunto da Rabiner [1] è stato implementato un algoritmo iterativo che si rifà a una versione dell'algoritmo di Baum-Welch. Di fatto, uno dei problemi più difficili riguardanti gli HMM è quello di aggiustare i parametri del modello (A, B, π) in modo da massimizzare la probabilità della sequenza di osservazioni dato il modello. Non è ancora noto, però, un metodo per risolvere analiticamente questo problema, cioè non c'è garanzia di trovare i valori ottimi dei parametri. Tuttavia, con il metodo proposto si dovrebbe riuscire ad avere una massimizzazione almeno locale (non è infatti sempre garantita la convergenza entro i limiti desiderati).

Si definisce innanzitutto la variabile relativa alla passata in avanti come

$$\alpha_t(i) = \mathcal{P}[O_1, O_2, \ldots, O_t, q_t = S_i|\lambda] \tag{3.1}$$

che rappresenta la probabilità che si verifichi la parziale sequenza di osservazioni $O_1, O_2, \ldots, O_t$ (fino all'istante t) e di trovarsi nello stato S_i al tempo t, dato il modello λ. Essa può essere

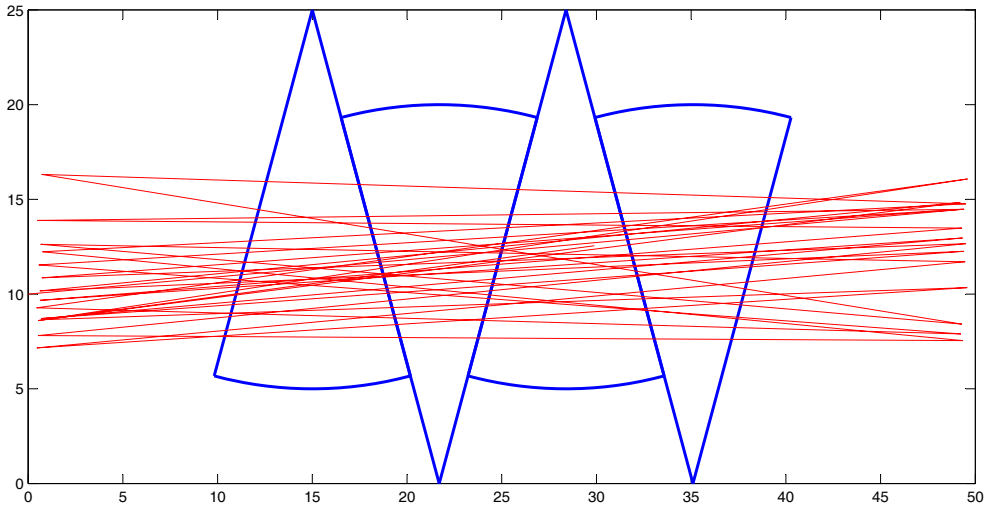

Figura 6: Esempio di traiettoria tipo corridoio utilizzata nelle simulazioni.

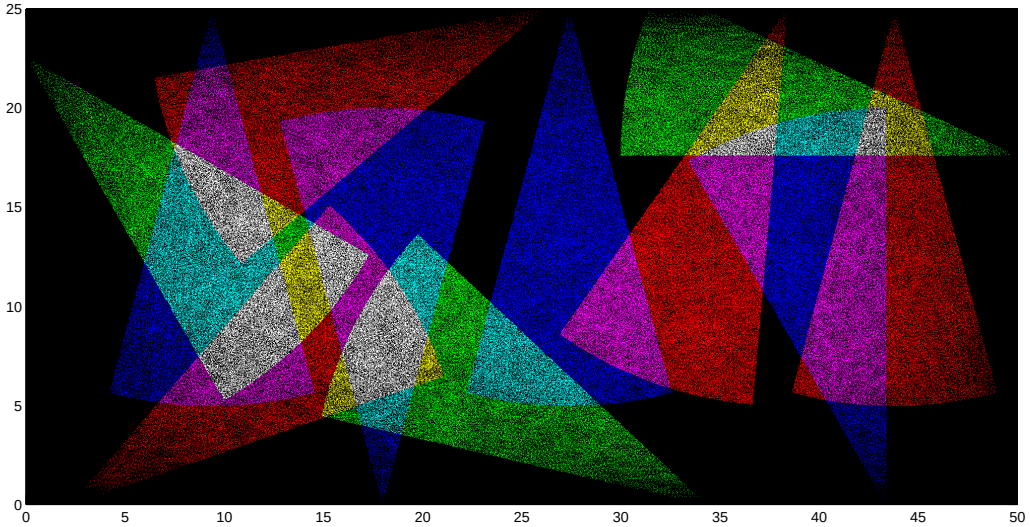

Figura 7: Prova che il simulatore che genera i dati funziona (i colori degli stati non sono tutti diversi per costruzione, ma hanno lo stesso colore quelli che si riferiscono alla stessa osservazione).

calcolata iterativamente secondo

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij}\right] b_j(O_{t+1}) \qquad 1 \le t \le T-1 \qquad 1 \le j \le N \tag{3.2}$$

e inizializzata con

$$\alpha_1(i) = \pi_i b_i(O_1) \qquad 1 \le i \le N\,. \tag{3.3}$$

Si ha in particolare che la probabilità dell'intera sequenza di osservazioni O è data da

$$\mathcal{P}[O|\lambda] = \sum_{i=1}^{N} \alpha_T(i)\,. \tag{3.4}$$

Analogamente si definisce la variabile relativa alla passata all'indietro come

$$\beta_t(i) = \mathcal{P}[O_{t+1}, O_{t+2}, \ldots, O_T | q_t = S_i, \lambda] \tag{3.5}$$

che rappresenta la probabilità che si verifichi la parziale sequenza di osservazioni dall'istante $t+1$ fino alla fine, dato di trovarsi nello stato S_i al tempo t e dato il modello λ. Anch'essa si calcola iterativamente, secondo

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(O_{t+1}) \beta_{t+1}(j) \qquad t = T-1, T-2, \ldots, 1 \qquad 1 \le i \le N \tag{3.6}$$

e inizializzata con

$$\beta_T(i) = 1 \qquad 1 \le i \le N\,. \tag{3.7}$$

Si definisce poi una variabile $\xi_t(i,j)$ che rappresenta la probabilità di trovarsi nello stato S_i al tempo t e nello stato S_j al tempo $t+1$, dato il modello e l'intera sequenza di osservazioni, ossia

$$\xi_t(i,j) = \mathcal{P}[q_t = S_i, q_{t+1} = S_j | O, \lambda]\,. \tag{3.8}$$

Per come sono definite le variabili α e β, risulta che

$$\xi_t(i,j) = \frac{\alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)}{\mathcal{P}[O|\lambda]}\,. \tag{3.9}$$

Sia infine $\gamma_t(i)$ la probabilità di trovarsi nello stato S_i al tempo t, date tutte le osservazioni e il modello:

$$\gamma_t(i) = \mathcal{P}[q_t = S_i | O, \lambda]\,. \tag{3.10}$$

Essa è in relazione con la precedente variabile secondo

$$\gamma_t(i) = \sum_{j=1}^{N} \xi_t(i,j)\,. \tag{3.11}$$

Se si sommano le γ sull'indice t si ottiene ua quantità che può essere interpretata come il numero atteso di volte che lo stato S_i sia visitato o, equivalentemnente, il numero atteso di transizioni che avvengono dallo stato S_i (escludendo dalla somma l'istante $t = T$).

Analogamente la somma delle ξ su t (tra 1 e $T-1$) può essere interpretata come il numero atteso di transizioni dallo stato S_i allo stato S_j. Quindi

$$\sum_{t=1}^{T-1} \gamma_t(i) = \text{numero atteso di transizioni da } S_i \tag{3.12a}$$

$$\sum_{t=1}^{T-1} \xi_t(i,j) = \text{numero atteso di transizioni da } S_i \text{ a } S_j\,. \tag{3.12b}$$

Si arriva così a considerare come ragionevole "ri-stima" dei parametri del modello le seguenti formule:

$$\begin{aligned} \bar{\pi}_i &= \text{numero atteso di volte nello stato } S_i \text{ al tempo } t=1 \\ &= \gamma_1(i) \end{aligned} \tag{3.13a}$$

$$\begin{aligned} \bar{a}_{ij} &= \frac{\text{numero atteso di transizioni da } S_i \text{ a } S_j}{\text{numero atteso di transizioni da } S_i} \\ &= \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)} \end{aligned} \tag{3.13b}$$

$$\begin{aligned} \bar{b}_j(k) &= \frac{\text{numero atteso di volte di essere nello stato } S_j \text{ e osservare } v_k}{\text{numero atteso di volte di essere nello stato } S_j} \\ &= \frac{\sum_{\text{t.c. } O_t = v_k}^{T-1} \gamma_t(j)}{\sum_{t=1}^{T-1} \gamma_t(i)} \end{aligned} \tag{3.13c}$$

E' stato dimostrato da Baum e colleghi che per il nuovo modello $\bar{\lambda} = (\bar{A}, \bar{B}, \bar{\pi})$ vale $\mathcal{P}[O|\bar{\lambda}] \geq \mathcal{P}[O|\lambda)]$ cioè con $\bar{\lambda}$ si ha una probabilità maggiore (o al più uguale) che la sequenza sia generata secondo esso piuttosto che secondo il modello iniziale λ.

Nella pratica questa procedura sarà ripetuta più volte usando $\bar{\lambda}$ al posto di λ fino ad ottenere una sorta di convergenza dei parametri. Va sottolineato che comunque questo è un algoritmo che tende a trovare un ottimo locale. E' importante quindi la scelta dei parametri iniziali: meglio questi descrivono la situazione corrente, più speranze si avrà di trovare il punto di ottimo globale.

Versione scalata dell'algoritmo

Dato che nello specifico del problema trattato la sequenza di osservazioni è molto lunga, le variabili α e β tendono ad assumere valori sempre più vicini allo 0 al crescere dei campioni, fino a che la precisione del calcolatore non è più sufficiente a rappresentarli non nulli. La soluzione più ragionevole è quella di eseguire un riscalamento delle variabili in modo da avere valori numerici in un *range* gestibile dal calcolatore. Riferendosi ad esempio ad $\alpha_t(i)$, di fatto si moltiplica per un fattore di scala che non dipende da i ma solo da t, e lo stesso si fa per $\beta_t(i)$; alla fine tali coefficienti si compenseranno perfettamente.

Si consideri ad esempio la formula (3.13b) scritta in modo esplicito in α e β:

$$\bar{a}_{ij} = \frac{\sum_{t=1}^{T-1} \alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)}{\sum_{t=1}^{T-1} \sum_{j=1}^{N} \alpha_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)} . \tag{3.14}$$

Per quanto riguarda $\alpha_t(j)$, si pone innanzitutto $\tilde{\alpha}_0(j) = \alpha_0(j)$, poi si definisce $c_0 = 1/\sum_{j=1}^{N} \tilde{\alpha}_0(j)$ e si ha la prima riscalatura $\hat{\alpha}_0(j) = c_0 \tilde{\alpha}_0(j)$. Per $t = 1, \ldots, T-1$ e $j = 1, \ldots, N$ si calcola

$$\tilde{\alpha}_t(j) = \sum_{i=1}^{N} \hat{\alpha}_{t-1}(i) a_{ij} b_j(O_t) \tag{3.15a}$$

$$c_t = \frac{1}{\sum_{j=1}^{N} \tilde{\alpha}_t(j)} . \tag{3.15b}$$

e si hanno i valori scalati tramite la

$$\hat{\alpha}_t(j) = c_t \tilde{\alpha}_t(j) . \tag{3.16}$$

Si osserva che $\hat{\alpha}_0(j) = c_0 \alpha_0(j)$ e per induzione

$$\hat{\alpha}_t(j) = c_0 c_1 \cdots c_t \alpha_t(j) = \left[\prod_{s=1}^{t} c_s \right] \alpha_t(j) . \tag{3.17}$$

L'aggiornamento è eseguito con la

$$\hat{\alpha}_t(j) = \frac{\sum_{i=1}^{N} \hat{\alpha}_{t-1}(i) a_{ij} b_j(O_t)}{\sum_{j=1}^{N} \sum_{i=1}^{N} \hat{\alpha}_{t-1}(i) a_{ij} b_j(O_t)} \tag{3.18}$$

e utilizzando la (3.17) si ha anche che

$$\hat{\alpha}_t(j) = \frac{\sum_{i=1}^{N} \alpha_{t-1}(i) \left[\prod_{s=1}^{t-1} c_s \right] a_{ij} b_j(O_t)}{\sum_{j=1}^{N} \sum_{i=1}^{N} \alpha_{t-1}(i) \left[\prod_{s=1}^{t-1} c_s \right] a_{ij} b_j(O_t)} = \frac{\alpha_t(j)}{\sum_{j=1}^{N} \alpha_t(j)} . \tag{3.19}$$

Per quanto riguarda $\beta_t(i)$ si ha analogamente che

$$\hat{\beta}_t(i) = c_t \beta_t(i) \tag{3.20}$$

con la particolarità che i fattori di scala sono gli stessi che per le α.

Con queste nuove variabili la (3.14) diventa

$$\bar{a}_{ij} = \frac{\sum_{t=1}^{T-1} \hat{\alpha}_t(i) a_{ij} b_j(O_{t+1}) \hat{\beta}_{t+1}(j)}{\sum_{t=1}^{T-1} \sum_{j=1}^{N} \hat{\alpha}_t(i) a_{ij} b_j(O_{t+1}) \hat{\beta}_{t+1}(j)} \tag{3.21}$$

e dato che le $\hat{\alpha}$ e le $\hat{\beta}$ si possono scrivere rispettivamente anche come

$$\hat{\alpha}_t(i) = \left[\prod_{s=1}^{t} c_s\right] \alpha_t(i) = C_t \alpha_t(i) \tag{3.22}$$

$$\hat{\beta}_{t+1}(j) = \left[\prod_{s=t+1}^{T} c_s\right] \beta_{t+1}(j) = D_{t+1} \beta_{t+1}(j) \tag{3.23}$$

segue che

$$\bar{a}_{ij} = \frac{\sum_{t=1}^{T-1} C_t \alpha_t(i) a_{ij} b_j(O_{t+1}) D_{t+1} \beta_{t+1}(j)}{\sum_{t=1}^{T-1} \sum_{j=1}^{N} C_t \alpha_t(i) a_{ij} b_j(O_{t+1}) D_{t+1} \beta_{t+1}(j)} . \tag{3.24}$$

Infine la quantità $C_t D_{t+1}$ risulta indipendente da t infatti

$$C_t D_{t+1} = \prod_{s=1}^{t} c_s \prod_{s=t+1}^{T} c_s = \prod_{s=1}^{T} c_s = C_T \tag{3.25}$$

e si può dunque elidere nella formula (3.24) ottenendo proprio la (3.13b).

Si è deciso di dare in ingresso al posto della matrice A, la stessa A con un leggero rumore additivo al fine di non avere dei balori nulli in nessuna posizione: in pratica, si è sommato una matrice NxN piena di $\delta = 10^{-6}$ e poi rinormalizzata per renderla ancora stocastica per righe.

Una possibile implementazione di questo algoritmo è suggerita in [2], mentre in [3] è proposta una soluzione che utilizza non fattori di scala ma logaritmi di probabilità, sempre con lo scopo di avere qualcosa di numericamente gestibile.

3.3 *Split* degli stati

Come accennato nell'introduzione, dai dati di ingresso è possibile solo dire da quali telecamere il target è visto nei vari istanti di campionamento ma non si ha una informazione precisa di locazione spaziale. In particolare, in due diversi istanti è possibile avere la stessa osservazione ma l'oggetto può appartenere a due aree di visione fisicamente distinte, ad esempio perchè il cono di visibilità di una telecamera è "tagliato" da quello di un'altra (vedi Figura 8). In casi come questo sarebbe oppurtuno avere due stati distinti che riferiscono alla stessa osservazione, perchè la traiettoria futura potrebbe essere anche sostanzialmente diversa, dato che sono diversi sia gli stati confinanti, sia le probabilità di transizione associate. Questa quindi è una prima motivazione per fare uno *split* (cioè scissione) di alcuni stati, e lo si può indicare come *split* di tipo "fisico".

Un altro caso che si può presentare è più legato alla traiettoria compiuta dal target.

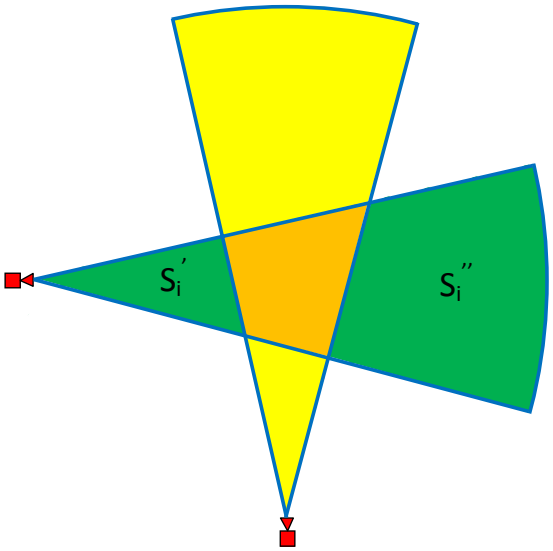

Figura 8: Schema per lo *split* fisico. Le aree evidenziate con lo stesso colore generano la stessa osservazione.

Si abbia ad esempio un corridoio, e questo sia percorso sempre interamente da un capo all'altro (senza tornare indietro dalla zona centrale per intendersi) come schematizzato in Figura 9 (in questa e in altre rappresentazioni che seguiranno, per semplicità non sono riportati gli archi degli autoanelli, ma vanno considerati sempre presenti, addirittura sono quelli con i valori più alti).

Qui le videocamere sono state volutamente poste in modo da scansionarlo ortogonalmente, quando invece potrebbe bastarne una che lo inquadri longitudinalmente. In genere il fatto di trovarsi nel cono di visione di una telecamera porta a identificare una probabiltà $1/2$ di spostarsi in quello di destra e $1/2$ in quello di sinistra, supponendo che non ci sia una direzione di percorrenza privilegiata. Se si introduce però l'informazione di quale è il capo del corridoio da cui si arriva, ad esempio quello di destra, allora per la prima ipotesi si può certamente dire che ci si sposterà verso il capo opposto, e quindi si ha probabilità 1 di andare nel cono di sinistra e 0 in quello di destra. Anche in questo caso ha senso fare una divisione di alcuni stati perchè il comportamento futuro è diverso data la posizione (osservazione) presente e la traiettoria passata. Questo si può indicare allora come *split* di tipo "logico". Si osserva inoltre che gli stati che possono essere scissi con questa motivazione spesso comprendono anche quelli del caso precedente.
In entrambi i casi segue un aumento del numero di stati e di archi del grafo, nonchè una opportuna sistemazione delle probabilità di transizione.

E' ovvio che, nota la distribuzione delle telecamere e di conseguenza delle regioni individuate, gli stati da scindere per motivo fisico siano riconoscibili abbastanza facilmente, ma pensare a un algoritmo che li individui autonomamente non è cosa semplice. Si è deciso di arricchire il grafo in modo che contenga, per ogni stato (all'inizio unico perchè corrispondente alle osservazioni), anche un minimo di informazione circa la storia passata e futura, legata alla traiettoria che si utilizza per la creazione dello stesso. In particolare, ignorando gli autoanelli, si memorizzano tutte le possibili combinazioni di stati passati-futuri che si verificano, nonchè il numero di volte che accadono, intesi come quegli stati legati al nodo in questione da cammini di lunghezza 1.

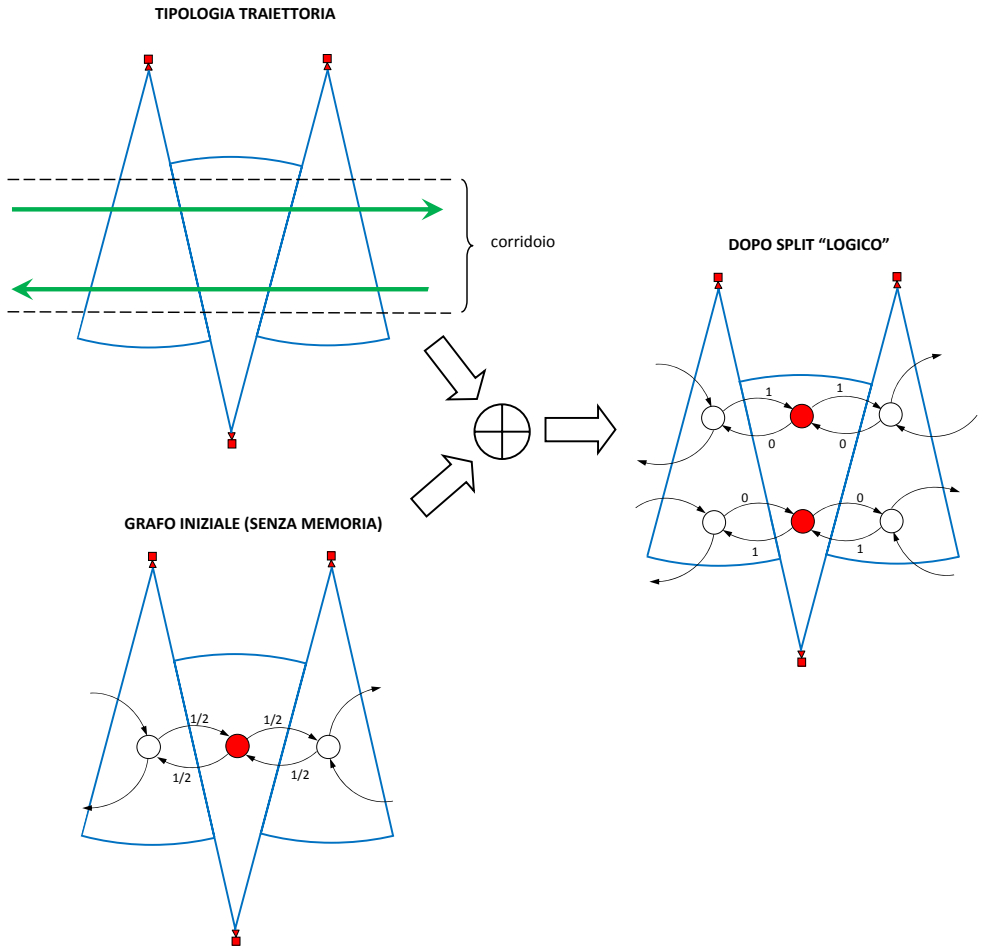

Figura 9: Schema per lo *split* logico.

Schematicamente, per un generico nodo S_i si può costruire una tabella che, letta per colonne, dia tali combinazioni di nodi passati S_{i-1} e futuri S_{i+1} legati a tale stato:

S_{i-1}	S_a	S_b	S_b	S_c	S_c
S_{i+1}	S_b	S_a	S_c	S_a	S_c

Corrispondentemente si costruisce un'altra tabella con il numero di volte che si verifica ogni coppia:

		S_{i-1}		
		S_a	S_b	S_c
S_{i+1}	S_a	0	2	4
	S_b	5	0	0
	S_c	0	3	1

In questo esempio si ha che lo stato futuro è S_b solo quando si proviene da S_a mentre gli stati futuri sono $\{S_a, S_c\}$ solo quando si proviene da $\{S_b, S_c\}$.

L'idea è stata quella di passare ad un'interpretazione di carattere di algebra lineare: nell'esempio si osserva che la prima colonna è ortogonale alle altre due e rappresenta la

parte in verde di Figura 10, mentre le colonne 2 e 3 sono invece "simili" tra loro e descrivono la parte in rosso. Si ottiene così che il nodo S_i può essere suddiviso in due sottostati separati S_i' e S_i''. Un altro modo per visualizzare la separazione è considerare i passati e i futuri dello stato: si individuano infatti dei sottoinsiemi ben precisi, come mostrato in Figura 11.

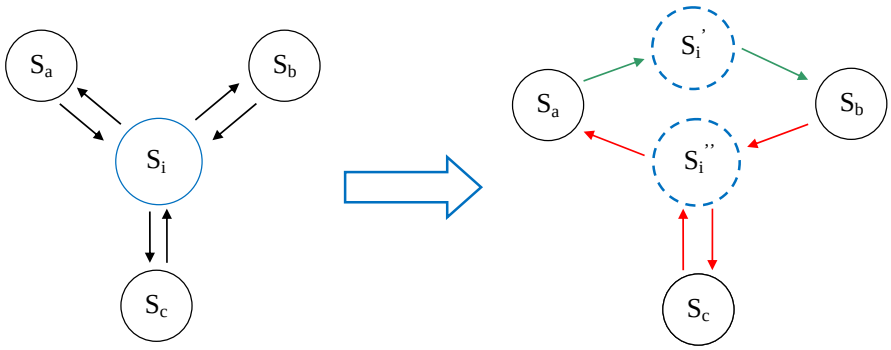

Figura 10: Rappresentazione riguardo l'esempio proposto.

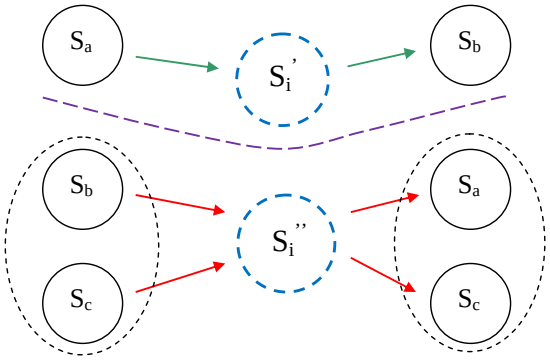

Figura 11: Rappresentazione riguardo l'esempio proposto.

Quel che si cerca è allora un indice che dia una "misura di ortogonalità" tra questi vettori colonna, e in particolare si è scelto, rifacendosi alla geometria, il coseno dell'angolo tra due vettori a e b definito come:

$$\sigma_{ab} = \frac{a^T \cdot b}{\|a\|\|b\|} \tag{3.26}$$

e che vale 0 quando sono otogonali, 1 quando sono paralleli.
In una matrice (che risulta simmetrica) si raccolgono poi questi valori per ogni coppia di vettori, e si decide di fare uno *split* per il nodo in questione quando si trova uno 0 (questa è una condizione abbastanza forte ma che dà una certa sicurezza che l'operazione non sia fatta inutilmente; successivamente si sono fatte prove rilassando il valore di σ a 0.1 e 0.2). Significa infatti (ricordando che la tabella è letta per colonne) che i due stati del passato che determinano tale 0 producono un comportamento del tutto diverso. Detti S_{P_1} e S_{P_2} tali stati, si distribuiscono tutti gli altri passati del nodo S_i a seconda che siano più "paralleli" a

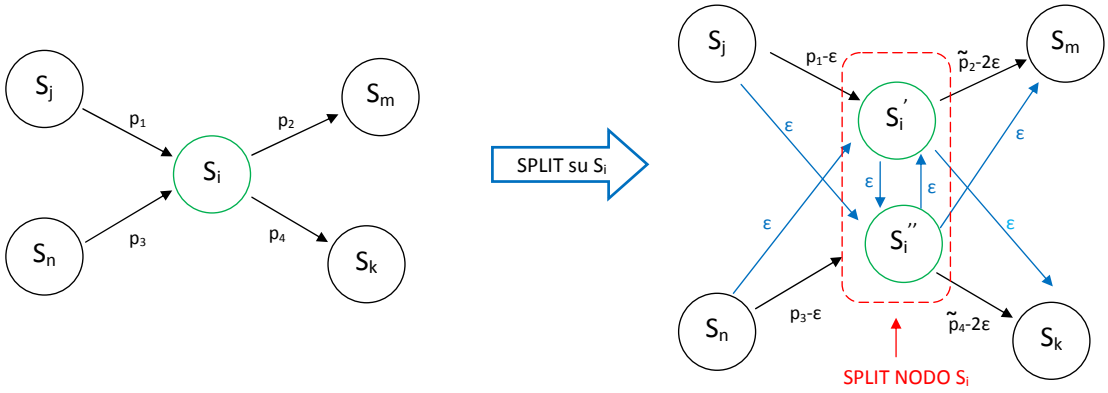

Figura 12: Schema che rappresenta l'operazione di *split* per un generico nodo S_i.

uno o all'altro sempre secondo l'indice (3.26). In questo modo si costruisce un sottoinsieme di passati associato al nuovo nodo S_i' e uno associato a S_i'', che contengono rispettivamente anche S_{P_1} e S_{P_2}. Ne risulta, inoltre, una naturale divisione in due sottoinsiemi anche dei futuri (come nel precedente esempio).

C'è poi il problema di assegnare le nuove probabilità di transizione. Si è deciso di lavorare non direttamente su di esse, bensì sul numero delle varie transizioni da uno stato all'altro, e spartirle sempre secondo gli indici trovati in precedenza. La matrice A viene poi ricalcolata sulla base di questi nuovi valori.
Riferendosi allo schema generale della procedura di *split* in Figura 12, si potrebbe immaginare, da quello che si è detto sopra, che non ci siano archi tra il sottoinsieme di passati legati a S_{P_1} e il nodo S_i'', perchè di fatto S_{P_1} transita verso S_i'. Si decide invece di porre una probabilità non nulla (ma bassa) per tali transisioni, sia perchè non si possono escludere deterministicamente, sia perchè in tale insieme possono esserci stati che non hanno "distanza" 0 da S_{P_2} e quindi non sono del tutto slegati da S_i''. Alla fine gli unici elementi di A che vengono modificati sono, com'è intuibile, quelli delle righe e colonne relativi ai due nuovi stati, in particolare in tale matrice S_i' prende il posto di S_i e si aggiunge una colonna a destra e una riga in basso per S_i'', lasciando gli altri stati inalterati.
Per la matrice B l'aggiornamento riguarda solo il numero di colonne, infatti le osservazioni (sulle righe) rimangono le stesse: la colonna relativa a S_i diventa quella di S_i' e se ne aggiunge una a destra, uguale a questa, per S_i''.

3.4 Metodo di soluzione proposto

L'operazione di *split* proposta nel paragrafo precedente non è, per cos dire, "autosufficiente": l'aggiornamento che viene fatto sulla matrice delle probabilità di transizione è, infatti, solo una stima grossolana dei valori, cioè non è detto sia la descrizione più giusta della nuova configurazione degli stati. La soluzione a questo problema è data dall'algoritmo di Baum, il quale, come spiegato in precedenza, fa una ri-stima dei parametri del modello in funzione delle osservazioni. L'idea è quella di applicarlo ripetutamente dopo ciascuno *split* in modo da far convergere i parametri entro una certa tolleranza. In particolare, si impone che il ciclo continui finchè nessun aggiustamento dei valori tra la A in ingresso e quella in uscita supera un ν fissato (inizialmente posto pari a 10^{-4} ma poi portato a 10^{-3} nella parte sperimentale per ridurre l'onerosità computazionale).
Tutto questo viene ripetuto finchè ci sono condizioni che suggeriscono di fare uno *split*, che

come si è detto, si ha quando un qualche σ assume il valore 0.
Il procedimento seguito dal programma principale implementato è ben descritto nel diagramma di flusso di Figura 13.

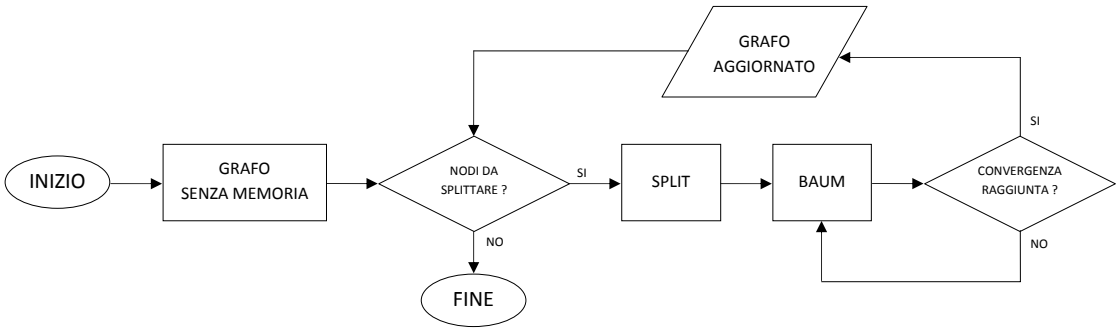

Figura 13: Diagramma di flusso rappresentante il procedimento seguito dal programma principale implementato.

Indice di prestazione

Come accennato negli obiettivi, si considera l'indice di prestazione

$$\eta = \frac{\sum_{t=0}^{T-2} \mathcal{P}[\hat{y}_{t+1} = y_{t+1} | y_{0:T-1}]}{T-1} \,. \tag{3.27}$$

Può essere interpretato come una sorta di rendimento: infatti vale al più 1, essendo una somma di probabilità divisa per il numero di elementi della sommatoria stessa. Questi sono $T-1$ perchè l'uscita all'istante $t=0$ non è stimata ma supposta nota.
Per calcolarlo, si supponga di generare le stime delle osservazioni $\hat{y}_{t+1}$ da uno stato x_t secondo la distribuzione data dalle probabilità degli archi uscenti da esso (in realtà la stima sarebbe $\hat{x}_{t+1}$ ma ricordando che ad ogni stato è associata sempre una sola osservazione, è come avere direttamente $\hat{y}_{t+1}$). Riferendosi all'esempio di Figura 14 (in cui le osservazioni sono indicate con A, B e C) per un generico istante t, le probabilità composte tra stima e osservazione sono calcolate come prodotto delle relative probabilità di transizione (perchè i due eventi sono indipendenti) e sono riportate in Tabella 1. Di questi valori si sommano quelli che corrispondono alla corretta scelta della stima, cioè gli eventi AA, BB e CC, ottenendo

$$\eta(t+1) = 0.09 + 0.04 + 0.25 = 0.38 \,. \tag{3.28}$$

Si può dimostrare che questo valore è sempre compreso tra $1/N_x$ e 1, dove N_x è il numero di nodi futuri del nodo x: il primo caso corrisponde al fatto che tutti gli archi portano lo stesso valore (ed è come scegliere una transizione qualunque); il secondo si ha quando $N_x = 1$ cioè comunica con un unico nodo (con probabilità 1). Il precedente calcolo può anche risultare direttamente considerando la riga della matrice A corrispondente al nodo x. Detta questa $a_{x,:}$ si ha cioè

$$\eta(t+1) = \|a_{x,:}\|^2 \tag{3.29}$$

(la condizione migliore si ha con η elevato).

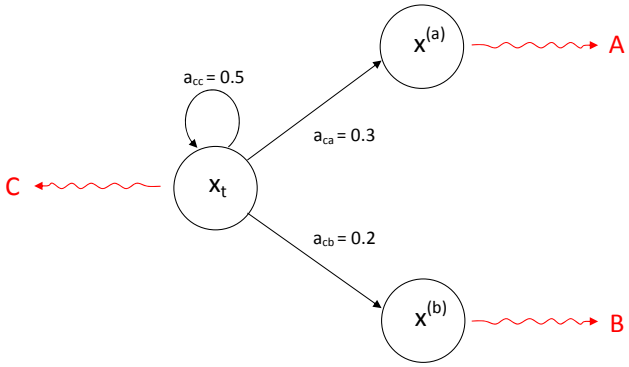

Figura 14: Esempio per spiegare l'indice di prestazione.

$\hat{y}_{t+1}$	y_{t+1}	prob. che si verifichi la combinazione
A	A	$0.3 \cdot 0.3 = 0.09$
A	B	$0.3 \cdot 0.2 = 0.06$
A	C	$0.3 \cdot 0.5 = 0.15$
B	A	$0.2 \cdot 0.3 = 0.06$
B	B	$0.2 \cdot 0.2 = 0.04$
B	C	$0.2 \cdot 0.5 = 0.10$
C	A	$0.5 \cdot 0.3 = 0.15$
C	B	$0.5 \cdot 0.2 = 0.10$
C	C	$0.5 \cdot 0.5 = 0.25$

Tabella 1: Dati di esempio per spiegare l'indice di prestazione.

La prova di "buona predizione" può essere fatta sulla stessa traiettoria di creazione del grafo oppure su una diversa. Per capire intanto se le intuizioni sono fondate ci si concentra sulla prima strada. Si ha motivo di credere che ad ogni passo di *split* l'indice di qualità aumenti perchè si ha un maggior numero di stati, i quali, per costruzione, descrivono più accuratamente il legame tra la traiettoria e il posizionamento delle telecamere. Al contrario, provando su una traiettoria diversa si pensa che l'indice aumenti fino a un certo punto ma poi peggiori, come insegna anche la teoria dell'identificazione dei modelli, perchè il grafo è stato modificato da *split* motivati dalla prima.

4 Parte simulativa

4.1 Caso corridoio

Questa sezione si riferisce alla situazione di Figura 6 dove, nell'algoritmo di *split*, è stato lasciato $\sigma = 0$ fisso.

La prima identificazione del modello ha definito 5 stati: 4 corrispondono alle altrattante VCC e 1 rappresenta la partizione in cui nessuna telecamera vede il target (sia detto stato uno). Dopo l'azione della procedura completa di *split*, il risultato è stato un grafo con 10 stati; ciò era auspicabile in quanto ognuno di questi stati iniziali aveva motivo di essere scisso. Allo stato uno corrisponde, infatti, un evidente *split* fisico mentre gli altri 4 sono

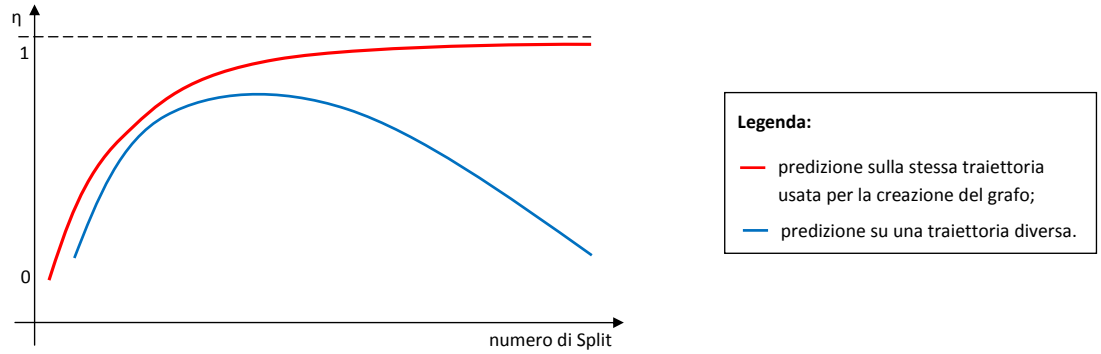

Figura 15: Ipotesi dell'andamento di η in funzione del numero di split.

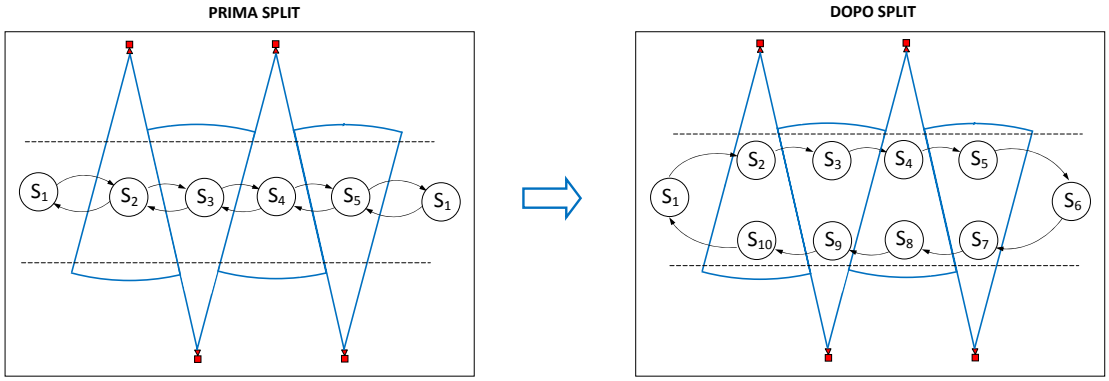

Figura 16: Rappresentazione che mostra come si spaccano i nodi.

interessati da *split* logico, in quanto i coni visuali sono, per costruzione, percorsi sempre interamente da destra a sinistra o viceversa.

L'algoritmo spezza gli stati nel seguente modo:

$$\begin{aligned} S_1 &\rightarrow S_1, S_6 \\ S_2 &\rightarrow S_2, S_{10} \\ S_3 &\rightarrow S_3, S_9 \\ S_4 &\rightarrow S_4, S_8 \\ S_5 &\rightarrow S_5, S_7 \end{aligned}$$

Si riporta in Tabella 2 la matrice A iniziale e in Tabella 3 quella finale entrambe approssimate a tre cifre decimali.

Dalla prima di queste matrici si nota come ogni stato comunichi, autoanello a parte, con altri due, in modo praticamente identico; dalla seconda, invece, ogni stato comunica con un unico stato. Le probabilità di autoanello presenti nella prima rimangono quasi le stesse anche dopo la procedura e sono duplicate, a seconda di come gli stati vengono scissi.

Per quanto riguarda l'indice di prestazione si ha l'andamento di Figura 17. Questo è come si sperava: mostra infatti un miglioramento al crescere del numero di stati *splittati*. Il modulo però ha una variazione molto bassa: ciò è dovuto alla presenza di una probabilità di

	S_1	S_2	S_3	S_4	S_5
S_1	0.950	0.025	0	0	0.025
S_2	0.074	0.851	0.074	0	0
S_3	0	0.075	0.851	0.075	0
S_4	0	0	0.073	0.853	0.073
S_5	0.077	0	0	0.077	0.847

Tabella 2: Matrice A iniziale.

	S_1	S_2	S_3	S_4	S_5	S_6	S_7	S_8	S_9	S_{10}
S_1	0.951	0.048	0	0	0	0	0	0	0	0
S_2	0	0.850	0.154	0	0	0	0	0	0	0
S_3	0	0	0.850	0.146	0	0	0	0	0	0
S_4	0	0	0	0.854	0.150	0	0	0	0	0
S_5	0	0	0	0	0.846	0.150	0	0	0	0
S_6	0	0	0	0	0	0.950	0.050	0	0	0
S_7	0	0	0	0	0	0	0.853	0.147	0	0
S_8	0	0	0	0	0	0	0	0.851	0.149	0
S_9	0	0	0	0	0	0	0	0	0.853	0.147
S_{10}	0.153	0	0	0	0	0	0	0	0	0.847

Tabella 3: Matrice A dopo la procedura di *split*.

autoanello significativamente alta. Così, la predizione è quasi certamente di rimanere nello stesso stato, e si modifica di poco se gli stati comunicanti sono uno o due. Si è pensato, dunque, ad una variante η_v dell'indice η che desse una misura di predizione corretta, supposto che certamente, al passo successivo, si esca dallo stato. In pratica, si considera separatamente lo studio del tempo di permanenza[1] e la giusta predizione quando capita una transizione verso uno stato diverso. Per calcolare questo nuovo indice si pongono nulli gli elementi della matrice A e la si rinormalizza per righe. Si procede, poi, come per η (risulterà ancora compreso tra 0 e 1). Così facendo, si ottiene il grafico di Figura 18: inizialmente la predizione è corretta nel 50% dei casi mentre dopo gli *split* è praticamente perfetta, difatti la curva si porta a 1.

4.2 Caso generale

Questa sezione riferisce alla situazione di Figura 5 dove, nell'algoritmo di *split*, al parametro σ sono stati assegnati diversi valori. La prima simulazione è stata svolta con $\sigma = 0$: è stato riscostrato come la procedura splittasse 6 stati, di cui la maggior parte riconducibili allo stato uno. Date che i nuovi nodi aggiunti al grafo sono considerati pochi, in relazione all'effettivo numero di partizioni visibili, è stata rilassata la condizione ponendo $\sigma = 0.1$ e poi $\sigma = 0.2$. Inoltre per evitare tempi di simulazione inaccettabili è stato necessario imporre $\nu = 10^{-3}$ per la convergenza dell'algoritmo di Baum.

Nelle figure 19 e 20 sono state riportate comparazioni di andamento degli indici η e η_v in funzione di σ.

[1] Più precisamente si intende la distribuzione di permanenza, cioè la probabità di continuare a rimanere nello stesso stato dato che ci si è rimasti per un tempo t_p. L'andamento di questa funzione è decrescente da 1 a 0 al crescere di $t_p \in [0, +\infty]$. Data la alte probabilità di autoanello dei casi considerati, la curva avrà inizialmente una parte costante a modulo circa unitario e poi una decrescita piuttosto rapida. In questo articolo, però, tutto ciò non è stato approfondito per motivi di tempo.

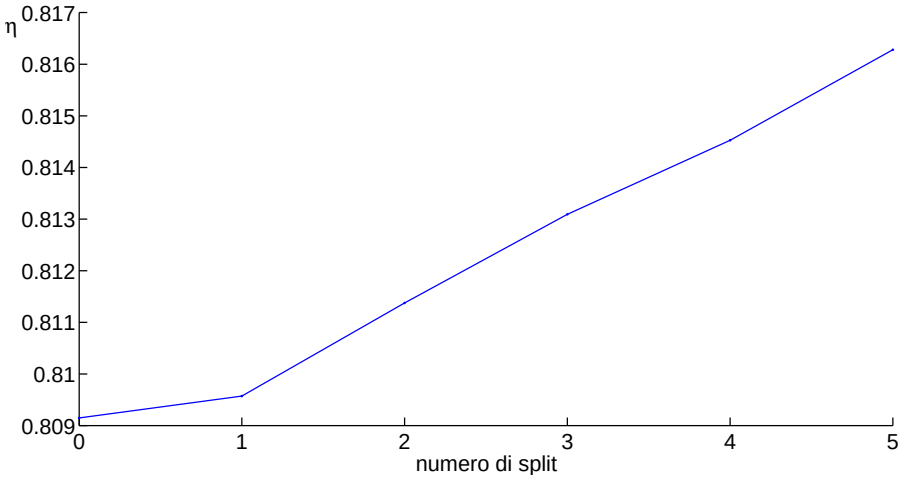

Figura 17: Indice di prestazione η per il caso corridoio.

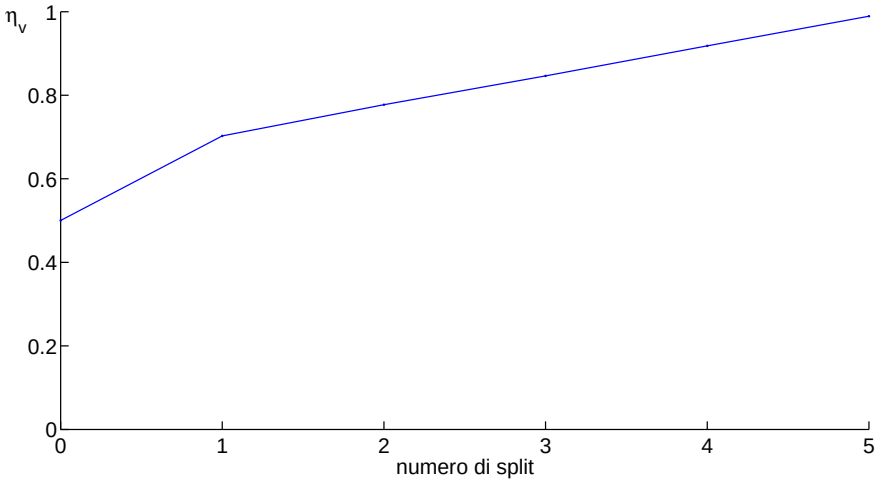

Figura 18: Indice di prestazione η_v per il caso corridoio.

Si nota come, al crescere di σ, il numero di *split* aumenta, così come entrambi gli indici di prestazione. Le curve non sono in generale monotone crescenti, come schematizzato dalla curva rossa di Figura 15, ma la tendenza è comunque quella di un miglioramento.

Altra osservazione è che a parità di numero di *split*, l'indice η cala al crescere di σ, questo perchè vengono eseguite scissioni "più approssimative". Non è altrettanto vero per η_v, e il motivo non è ben noto.

Rispetto al caso del corridoio, in cui η_v arrivava praticamente a 1, qui il valore nel migliore dei casi si porta a 0.48, ossia la predizione quando si ha un cambiamento di stato è giusta nel 50% circa delle volte. Stranamente però, i valori di η, che tengono conto indistintamente anche dell'autoanello, risultano maggiori. Ci si aspettava che il caso del corridoio, essendo costruito ad hoc, desse a questo proposito risultati più promettenti. La cosa più conveniente da fare, probabilmente, è considerare l'autoanello a parte, con quella che precedente è stata definita come distribuzione di permanenza.
Si tenga comunque conto che la scala dei grafici è diversa per i due casi. In particolare η_v

di questa sezione è stato riportato espandendo la scala, altrimenti il grafico sarebbe stato solo leggermente crescente attorno a 0.4.

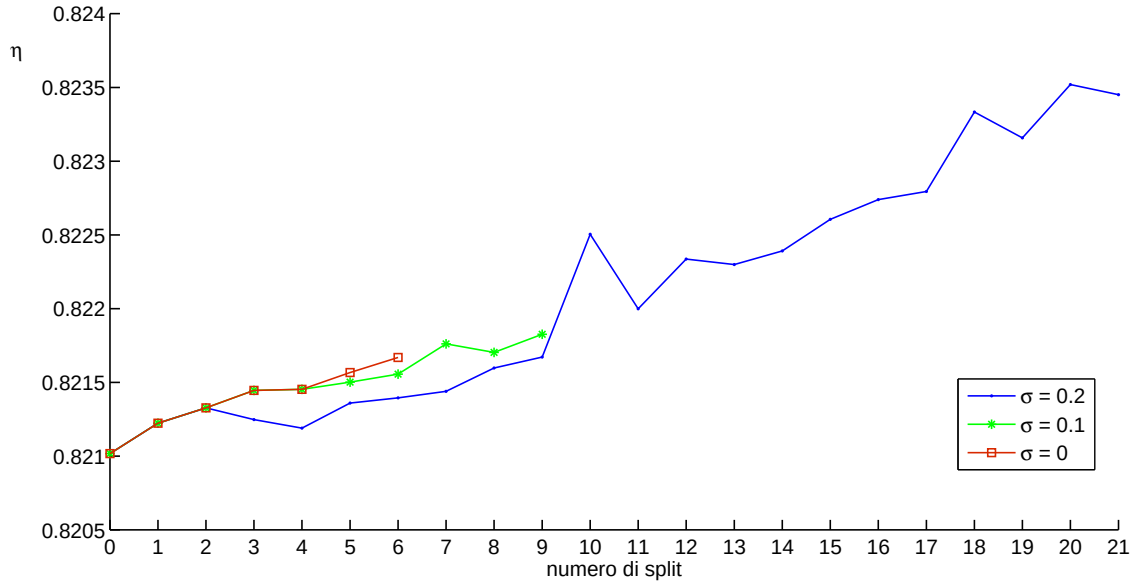

Figura 19: Indice di prestazione η per il caso generale al variare di σ.

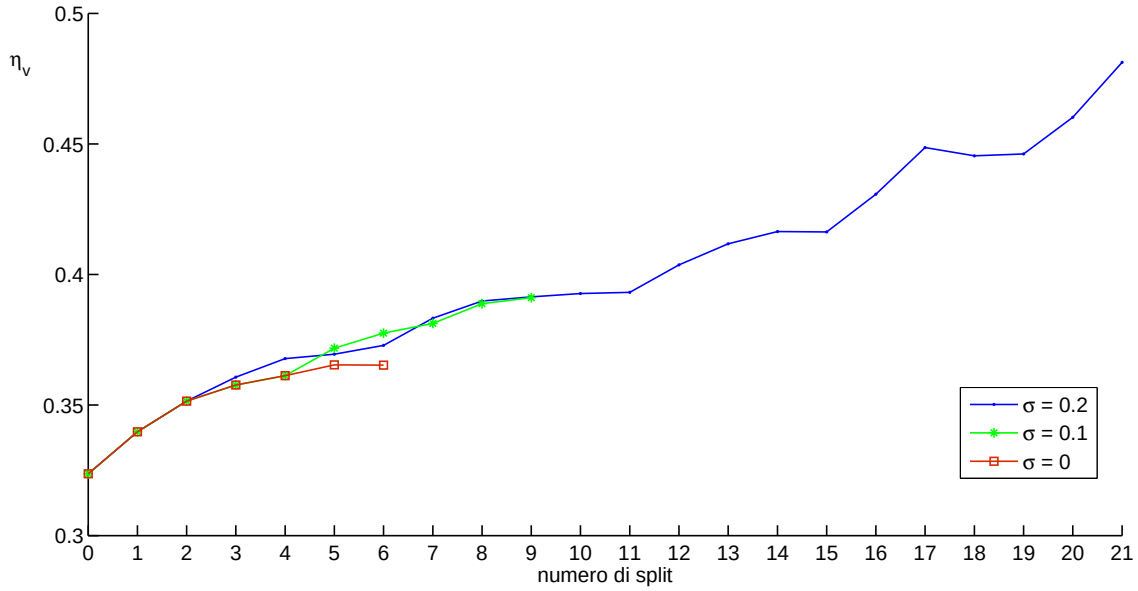

Figura 20: Indice di prestazione η_v per il caso generale al variare di σ.

5 Conclusioni, osservazioni e sviluppi futuri

Complessivamente si può essere soddisfatti dei risulatati conseguiti, almeno dal punto di vista qualitativo. L'indice di prestazione si è rivelato un buon modo per giudicare la

bontà degli algoritmi implementati, in prospettiva di fare predizione nel modo più corretto possibile. Purtroppo è mancato il tempo per un'analisi quantitativa e più approfondita su certi punti.

Per quanto riguarda la procedura di *split*, nell'implementazione, si sono considerate le combinazioni di stati passati e futuri legati al nodo in questione da cammini di lunghezza 1. Naturalmente un'informazine più completa, e quindi meglio approssimante il legame tra traiettoria e VCC, si ha considerando cammini di lunghezza 2 o 3. Così facendo però, a livello computazionale la cosa diviene troppo onerosa, almeno a livello centralizzato.
Ecco che uno sviluppo futuro molto interessante è quello di costruire il grafo fisico-logico delle relazioni tra le partizioni in modo distribuito, in cui cioè ogni telecamera ha a disposizione solo i dati prodotti di alcune altre. Un approccio di questo tipo è doveroso anche prospettando di gestire reti con molte più telecamere e su aree spaziali molto più estese, anche per meglio gestire lo scambio dei dati

Il grafo composto al primo passo, ossia prima della procedura di *split*, è costruito in modo da rappresentare una catena di Markov "senza memoria": infatti si sono prese in considerazione, per ogni stato, le transizioni verso, e non da, altri nodi. Per le operazioni di *split* invece, si tiene conto anche di passi precedenti e quindi sembra che si perda la relazione grafo-catena così chiara inizialmente. In realtà la cosa riguarda solo la costruzione del grafo, non la sua interpretazione: dopo che le probabilità di transizione sono state aggiornate, esso continua a definire una catena senza memoria, perchè conta solo qual è lo stato presente di permanenza e quelli futuri di transito.

Infine, la tipologia di dati considerati (telecamera che vede o non vede il target) è abbastanza generica: tali dati potrebbero anche essere prodotti da sensori di diverso tipo che rilevano la presenza del target, basta solo che i segnali siano riconducibili a impulsi binari di tipo "presente/assente". Anche le applicazioni possono essere molteplici, prima fra tutte l'estensione a tre dimensioni del problema studiato con telecamere PTZ, anche se in tal caso diventa più difficile stabilire la corrispondenza tra uno stato e una partizione dello spazio.

Riferimenti bibliografici

[1] Lawrence R. Rabiner, *A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition*, Proceedings of the IEEE, vol. 77, no. 2, febbraio 1989.

[2] Mark Stamp, *A Revealing Introduction to Hidden Markov Models*, 18 gennaio 2004.

[3] Tobias P. Mann, *Numerically Stable Hidden Markov Model*, 21 febbraio 2006.