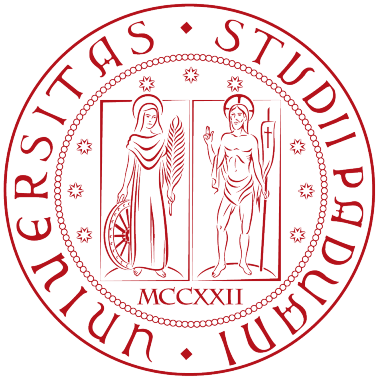
UNIVERSITA' DEGLI STUDI DI PADOVA



Facoltà di Ingegneria
Corso di Laurea in Ingegneria dell'Automazione

Multi-agent perimeter patrolling with asynchronous algorithms in discrete spaces

Gabriele Menegazzo 1041321 | Mattia Sessolo 1041152 | Elia Vidali 1033926

18 Febbraio 2013

Indice

Capitolo 1

Introduzione

In questi ultimi anni si è assistito ad un continuo aumento della domanda di sicurezza, infatti, basta osservare gli ambienti che ci circondano per rendersi conto dell'elevato numero di telecamere per videosorveglianza presenti nei luoghi che frequentiamo ogni giorno. Un'importante applicazione della videosorveglianza è quella della sorveglianza perimetrale (*perimeter patrolling*) ovvero un tipo di sorveglianza che consiste nel percorrere un perimetro o una parte di esso per mezzo di un insieme di agenti (che nel nostro caso sono videocamere) ad intervalli regolari per rilevare un possibile evento oppure più semplicemente per monitorare la situazione lungo il perimetro . Inizialmente, si utilizzavano delle videocamere fisse ovvero videocamere potevano inquadrare una sola area nel tempo; ai giorni nostri invece, si stanno sempre più affermando (o meglio, si sono affermate) le videocamere PTZ *(Pan Tilt Zoom cameras)* ovvero videocamere che possono spostare il loro campo visivo sui tre assi in base alle necessità come ad esempio seguire il movimento di una persona o di un oggetto oppure semplicemente per monitorare un'area; proprio per questo motivo le videocamere PTZ stanno ormai soppiantando le videocamere fisse in quanto potendo spostare il loro campo visivo riescono a monitorare zone più vaste rispetto ad una telecamera fissa; questo fatto permette, a parità di dimensioni del perimetro da controllare, di utilizzare un numero inferiore di telecamere. Nasce a questo punto il problema di come controllare le telecamere; allo stato attuale, i sistemi di videosorveglianza prevedono una stanza, chiamata centrale di videosorveglianza, nella quale sono presenti diversi monitor che mostrano le riprese che vengono inviate dalle varie videocamere, dove, uno o più operatori guardando i monitor vedono c'è un attività sospetta o meno e in caso di emergenza oltre a dare l'allarme devono comandare per mezzo di opportuni joystick le eventuali videocamere PTZ per poter seguire gli spostamenti di un evento; il problema di questa soluzione è dovuto principalmente alla presenza di un operatore umano in quanto, dovendo osservare un certo numero di monitor, può distrarsi e quindi non accorgersi di una eventuale situzione di allarme che si è verificata, oppure, nel caso stia controllando una telecamera per seguire un evento, essendo lui stesso concentrato su di una sola telecamera(cioè osseerva un solo monitor), non si

accorge di altre eventuali situazioni di allarme che si verificano in altre zone, che al momento non sono monitorate da l'operatore stesso, rendendo così necessaria un eventuale presenza di un altro operatore. Si evince così che per monitorare una vasta area sono neccessari diversi operatori. Per ovviare a questi problemi sono stati introdotti i sistemi di videosorveglianza automatici cioè sistemi che riescono a sorvegliare un perimetro in maniera autonoma e, tramite tecniche di visione computazionale (che non fanno parte di questo progetto) riescono ad identificare una situazione di emergenza (come ad esempio un intrusione oppure la presenza di veicoli con una targa non identificata...) che avviene nella loro zona di competenza. A questo punto ci sono due tipi di possibli strategie di controllo di un sistema automatizzato: uma strategia centralizzata e una distribuita. In un ambito centralizzato si ha che il controllo è svolto da un unità di elaborazione centrale, la quale gestisce e comunica con tutte le telecamere; questo tipo di soluzione presenta alcuni svantaggi come ad esempio il fatto che se l'unità di elaborazione, che tra le altre cose deve avere prestazioni maggiori rispetto al caso distribuito, avesse un guasto, di conseguenza l'intero sistema sarebbe fuori uso ed inoltre la sua complessità aumenta all'aumentare del numero di videocamere presenti.

Nel caso di controllo distribuito invece il sistema è formato solo da videocamere le quali a differenza del caso precedente sono intelligenti ovvero sono dotate di un microprocessore per mezzo del quale possono eseguire tutte le elaborazioni richieste. Questo tipo di architettura presenta alcuni vantaggi che sono rilevanti in particolare nell'ambito della videosorveglianza:

- la comunicazione avviene solo a livello locale (in particolare con le videocamere adiacenti) e questo consente l'utilizzo di sistemi di comunicazione meno potenti e indipendenti l'uno dall'altro
- il sistema così implementato è robusto rispetto ai guasti in quanto un eventuale problema ad una telecamera non si ripercuote sull'intero sistema
- garantisce la modularità del sistema ovvero, se si vuole aggiungere un qualche nodo al sistema di videocamere non occorre apportare modifiche all'intero sistema di telecamere.

In questo progetto vogliamo analizzare degli algoritmi *distribuiti* (in particolare ci occuperemo di algoritmi asincroni e broadcast) che permettano a delle telecamere PTZ di potersi gestirsi autonomamente cioè che possano dividersi in maniera ottima il perimetro da patrollare e riescano a seguire un evento che si verifica nel loro campo visivo, eventualmente, se necessario anche coordinandosi tra di loro tramite uno scambio di mesasggi; il tutto senza la neccessità di avere un unità centrale di elaborazione ma solo interagendo tra di loro a livello locale.

Figura 1.1: Esempio di centrale di videosorveglianza

Figura 1.2: Esempio di videocamera PTZ

1.1 Stato dell'arte e lavori precedenti

Il problema del patroling (nell'ambito di controllo distribuito) è stato ampiamente studiato in letteratura nel caso di agenti autonomi che si muovono in un determinato spazio per poter determinare la regione ottima di pattugliamento; nel nostro caso invece dobbiamo applicare questa teoria ad un sistema di videocamere formato da telecamere in posizione fisse di tipo PTZ che possano determinare la sezione di perimetro ottimale da patrollare comunicando solo con le telecamere adiacenti. Il presupposto fondamentale per poter svolgere operazioni di patrolling in maniera ottima (nel capitolo successivo verrà definito formalmente cosa si intende per ottimo) è quello di riuscire a partizionare in maniera ottima il nostro perimetro da sorvegliare: negli articoli [1], [2] e [3] sono stati studiati degli algoritmi per il partizionamento ottimo di un perimetro

nell'ambito di reti di telecamere intelligenti in particolare sono stati analizzati diversi algoritmi nel contesto del tipo di protocollo di comunicazione che veniva usato dalle telecamere. In particolare, in [1] è stata introdotta l'idea di partizionamento ottimo di un perimetro (cioè le telecamere devono andare alla loro massima velocità, e la soluzione ottima è quella che minimizza i tempi tra due visite successive dello stesso punto) ed è stato sviluppato l'algoritmo di partizionamento nel caso di comunicazione di tipo sincrono, è stato proposto un valido algoritmo di tracking, in particolare nella gestione del trade-off tra patrolling e tracking; in [2],[3] invece sono stati studiati algoritmi per il partrizionamento di un perimetro che utilizzassero protocolli di comuicazione di tipo sincrono e gossip simmetrico, di questi due algoritmi ne è stata dimostrata la convergenza alla soluzione ottima. In [2][3] inoltre è stato introdotto un algoritmo di partizionamento con comunicazione di tipo asimmetrico-broadcast ma non ne è stata dimostrata formalmente la convergenza, se non per mezzo di simulazioni numeriche.

Per quanto riguarda invece il secondo punto del nostro progetto cioè il tracking di un evento, nella letteratura si trovano quasi sempre riferimenti ad articoli riguardanti problemi di tracking di oggetti per mezzo di visione computazionale, il che come accennato in precedenza è ben oltre allo scopo del nostro progetto che invece è più orientato al tracking di un generico evento, nel senso di seguire un oggetto conoscendone posizione e velocità, (e quindi non al riconoscimento di un oggetto in un immagine vista dalla telecamera) e alla coordinazione tra le telecamere adiacenti che dovranno risucire a "intercettare" l'evento una volta che esso esce dal campo visivo della telecamera vicina; per svolgere questo punto ci siamo basati sull'idea descritta in [4], dove viene descritto un valido algoritmo per gestire il tracking di un evento, viene descritta la fase di *trade-off* da patrolling a tracking ovvero il comportamento di una telecamera quando si trova a dover passare da una fase di patrolling ad una fase di tracking, in particolare il comportamento di una telecamera quando una della telecamere adiacenti è nella fase di tracking di un evento.

1.2 Il nostro lavoro

Il nostro lavoro consiste nel studiare un algoritmo di controllo *distribuito* per un sistema di sorveglianza perimetrale, in sostanza sarà da costruire un algoritmo (si intende sempre distribuito) che sarà poi implementato direttamente all'interno della telecamera per fare in modo che esa riesca ad eseguire i seguenti compiti fondamentali:

- Riesca a partizionare in maniera ottima il perimetro di cui bisogna fare patrolling
- riesca a seguire un evento che si presenta nella sua zona di competenza
- gestisca la fase di trade-off, comunicando e/o ricevendo informazioni dalle telecamere adiacenti in modo da non perdere l'evento che deve essere

seguito quando esso esce dal campo visivo di una telecamera ed entra in quello della telecamera vicina.

Tutto questo dovrà essere svolto nel caso in cui le telecamere utilizzino un protocollo di comunicazione di tipo *asincrono e broadcast* che ha come vantaggi il fatto di non richiedere nessun tipo di meccanismo di sincronizzazione temporale tra le telecamere(come accade nel caso sincrono) e inoltre, essendo di tipo broadcast, non richiede la presenza di messaggi di ack da parte della telecamera destinataria, sostanzialmente, una telecamera invia un messaggio quando lo ritiene opportuno e non si preoccupa di chi lo ha ricevuto e se lo ha ricevuto; inoltre, cosa non da poco, grazie alle sue caratteristiche questo tipo di protocollo è quello più semplice da implementare. Dei tre punti descritti sopra, il primo è senz'altro quello più importante, infatti per poter fare il patrolling di un perimetro (o di un area in generale) in maniera ottimale è prima di tutto neccessario partizionare la suddetta area in maniera da minimizzare il tempo tra due visite successive. Trovare un algoritmo distribuito, quindi che utilizzi solamente le informazioni che vengono ricevute dalle telecamere adiacenti, che realizzi un partizionamento ottimo non è una cosa partircolarmente difficile, il problema è dimostrare che data una qualunque configurazione iniziale, l'algoritmo faccia si che si ottenga una suddivisione ottima del perimetro. Questa dimostrazione è molto difficile, se non impossibile nel caso di un perimetro continuo e quindi, per cercare di poter dimostrare analiticamente la convergenza dell'algoritmo si è pensato di *discretizzare il perimetro* in modo da poterci ricondurre ad un numero finito di casi possibili, e mostrare che per ognuno di questi casi esiste una sequenza di comunicazione tra le telecamere che fa si che l'algoritmo converga al parizionamento ottimo. Si osservi che questo è un algoritmo randomizzato in quanto le telecamere non hanno un ordine prefissato con cui comunicano, quindi l'unica cosa che si può dire è che sicuramente prima o poi capiterà tale sequenza, ma non si può dire quando.

La videocamera è sempre in fase di patrolling, che consiste nel "percorrere" avanti e dietro la parte di perimetro che le è stata assegnata, e ci resta fin tanto che non intercetta un evento; quando intercetta un evento, la videocamera passa allo stato di tracking e, con l'ausilio di un predittore (filtro di kalman) cerca di seguire l'evento; nel frattempo inizia a comunicare alle telecamere vicine il suo stato e alcune informazioni sull'evento in modo che queste ultime riescano a raggiungere in tempo utile la posizione oltre alla quale la videocamera in tracking non riesce più a sorvegliare l'evento in modo da non perderlo; inoltre, durante la fase di tracking, la telecamera che è in quello stato, ovvero che sta seguendo l'evento, eventualmente comuinicherà più spesso perchè la fase di tracking ha la priorità massima rispetto al patrolling. Di seguito forniamo un brevissimo sunto del contenuto dei capitoli succesivi:

Nel capitolo 2 verrà formalizzato matematicamente il problema di partizionamento e viene fornita una dimostrazione della convergenza di tale algoritmo;

Nel capitolo 3 vengono introdotti gli altri punti del progetto ovvero l'algoritmo di tracking e la messa in opposizione di fase delle telecamere, che serve a

svolgere il patrolling in maniera più efficiente come verrà spiegato nel suddetto capitolo e la gestione del trade-off;

Nel capitolo 4 invece verranno presentate le simulazioni del sistema di telecamere in presenza di un evento da seguire e al variare di alcuni parametri del sistema come ad esempio il rate di comunicazione oppure la simulazione di un guasto ad una telecamera.

Capitolo 2

Definizione del problema

Questo capitolo rappresenta un punto chiave dell'intero lavoro in quanto è il punto di partenza per l'esecuzione dell'intero progetto. In questa sezione ci occupiamo di definire in modo completo e rigoroso le variabili utilizzate. Per fare questo ci siamo rifatti a lavori precedenti come [1], [2] e [3]. Come già illustrato in precedenza in vari casi è stata dimostrata in modo rigoroso e matematico la convergenza all'ottimo di algoritmi di consenso per la risoluzione di problemi simili al nostro, in particolare tali lavori sono stati affrontati in [2]. In tale articolo si è però messo in evidenza che nel caso di comunicazione broadcast asimmetrica si riusciva solo a verificare sperimentalmente la convergenza all'ottimo ma non si riusciva a dimostrarla come negli altri casi. In questo caso si vuole tentare una strada diversa per cercare di dimostrare tale convergenza. Il punto focale di questo nuovo approccio è la discretizzazione del perimetro da patrollare. Ovviamente il numero di quanti è assolutamente arbitrario e va scelto in maniera tale da garantire il corretto funzionamento del sistema; in particolare la dimensione del quanto deve essere abbastanza piccola in modo da non compromettere le prestazioni e la precisione del sistema. Se il quanto è piccolo si possono approssimare i limiti fisici delle videocamere all'ultimo quanto totalmente compreso nel loro campo d'azione.

Come specificato in [3] assumiamo che la posizione delle videocamere sia un input del problema e quindi non modificabile. Supponiamo inoltre i limiti fisici delle videocamere siano tali da avere delle sovrapposizioni altrimenti se non ci fossero sovrapposizione, la ricerca dell'ottimo sarebbe infondata.

2.1 Definizione matematica del problema

Come già detto in precedenza, nel nostro lavoro si considera il patrolling di un area perimetrale; ipotizziamo che il tratto da patrollare sia unidimensionale;tale ipotesi semplifica la formulazione del problema e non è limitativa per la risoluzione del problema. Altre ipotesi semplificative si possono fare per le videocamere:

- il campo visivo delle videocamere può essere approssimato ad un punto
- il campo visivo delle videocamere non viene alterato dalla prospettiva durante il movimento laterale della stessa

Fatte tali premesse definiamo in maniera matematica le grandezze coinvolte nel seguito del documento.

- $L = [0, L_{TOT}] \subset \mathbb{R}^+$è la lunghezza totale del perimetro da patrollare. Nel nostro caso si considera che il perimetro sia una "rettilineizzato".
- N è il numero di videocamere del sistema di videosorveglianza in questione.
- $\bar{q}$ è la dimensione lineare del quanto.
- $M = \frac{L_{TOT}}{\bar{q}}$ è il numero totale di quanti. Ovviamente il numero di quanti è un numero arbitrario che può essere scelto in maniera arbitraria da parte del progettista. Come vedremo in seguito scegliere M multiplo di N porta a dei vantaggi in termini di prestazioni del sistema, è quindi consigliabile fare una scelta di $\bar{q}$ che garantisca tale situazione.
- si può riscivere L come $L = \bigcup_{i=1}^{M} q_i$
- $D_i = \bigcup_{i=q_{i,inf}}^{q_{i,sup}} q_i \subset L$ è la copertura totale della i-esima videocamera dovuta alla topologia in esame, la configurazione delle videocamere e i loro vincoli fisici.
- $A_i = \bigcup_{i=q_{i,left}}^{q_{i,right}} q_i \subset D_i$ rappresenta la zona di perimetro effetivamente coperta dalla i-esima videocamera e ovviamente deve esssere compresa all'interno dei vincoli fisici.
- $z_i(t) : \mathbb{R}^+ \to D_i$è la funzione che mappa la posizione del centro del campo visivo della ì-esima videocamera in funzione del tempo.
- ogni videocamera ha una propria velocità $v_i(t) \in [-V_{i,max}, V_{i,max}]$. Data la definizione delle grandezze precedenti rsulta più comodo espremierla in (n° quanti/secondo).

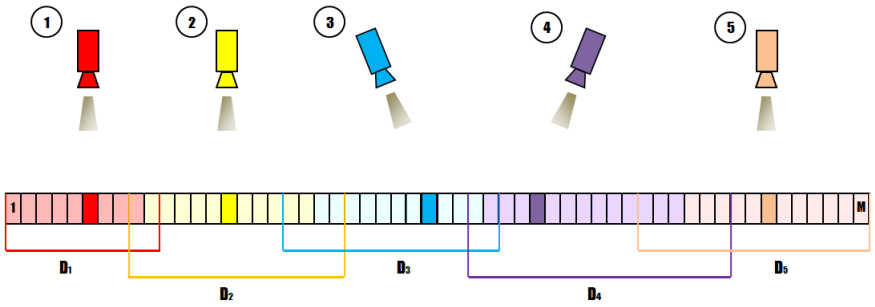


Figura 2.1: Rappresentazione di un sistema di videosorveglianza con grandezze in evidenza. Con colore tenue vengono evidenziate le A_i mentre con colore più deciso vengono evidenziati i valori assunti da $z_i(t)$

2.2 Comunicazione tra le videocamere

Il modello di comunicazione tra videocamere appartiene al sistema broadcast asimmetrico. Questo schema è del tutto generale e prevede che:

- una sola videocamera stia comunicando ad un dato istante
- le videocamere che sono interessate dalla comunicazione in quel dato istante ricevono l'informazione ma non comunicano a loro volta

In particolare nel nostro caso ipotiziamo che ogni videocamera può trasmettere e/o ricevere solo con le videocamere adiacenti. In particolare quindi assumiamo il seguente grafo di comunicazione.

Figura 2.2: Grafo di comunicazione

Come si può notare il grafo è fortemente connesso e quindi utilizzando un algortimo di consensus certamente si può arrivare ad una condizione di consenso.

Dato che il nostro schema di comunicazione è il più generale e realistico sistema che si possa immaginare ipotizziamo anche che ogni videocamera comunichi ad un istante aleatorio. Per questo motivo il processo di comunicazione di ogni videocamera può essere modellizzato come un processo aleatorio di Poisson di parametro $\lambda > 0$, ovvero il numero di comunicazioni che avviene in un dato intervallo di tempo è una variabile aleatoia distribuita nel seguente modo:

$$P[N(t+s) - N(s) = n] = e^{-\lambda t} \frac{(\lambda t)^n}{n!}$$

che ha media

$$\mu = \lambda t$$

Un aspetto importante di questo processo è la distribuzione dei tempi di interarrivo. Se indichiamo con t_n l'istante in cui la videocamera esegue una comunicazione, definiamo tempo di interarrivo $\tau_n = t_n - t_{n-1}$. I tempi di interarrivo sono ovviamente anche'essi aleatori e sono distribuiti come variabili aleatorie esponenziali indipendenti ed identicamente distribuite di parametro λ e quindi hanno distribuzione:

$$P_\tau(a) = 1 - e^{-\lambda a}$$

e densità.

$$p_\tau(a) = \lambda e^{-\lambda a}$$

2.3 Analisi sul partizionamento ottimo

Come già accenanto in precedenza il primo obbiettivo del nosro lavoro è quello di arrivare a determinare la condizione di partizionamento ottimo del perimetro. La partizione ottima del perimetro è quella che assicura che il periodo di patrolling sia quello minore possibile, ovvero, quello che minimizza il tempo massimo che intercorre tra due visite consecutive ad uno stesso luogo. Come è stato fatto nei lavori precedenti è bene fare delle analisi su questo tema per avere l'idea dell'obbiettivo a cui dovremo giungere attraverso l'algoritmo di controllo distribuito.

2.3.1 Partizionamento ottimo senza vincoli fisici

Per iniziare si può dire che il Lemma 1 riportato in [1] è valido anche nel nostro caso. Le cose iniziano a cambiare se $N \geq 2$. Infatti mentre nel caso continuo la soluzione ottima era una tra le infinite combinazioni delle regioni sorvegliate da ciascuna videocamera, nel caso di perimetro discretizzato si deve procedere ad una approssimazione di tale soluzione ottima in quanto si è vincolati a scegliere dei limiti che coincidono con un numero intero di quanti. Quindi intuitivamente si può affermare che nel caso in cui la soluzione discretizata non coincide con quella ottima si ha un tempo di patrolling leggermente più alto. Come in [1] riportiamo la seguente proposizione.

Proposizione 1. *La copertura ottima del percorso rispetto alla minimizzazione del periodo di patrolling senza i vincoli di copertura, nell'ipotesi che ciascuna telecamera vada alla sua velocita massima $v_{i,max}$ e pattugli la propria sezione con un moto periodico di periodo Ti, si ottiene imponendo che le sezioni di copertura A_i siano a due a due disgiunte. Considerando il tempo ottimo di patrolling nel perimetro continuo $\bar{T}_{cont}$ si ha che il tempo ottimo di patrolling nel perimetro discretizzato è limitato dalla seguente disuguaglianza:*

$$\bar{T}_q \leq \bar{T}_{cont} + \frac{2\bar{q}}{v_{min,max}}$$

dove

$$v_{min,max} = \min_i(v_{i,max})$$

e

$$\bar{T}_{cont} = \frac{2L_{TOT}}{\sum_{i=1}^{N} v_{i,max}}$$

Dimostrazione. Se si considera la partizione del perimetro continuo si possono avere diversi casi in cui tali soglie non coincidono con gli intervalli discretizzati. In tali situazioni è necessario discretizzare la soluzione assegnando il quanto coinvolto alla videocamera più a sinistra o alla videocamera più a destra. In questo caso si aggiungono al più 2 elementi da patrollare (nel caso peggiore uno viene aggiunto a destra e uno a sinistra) e quindi il tempo di patrolling della videocamera in questione aumenta al più di $\frac{2\bar{q}}{v_{i,max}}$. Se scegliamo la minima velocità massima si ottiene il bound scritto nella proposizione.

□

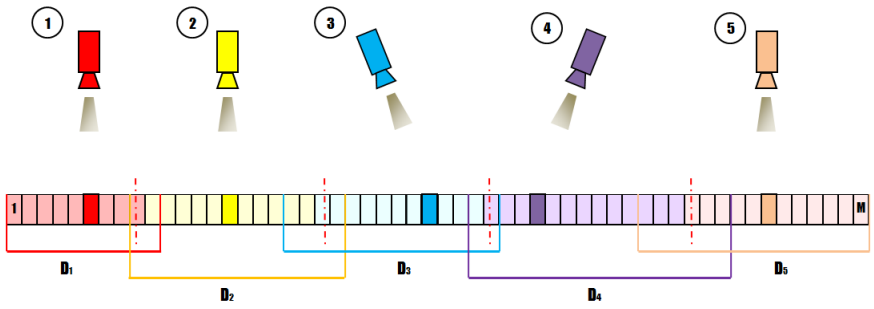


Figura 2.3: In questa immagine vengono evidenziate le differenze tra le partizioni ottime nel perimetro continuo (assi rossi tratteggiati) e quella nel perimetro discretizzato evidenziata dai colori

Si può fare un osservazione su questa proposizione: purtroppo data l'aleatorietà del metodo di partizionamento nel caso discreto non è possibile fornire un espressione in forma chiusa, tuttavia l'upper buond precedente mostra che i tempi ottimi nel perimetro continuo e nel quantizzato differiscono per una quantità proporzionale alla lunghezza del quanto. Come è ovvio facendo tendere $\bar{q} \to 0$ si converge alla soluzione ottima nel continuo. Mettendo in evidenza quanto detto nella sezione in cui sono state definite le grandezze è bene scegliere $\bar{q}$ abbastanza piccolo; la scelta di un tale $\bar{q}$ porta ad avere una convergenza $\bar{T}_q \to \bar{T}_{cont}$.

2.3.2 Partizionamento ottimo con vincoli fisici

La soluzione determinata nel paragrafo precedente non è la soluzione che si ottiene in condizioni generali, in quanto nella realtà sono preseni dei vincoli dovuti alle caratteristiche tecniche delle videocamere e alla conformazione del perimetro. La soluzione non vincolata coincide con quella vincolata se e solo se quest'ultima soddisfa $A_i \subseteq D_i \, \forall i$. Introducendo il periodo di patrolling in presenza di vincoli $\bar{T}_{q,c}$ si ha che vale la seguente disuguaglianza

$$\bar{T}_{q,c} \geq \bar{T}_q$$

dove il segno di uguaglianza vale solo nel caso in cui i due partizionamenti coincidono.

La proposizione II.2 scritta in [3] per il caso di perimetro continuo vale anche nel caso di perimetro discretizzato, per tale motivo la riportiamo di seguito.

Proposizione 2. *Se la soluzione non vincolata è tale per cui* $\exists i$ *tale che* $A_i \nsubseteq D_i$ *, la copertura ottimale si ottiene dividendo il dominio in* $L_l = \bigcup_{i=1}^{q_l} q_i$ *e* $L_r = \bigcup_{i=q_r}^{M} q_i$*, con* $q_l = q_{i,inf} - 1$*,* $q_r = q_{i,sup} + 1$ *e considerando due distinti problemi di copertura. Siano* T_o^l *e* T_o^r *i periodi ottimali per i sottodomini, il periodo globale di copertura viene ottenuto come* $T_{o,c} = max\left\{T_o^l, T_o^r\right\}$*.*

La strategia per ottenere i tempi ottimi sui sottodomini è quella di applicare ricorsivamente le proposizioni 1 e 2 ogniqualvolta vengano violati i vincoli fisici, ovvero la soluzione ottimale si ottiene dividendo il perimetro in N segmenti separati e assegnandoli ciascuno a ogni telecamera che si sposta con un moto periodico alla velocità massima. In tal modo il problema di minimizzare il periodo di patrolling si riduce al problema del la scelta ottimale di A_i per ogni videocamera.

2.4 Algoritmo di partizionamento

Utilizzando la stessa impostazione mostrata in [3] in questa sezione esponiamo l'algoritmo che abbiamo sviluppato per ottenere il partizionamento ottimo del perimetro discretizzato. Consideriamo una consizione iniziale al tempo t=0 a cui corrisponde una partizione descritta dalle regioni $A_i(0)$ con $i = 1, ..., N$ che in generale non coincide con la soluzione ottima. Ogni videocamera iterativamente, sfruttando le informazioni relative al suo stato e alle proprie caratteristiche, e quelle provenienti dalle videocamere aggiorna la propria regione da controllare.

Inizialmente si ipotizza che la condizione iniziale $\{A_1(0), ..., A_N(0)\}$ soddisfi tre vincoli:

1. vincoli fisici: $A_i(0) \subseteq D_i$con $i = 1, ..., N$
2. vincolo di copertura: $\bigcup_{i=1}^{N} A_i(0) = L$

3. vincoli di interlacciamento, ovvero

$$q_{i,left}(0) \leq q_{i+1,left}(0)$$

$$q_{i,right}(0) \leq q_{i+1,right}(0)$$

In particolare per effetto dei vincoli precedenti si deve avere $q_{1,left}(0) = 1$e $q_{N,right}(0) = M$.

Lo scopo è quello di raggiungere in questo modo la configurazione ottima, cioè quella che garantisce il periodo di patrolling minore e il soddisfacimento dei vincoli appena elencati.

Come detto in precedenza l'intero sistema si basa sul protocollo di comunicazione broadcast asimmetrico. Questo protocollo è molto generale e non richiede alcun tipo di sincronizzazione tra le videocamere. Ogni videocamera trasmette in istanti casuali e ad ogni comunicazione avviene uno scambio unidirezionale di informazioni. Per tale ragione, ad ogni iterazione, una sola videocamera invia informazione ai vicini e solo questi ultimi agiscono sul partizionamento.

2.4.1 Implementazione senza vincoli fisici

Trasmissione

Ad ogni istante $t \in \mathbb{N}$ una sola videocamera sta trasmettendo delle informazioni ai suoi vicini. Supponiamo in questo frangente che la videocamera che trasmette sia la videocamera i e quindi solo le videocamere $i-1$ e $i+1$ stanno ricevendo le informazioni.

Variazione degli estremi

NB: per eseguire i calcoli correttamente si considerano gli estremi sinistro e destro. Per come è stato definito A_i si ha che il limite destro coincide con $q_{i,right}$ mentre il limite sinistro coincide con $q_{i,left} - 1$

Le videocamere diverse dalla $(i-1)$-esima e $(i+1)$-esima lasciano invariati i propri estremi mentre la $(i-1)$-esima e la $(i+1)$-esima li variano seguendo le seguenti relazioni:

- $(i-1)$-esima videocamera:
  - lascia inalterato il limite sinistro ovvero

$$q_{i-1,left}(t+1) = q_{i-1,left}(t)$$

  - modifica il limite destro, in base alle informazioni apperna ricevute, nel seguente modo:
    - calcolo il valore ottimale come nel caso continuo ovvero,

$$q^* = \frac{(q_{i-1,left}(t) - 1) \cdot v_i + q_{i,right}(t) \cdot v_{i-1}}{v_{i-1} + v_i}$$

* se $q^* < q_{i,left}(t)$ allora poni $q_{i-1,right}(t+1) = q_{i,leftt}(t)$ per non lasciare zone scoperte altrimenti procedi nel seguente modo.

  Generalmente q^* non assume valori interi e quindi si rende necessario assegnare il quanto in questione in modo da minimizzare il periodo di patrolling. Per fare ciò calcoliamo i tempi che si otterebbero nei due casi: si nota che per assegnare l'intero quanto alla videocamera $i-1$ si ha che $q_{i-1,right}(t+1) = \lfloor q^* \rfloor + 1$ mentre per assegnare il quanto in questione alla videocamera i si imposta $q_{i,left}(t+1) = \lfloor q^* \rfloor$.

$$T_{i-1}(t+1) = \frac{\lfloor q^* \rfloor + 1 - (q_{i-1,left}(t+1) - 1)}{v_{i-1}}$$

$$T_i(t+1) = \frac{q_{i,right}(t+1) - (\lfloor q^* \rfloor - 1)}{v_i}$$

  se $T_{i-1}(t+1) < T_i(t+1)$, allora

$$q_{i-1,right}(t+1) = \lfloor q^* \rfloor + 1$$

  se $T_{i-1}(t+1) > T_i(t+1)$, allora

$$q_{i-1,right}(t+1) = \lfloor q^* \rfloor$$

  se $T_{i-1}(t+1) = T_i(t+1)$, allora

  assegna in maniera randomizzata il quanto in questione alla videocamera $(i-1)$-esima o i-esima.

- $(i+1)$-esima videocamera:

  – lascia inalterato il limite destro, ovvero

$$q_{i+1,right}(t+1) = q_{i+1,right}(t)$$

  – modifica il limite destro, in base alle informazioni apperna ricevute, nel seguente modo:

    * calcolo il valore ottimale come nel caso continuo ovvero,

$$q^* = \frac{(q_{i,left}(t) - 1) \cdot v_{i+1} + q_{i+1,right}(t) \cdot v_i}{v_i + v_{i+1}}$$

    * se $q^* > q_{i,right}(t)$ allora poni $q_{i+1,left}(t+1) = q_{i,right}(t)$ per non lasciare zone scoperte altrimenti procedi nel seguente modo.

      Generalmente q^* non assume valori interi e quindi si rende necessario assegnare il quanto in questione in modo da minimizzare il periodo di patrolling. Per fare ciò calcoliamo i tempi che si otterebbero nei due casi: si nota che per assegnare l'intero quanto

alla videocamera i si ha che $q_{i,right}(t+1) = \lfloor q^* \rfloor + 1$ mentre per assegnare il quanto in questione alla videocamera $i+1$ si imposta $q_{i,left}(t+1) = \lfloor q^* \rfloor$.

$$T_i(t+1) = \frac{\lfloor q^* \rfloor + 1 - (q_{i,left}(t+1) - 1)}{v_i}$$

$$T_{i+1}(t+1) = \frac{q_{i+1,right}(t+1) - (\lfloor q^* \rfloor - 1)}{v_{i+1}}$$

se $T_i(t+1) < T_{i+1}(t+1)$, allora

$$q_{i+1,left}(t+1) = \lfloor q^* \rfloor + 1$$

se $T_i(t+1) > T_{i+1}(t+1)$, allora

$$q_{i+1,left}(t+1) = \lfloor q^* \rfloor$$

se $T_i(t+1) = T_{i+1}(t+1)$, allora

decidi in maniera randomizzata se assegnare il quanto in questione alla videocamera (i)-esima o $(i+1)$-esima.

Osservazione: siamo stati costretti a introdurre l'aleatorietà nel caso in cui i due tempi fossero risultati uguali. Inizialmente, in casi come questi assegnavamo il quanto in maniera deterministica sempre alla videocamera più a sinistra. Abbiamo trovato dei casi in cui tale algoritmo non portava all'ottimo e quindi siamo stati costretti ad abbandonare tale idea in favore di quela appena esposta. Riporto di seguito un esempio.

Consideriamo il caso di $N = 3$ videocamere con velocità $v_i = k$ con $i = 1, 2, 3$ e il caso in cui $M = 12$. È facile intuire che in tale situazione la partizione ottima sia quella di assegnare 4 quanti ad ogni videocamera. Se partiamo dalla condizione iniziale descritta da:

- $A_1 = \bigcup_{i=1}^{5} q_i$
- $A_2 = \bigcup_{i=6}^{9} q_i$
- $A_3 = \bigcup_{i=10}^{12} q_i$

e assegnamo il quanto sempre a sinistra si rimane sempre nella stessa condizione.

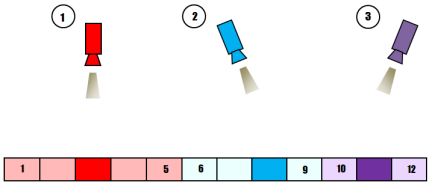


Figura 2.4: Rappresentazione caso in cui non funziona l'assegnazione a sinistra

Osservazione: Nel caso in cui il numero di celle non sia multiplo del numero di videocamere può accadere che, a causa della scelta randomizzata, una cella può essere continuamenrte scambiata tra due videocamere adiacenti, non giungendo mai ad una situazione di consernso stabile. Per evitare tale inconveniente, data l'arbitrarietà del numero delle celle è consigliabile dividere il perimetro in un numero multiplo di celle.

2.4.2 Implementazione con vincoli fisici

In questo caso l'algoritmo viene modificato per fare in modo che vengano rispettati i vincoli fisici $D_1, ..., D_N$. In condizioni generali l'algoritmo precedente può portare a violare i vincoli fisici. Per fare in modo che essi vengano rispettati è necessario che ogni videocamera memorizzi, oltre ai propri limiti fisici, anche quelli delle videocamere adiacenti. Per fare in modo che questi limiti vengano rispettati è sufficiente un controllo finale. Come nel caso precedente dividiamo l'algoritmo nel caso in cui si stia aggiornando il limite sinistro della videocamera $(i-1)$-esima o $(i+1)$-esima.

- $(i-1)$-esima videocamera: se dall'esecuzione dell'algoritmo precedente risulta $q_{i-1,right}(t+1) > q_{i,sup}$, allora

$$q_{i-1,right}(t+1) = q_{i-1,sup}$$

- $(i+1)$-esima videocamera: se dall'esecuzione dell'algoritmo precedente risulta $q_{i+1,left}(t+1) > q_{i,inf}$, allora

$$q_{i+1,left}(t+1) = q_{i+1,inf}$$

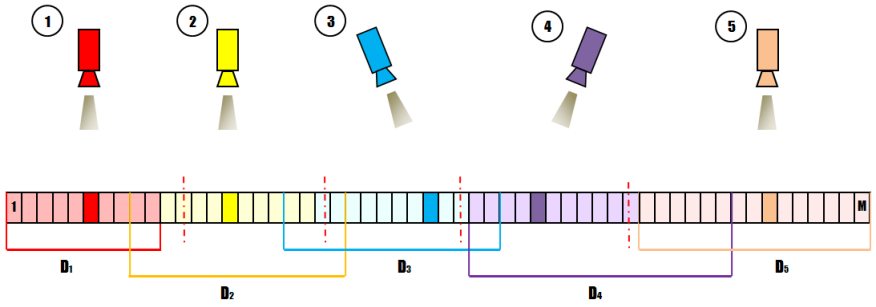


Figura 2.5: Rappresentazione del comportamento dell'algoritmo con rispetto dei vincoli fisici

2.5 Dimostrazione convergenza all'ottimo

Lo scopo di questa dimostrazione è quello di dimostrare che il nostro algoritmo per il partizionamento del perimetro da sorvegliare e convergente all'ottimo.Forniremo la dimostrazione solo per il caso di telecamere con velocita unitaria in quanto l'estensione al caso generale è fatta semplicemente considerando il periodo di patrolling ((numero di celle)/(velocità espressa in celle al secondo)) al posto del numero di celle che appartengono ad ogni telecamera, nella dimostrazione, per completezza, vengono considerati sia il caso in cui M è un multiplo di N e sia il caso in M non è multiplo di N.

Definiamo

- $C_i(t)$ il numero di celle presenti in $A_i(t)$ e quindi $C_i(t) = q_{i,right}(t) - q_{i,left}(t) + 1$.
- $\bar{C}(t) = \max_i C_i(t)$

L'ottimo e dato da:

$$\bar{C}_{OTT} = \begin{cases} \frac{M}{N} & \text{se M è multiplo di N} \\ \lfloor \frac{M}{N} \rfloor + 1 & \text{se M non è multiplo di N} \end{cases}$$

Per poter dimostrare la nostra tesi, vogliamo sfruttare il teorema limite di successione che afferma che una successione decrescente e inferiormente limitata, converge al suo estremo inferiore. La successione che vogliamo analizzare è la successione dei $\bar{C}_i(t)$.

Proposizione 3. *La successione $\bar{C}(t)$ con $t \in \mathbb{N}$ è non crescente ed è inferiormente limitata da $\bar{C}_{OTT}$.*

Dimostrazione. Il fatto che la successione è inferiormente limitata inferiormente da $\bar{C}_{OTT}$ può essere provato sfruttando il fatto che, poichè l'algoritmo soddisfa i vincoli di copertura, in ogni istante l'unione degli intervalli di ogni telecamera deve essere pari alla lunghezza del perimetro. Per provare che $\bar{C}(t)$ non può decrescere al di sotto dell'ottimo supponiamo per assurdo che ad un dato istante sia $\bar{C}(t) < C_{OTT}$ allora si ha che l'unione degli intervalli è minore della lunghezza del perimetro il che è assurdo in quanto ciò violerebbe i vincoli di copertura.

Dimostriamo ora la non crescenza della successione. Per fare ciò consideriamo tutti i casi che possono portare alla variazione degli estremi e verifichiamo che $\bar{C}(t)$ non può crescere per ogni possibile iterazione.

Consideriamo il caso in cui la videocamera i comunica e provoca lo spostamento del limite destro relativo alla videocamera $(i-1)$. Supponiamo inoltre che sia proprio la $(i-1)$-esima videocamera una delle videocamere con il massimo numero di celle da patrollare.

$$q^*(t+1) = \frac{q_{i-1,left}(t) - 1 + q_{i,right}(t)}{2} =$$

$$= \begin{cases} q^*(t+1) \in \mathbb{N} & (1) \\ q^*(t+1) < q_{i,left}(t) \Rightarrow q_{i-1,right}(t+1) = q_{i,left}(t) & (2) \\ q^*(t+1) \notin \mathbb{N} & (3) \end{cases}$$

Caso 1

Si ha che

$$C_{i-1}(t+1) = q_{i-1,right}(t+1) - q_{i-1,left}(t+1) + 1 = q^*(t+1) - q_{i-1,left}(t+1) + 1$$

con

$$q^*(t+1) = \frac{q_{i-1,left}(t) - 1 + q_{i,right}(t)}{2}$$

Nel caso peggiore, ovvero quello che massimizza $C_{i-1}(t+1)$, si ha che entrambe le videocamere interessate dalla comunicazione sorvegliano $2\bar{C}(t)$ celle e che non ci sia sovrapposizione. In queste condizioni si ha che:

$$q_{i,right}(t) = q_{i-1,left}(t) - 1 + 2\bar{C}(t)$$

e sostituendo si ottiene:

$$C_{i-1}(t+1) \leq \frac{q_{i-1,left}(t) - 1 + q_{i-1,left}(t) - 1 + 2\bar{C}(t)}{2} - q_{i-1,left}(t+1) + 1 = \bar{C}(t)$$

Caso 2

Questo caso rappresenta la situazione in cui intervengono i limiti fisici. In particolare in questa situazione si dovrebbe posizionare $q_{i-1,right}(t+1)$ in una posizione inferiore rispetto a $q_{i,left}(t)$ ma questo lascerebbe scoperta una zona del perimetro e quindi si pone $q_{i-1,right}(t+1) = q_{i,left}(t)$. In questo caso quindi si ha che:

$$C_{i-1}(t+1) = q_{i-1,right}(t+1) - q_{i-1,left}(t+1) + 1 = q_{i,left}(t) - q_{i-1,left}(t) + 1$$

per non avere zone scoperte al tempo t deve essere verificata:

$$q_{i-1,left}(t) \geq q_{i,right}(t)$$

quindi si ha che

$$C_{i-1}(t+1) \leq q_{i,left}(t) - q_{i,right}(t) + 1 = C_i(t+1) \leq \bar{C}(t)$$

Caso 3

In questo caso, si ha che il punto di ottimo cade tra due numeri naturali quindi, essendo in un contesto discreto e neccessario scegliere a quele camera assegnare la sorveglianza di quella cella. Ci sono quindi due ulteriori casi:

$$q^*(t+1) = \frac{q_{i-1,left}(t) - 1 + q_{i,right}(t)}{2} = \begin{cases} r_i(t+1) = \lfloor q^*(t+1) \rfloor & (4) \\ r_i(t+1) = \lfloor q^*(t+1) \rfloor + 1 & (5) \end{cases}$$

- **caso (4)**: in questo caso si ha che la cella contesa viene assegnata alla telecamera che sta a destra

  $C_{i-1}(t+1) = q_{i-1,right}(t+1) - q_{i-1,left}(t+1) + 1 = \lfloor q^*(t+1) \rfloor - q_{i-1,left}(t+1) + 1$

  dato che $q^*(t+1) \notin \mathbb{N}$si ha che, contrariamente a quanto affermato nel caso (1), deve essere:

  $$q_{i,right}(t) \leq q_{i-1,left}(t) - 1 + 2\bar{C}(t) - 1$$

quindi

$$q^*(t+1) \leq \frac{q_{i-1,left}(t) - 1 + q_{i-1,left}(t) - 1 + 2\bar{C}(t) - 1}{2} = q_{i-1,left}(t) - 3/2 + \bar{C}(t)$$

$$C_{i-1}(t+1) \leq \left\lfloor q_{i-1,left}(t) - 3/2 + \bar{C}(t) \right\rfloor - q_{i-1,left}(t+1) + 1 < \bar{C}(t)$$

- **caso (5)**: in questo caso si ha che la cella contesa viene assegnata alla telecamera che sta a sinistra

  $C_{i-1}(t+1) = q_{i-1,right}(t+1) - q_{i-1,left}(t+1) + 1 = \lfloor q^*(t+1) \rfloor + 1 - q_{i-1,left}(t+1) + 1$

  dato che in questo caso $q^*(t+1)$ è uguale al caso 4

  $$C_{i-1}(t+1) = q_{i-1,left}(t) + \bar{C}(t) - 2 + 1 - q_{i-1,left}(t+1) + 1 = \bar{C}(t)$$

Consideriamo il caso in cui la videocamera i comunica e provoca lo spostamento del limite sinistro relativo alla videocamera $(i+1)$. Analogamente a quanto fatto in precedenza, si ottiene:

$$q^*(t+1) = \frac{q_{i,left}(t) - 1 + q_{i+1,right}(t)}{2} =$$

$$= \begin{cases} q^*(t+1) \in \mathbb{N} & (1) \\ q^*(t+1) > q_{i,right}(t) \Rightarrow q_{i+1,left}(t+1) = q_{i,right}(t) & (2) \\ q^*(t+1) \notin \mathbb{N} & (3) \end{cases}$$

□

Dopo aver dimostrato che la successione dei massimi non è crescente proviamo a descrivere una strada che porta a dire che esiste sempre una seqienza di comunicazioni che porta a raggiungere l'ottimo. Per fare ciò utiliziamo un approccio induttivo sul numero di videocamere.

Proposizione 4. *L'algoritmo descritto in precedenza porta a partizionare il perimetro discretizzato in maniera ottima.*

Dimostrazione. Come detto in precedenza si procede in maniera induttiva sul numero di videocamere, cioe dimostriamo per dei casi base che esiste una sequenza tale da fare decrescere $\bar{C}(t)$, poi per induzione dimostriamo il caso generale. Nel seeguito ometteremo l'indicazione dell'istante temporale in quanto non è fondamentale in questa situazione.

Caso N=1

È banale.

Caso N=2

Scegliamo, senza nessuna perdita di generalita $C_1 = \bar{C} > C_{OTT}$ cioè la videocamera 1 ha piu celle della seconda. Allora basta che prima comunichi la videocamera 2 e poi la 1 per ottenere una decrescita di $\bar{C}$.

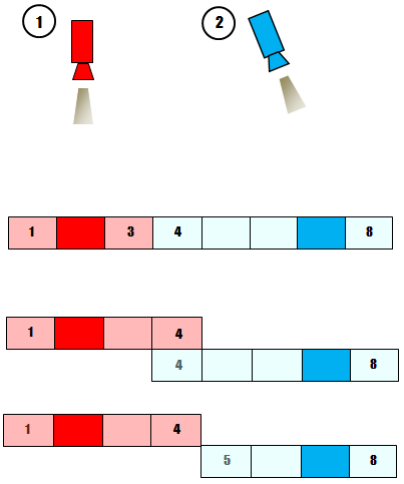


Figura 2.6: Funzionamento algoritmo con N=2. In questo caso prima comunica la videocamera 2 e poi la 1.

Caso N=3

Con riferminento alle figure mostriamo l'esistenza in quel dato caso di una sequenza che riduce $\bar{C}$: il numero totale di celle è 9, quindi essendoci 3 telecamere, si ha che $C_{OTT} = \frac{M}{N} = 3$. In questo caso basta che parlino nell'ordine le telecamere 2,1,3 per raggiungere l'ottimo.

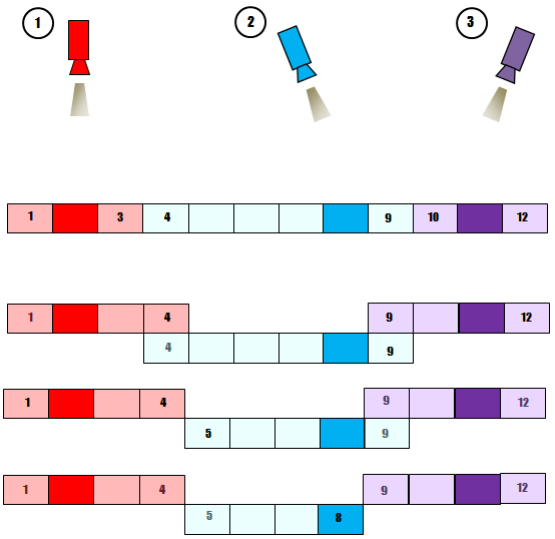


Figura 2.7: Funzionamento algoritmo con N=2. In questo caso l'ordine di comunicazione è 2,1,3.

Di seguito riportiamo un altra possibile sequenza di comunicazione.

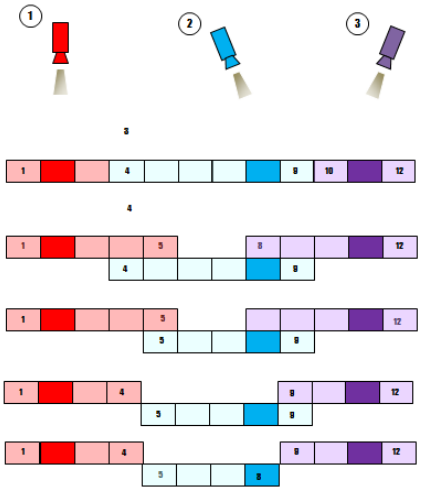


Figura 2.8: Funzionamento algoritmo con N=2. In questo caso l'ordine di comunicazione è 2,1,2,3.

Una volta definiti i casi base passiamo a formulare l'ipotesi induttiva:

supponiamo che l'algoritmo sia tale che, nel caso di N-1 videocamere, se esiste una videocamera tale che $\bar{C}_{N-1} > C_{OTT}(N-1)$ allora esiste una succesione di comunicazioni che fa decrescre $\bar{C}_{N-1}$ fino a quando viene raggiunta l'uguaglianza $\bar{C}_{N-1} = C_{OTT}(N-1)$ e verifichiamo che l'algoritmo funziona anche per N videocamere.

Innanzitutto mettiamo in evidenza un fatto:

Lemma 1. *Possiamo sempre ricondurci al caso di non-overlap tra le sezioni di perimetro assegnate a ciascuna videocamera.*

Dimostrazione. Se due videocamere presentano una situzione di overlap si può facilmene intuire che a seguito di due comunicazioni tra le videocamere, una in un verso e la seconda nel verso opposto la situzione di overlap sparisce.

□

La seconda considerazione riguarda i possibili casi che si possono presentare dipendentemente dalla posizione in cui si trova la videocamera con il maggior numero di celle da patrollare. In particolare si possono consideraretre casi:

- la videocamera è quella posizionata più a sinistra
- la videocamera è quella posizionata più a destra
- la videocamenra si trova in una posizione centrale

I primi due casi sono perfettamente simmetrici mentre per l'ultimo caso si può considerare il fatto che il problema si scompone in due sottoproblemi di complessità inferiore (ovvero il numero di videocamere è più basso). In particolare per l'ultimo caso si può considerare all'inizio il problema formato dalla videocamera con più celle da patrollare e da tutte le videocamere che la precedono e poi, se non è stato ragginuto l'ottimo, quello costituito dalla videocamera con più celle da patrollare e da tutte le videocamere che la seguono.

Fatte queste premesse consideriamo il caso in cui la videocamera con numero massimo di celle da pratrollare sia quella più a sinistra. A questo punto definiamo le grandezze in gioco:

- $\bar{C}(N)$ è il numero di celle che devono essere patrollate dalla videocamera con più celle assegnate, che per ipotesi è la videocamere numero 1. Considerando il caso peggiore si ha che $\bar{C}_N = C_{OTT}(N) + 1$.

$$\bar{C}_{OTT}(N) = \begin{cases} \frac{M}{N} & \text{se M è multiplo N} \\ \lfloor \frac{M}{N} \rfloor + 1 & \text{se M non è multiplo di N} \end{cases}$$

- M_R è la lungheza del sottoperimetro patrollato dalle videocamere 2,3,...,N

- $\bar{C}_R$ è il numero di celle che devono essere patrollate dalla videocamera con più celle assegnate, considerando le videocamenre dalla numero 2 alla N. Dato che per ipotesi induttiva il nostro algoritmo è in grado di partizionare in maniera ottima con N-1 videocamere si ha che:

$$\bar{C}_R = \begin{cases} \frac{M_R}{N-1} & \text{se } M_R \text{ è multiplo N-1} \\ \lfloor \frac{M_R}{N-1} \rfloor + 1 & \text{se } M_R \text{ non è multiplo di N-1} \end{cases}$$

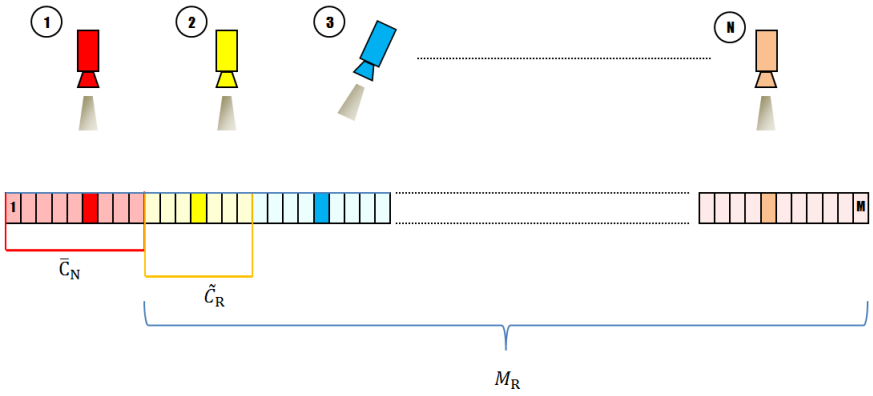


Figura 2.9: Conformazione del periodo in esame

A questo punto dobbiamo studiare tutti i possibili casi che si possono presentare in base alle lunghezze di M e M_R.

Caso 1

Ipotizzo che M sia multiplo di N e cerco le condizioni con le quali garantisco che M_R è multiplo di N-1:

in questo caso, come detto in precedenza, si ha che:

$$C_{OTT}(N) = \frac{M}{N}$$

e di conseguenza

$$\bar{C_N} = C_{OTT}(N) + 1 = \frac{M}{N} + 1$$

ovvero

$$M_R = M - \bar{C}(N) = M - \frac{M}{N} - 1 = \frac{(N-1)M}{N} - 1 = (N-1)(\frac{M}{N} - \frac{1}{N-1})$$

e quindi M_R è multiplo di N-1 se

$$(\frac{M}{N} - \frac{1}{N-1})$$

è un numero intero. Dato che $\frac{M}{N}$ è intero lo deve essere anche $\frac{1}{(N-1)}$ ma questo avviene se e solo se N=2 e si ricade n uno dei casi base.

Caso 2

Ipotizzo che M non sia multiplo di N e cerco le condizioni con le quali garantisco che M_R è multiplo di N-1:

in questo caso si ha che

$$C_{OTT}(N) = \left\lfloor \frac{M}{N} \right\rfloor + 1$$

e di conseguenza

$$\bar{C}_N = C_{OTT}(N) + 1 = \left\lfloor \frac{M}{N} \right\rfloor + 2$$

quindi

$$M_R = M - \bar{C}_N = M - \left\lfloor \frac{M}{N} \right\rfloor - 2 = M - \frac{M}{N} + \frac{MmodN}{N} - 2$$

dove abbiamo usato il fatto che

$$\left\lfloor \frac{M}{N} \right\rfloor = \frac{M}{N} - \frac{MmodN}{N}$$

a questo punto

$$M_R = M - \frac{M}{N} + \frac{MmodN}{N} - 2 = (N-1)\frac{M}{N} + \frac{MmodN}{N} - 2 =$$

$$= (N-1)\left(\left\lfloor \frac{M}{N} \right\rfloor + \frac{MmodN}{N}\right) + \frac{MmodN}{N} - 2 = (N-1)\left\lfloor \frac{M}{N} \right\rfloor + MmodN - 2 =$$

$$= (N-1)\left(\left\lfloor \frac{M}{N} \right\rfloor + \frac{MmodN}{N-1} - \frac{2}{N-1}\right)$$

e quindi M_R è un multiplo di N-1 se e solo se

$$\left(\left\lfloor \frac{M}{N} \right\rfloor + \frac{MmodN}{N-1} - \frac{2}{N-1}\right) \in \mathbb{N}$$

ovvero se e solo se

$$\left(\frac{MmodN}{N-1} - \frac{2}{N-1}\right) \in \mathbb{N}$$

e quindi $MmodN - 2$ deve essere un multiplo di N-1. Dato che

$$1 \leq MmodN \leq N - 1$$

si ha che la relazione precedente è valida se e solo se

$$MmodN = 2$$

Riassumendo, se N>2 posso incontrare i seguenti casi:

- *M è multiplo di N, M_R non è multiplo di N-1*
- *M non è multiplo di N, M_R è multiplo di N-1* $\Longleftrightarrow LmodN = 2$
- *M non è multiplo di N, M_R non è multiplo di N-1*

Come illustrato in precedenza, nel passo induttivo la prima videocamera inizia a pattugliare i $\bar{C}_N = C_{OTT}(N) + 1$ quanti perimetrali disposti presso il bordo sinistro. Si dimostra che indipendentemente dalle lunghezze M e M_R e dal numero di telecamere, $\bar{C}_N$ decresce convergendo al valore ottimo. Assumeremo l'ipotesi non restrittiva che la seconda videocamera sia sempre quella con il minor numero di celle da patrollare; questa ipotesi non è restrittiva in quanto esiste con probabilità 1 una sequenza di comunicazioni che ci porta in tale situazione. Il punto fondamentale su cui si basa la dimostrazione è il seguente:

partendo dal caso peggiore in cui $\bar{C}_N = C_{OTT}(N) + 1$, si deve dimostrare che in ognuno dei tre casi elencati in precedenza vale:

$$\bar{C}_N \geq \tilde{C}_R + 2$$

dove $\tilde{C}_R$rappresenta il numero di celle patrollate dalla videocamera 2. In questo modo si garantisce che sicuramente $\bar{C}_N$ decresce e converge all'ottimo quando le telecamere nelle posizioni 1 e 2 comunicano una di seguito all'altra.

Caso 1
M è multiplo di N, M_R non è multiplo di N-1
Sotto queste ipotesi valgono le seguenti relazioni:

- $C_{OTT} = \frac{M}{N}$
- $\bar{C}_N = C_{OTT} + 1 = \frac{M}{N} + 1$
- $C_{OTT}(N-1) = \left\lfloor \frac{M_R}{N-1} \right\rfloor + 1$
- $\tilde{C}_R = C_{OTT}(N-1) - 1 = \left\lfloor \frac{M_R}{N-1} \right\rfloor$dato che M_R non è multiplo di N-1

La disuguaglianza

$$\bar{C}_N \geq \tilde{C}_R + 2$$

viene riscritta nella forma equivalente:

$$\frac{M}{N} + 1 \geq \left\lfloor \frac{M_R}{N-1} \right\rfloor + 2$$

$$\frac{M}{N} \geq \left\lfloor \frac{M_R}{N-1} \right\rfloor + 1$$

sostituendo $M_R = M - \bar{C}_N$

$$\frac{M}{N} \geq \left\lfloor \frac{M - \bar{C}_N}{N-1} \right\rfloor + 1 = \left\lfloor \frac{M - \frac{M}{N} - 1}{N-1} \right\rfloor + 1 = \left\lfloor \frac{M}{N} - \frac{1}{N-1} \right\rfloor + 1$$

si ottiene che:

$$\bar{C}_N \geq \tilde{C}_R + 2 \Longleftrightarrow \frac{M}{N} \geq \left\lfloor \frac{M}{N} - \frac{1}{N-1} \right\rfloor + 1$$

dato che $\frac{1}{N-1} < 1$ si ha che $\left\lfloor \frac{M}{N} - \frac{1}{N-1} \right\rfloor = \frac{M}{N} - 1$

$$\bar{C}_N \geq \tilde{C}_R + 2 \Longleftrightarrow \frac{M}{N} \geq \frac{M}{N}$$

sempre vera.

Caso 2
M non è multiplo di N, M_R è multiplo di N-1
Sotto queste ipotesi valgono le seguenti relazioni:

- $MmodN = 2$
- $C_{OTT}(N) = \lfloor \frac{M}{N} \rfloor + 1$
- $\bar{C}_N = C_{OTT}(N) + 1 = \lfloor \frac{M}{N} \rfloor + 2$
- $C_{OTT}(N-1) = \frac{M_R}{N-1}$
- $\tilde{C}_R = C_{OTT}(N-1) = \frac{M_R}{N-1}$

In tal caso, essendo M_R suddiviso in intervalli di ugual dimensione, non è necessario fare delle ipotesi a priori sul segmento pattugliato dalla seconda telecamera.

$$\bar{C}_N = \left\lfloor \frac{M}{N} \right\rfloor + 2 = \frac{M}{N} - \frac{MmodN}{N} + 2$$

$$\tilde{C}_R = \frac{M_R}{N-1} = \frac{M - \bar{C}_N}{N-1} = \frac{M - \lfloor \frac{M}{N} \rfloor - 2}{N-1} = \frac{M}{N-1} - \frac{\frac{M}{N}}{N-1} + \frac{\frac{MmodN}{N}}{N-1} - \frac{2}{N-1} =$$

$$= \frac{(N-1)M}{N(N-1)} + \frac{MmodN}{N(N-1)} - \frac{2}{N-1} = \frac{M}{N} + \frac{MmodN}{N(N-1)} - \frac{2}{N-1}$$

Si riscrive la disuguaglianza $\bar{C}_N \geq \tilde{C}_R + 2$ utilizzando le espressioni equivalenti di $\tilde{C}_R$ e $\bar{C}_N$:

$$\frac{M}{N} - \frac{MmodN}{N} + 2 \geq \frac{M}{N} + \frac{MmodN}{N(N-1)} - \frac{2}{N-1} + 2$$

semplificando i termini comuni nei due membri:

$$-\frac{MmodN}{N} \geq \frac{MmodN}{N(N-1)} - \frac{2}{N-1}$$

si ottiene:

$$\frac{2}{N-1} \geq \frac{MmodN}{N(N-1)} + \frac{MmodN}{N} = \frac{(1+N-1)(MmodN)}{N(N-1)}$$

Di conseguenza risulta:

$$\bar{C}_N \geq \tilde{C}_R + 2 \iff \frac{2}{N-1} \geq \frac{MmodN}{N-1} \iff 2 \geq MmodN$$

Essendo $MmodN = 2$ per ipotesi, allora $\bar{C}_N$ supera sempre di almeno due unità il valore di $\tilde{C}_R$ e la convegenza all'ottimo è assicurata.

Caso 3
M non è multiplo di N, M_R non è multiplo di N-1
Sotto queste ipotesi valgono le seguenti relazioni:

- $C_{OTT}(N) = \lfloor \frac{M}{N} \rfloor + 1$
- $\bar{C}_N = C_{OTT}(N) + 1 = \lfloor \frac{M}{N} \rfloor + 2$
- $C_{OTT}(N-1) = \left\lfloor \frac{M_R}{N-1} \right\rfloor + 1$
- $\tilde{C}_R = C_{OTT}(N-1) - 1 = \left\lfloor \frac{M_R}{N-1} \right\rfloor$

$$\bar{C}_N \geq \tilde{C}_R + 2 \Longleftrightarrow \frac{M}{N} + 2 \geq \left\lfloor \frac{M_R}{N-1} \right\rfloor + 2 \Longleftrightarrow \left\lfloor \frac{M}{N} \right\rfloor \geq \left\lfloor \frac{M_R}{N-1} \right\rfloor$$

data l'implicazione:

$$\frac{M}{N} \geq \frac{M_R}{N-1} \Rightarrow \left\lfloor \frac{M}{N} \right\rfloor \geq \left\lfloor \frac{M_R}{N-1} \right\rfloor$$

se si prova che la disuguaglianza al primo membro vale sempre, allora anche quella al secondo membro è sempre vera e di conseguenza $\bar{C}_N \geq \tilde{C}_R + 2$. Sostituendo $L_R = L - \bar{C}_N$:

$$\frac{M}{N} \geq \frac{M - \bar{C}_N}{N-1} = \frac{M - \lfloor \frac{M}{N} \rfloor - 2}{N-1}$$

e sviluppando la parte intera:

$$\frac{M}{N} \geq \frac{M}{N-1} - \frac{\frac{M}{N}}{N-1} + \frac{\frac{MmodN}{N}}{N-1} - \frac{2}{N-1}$$

$$\frac{M}{N} \geq \frac{M}{N} + \frac{MmodN}{N(N-1)} - \frac{2}{N-1}$$

si ottiene la relazione:

$$\frac{\frac{MmodN}{N}}{N-1} < \frac{2}{N-1}$$

che ha valenza generale essendo$1 \leq MmodN \leq N-1$.

Di conseguenza risulta:

$$\frac{M}{N} \geq \frac{M}{N-1}\ vera \implies \left\lfloor \frac{M}{N} \right\rfloor \geq \left\lfloor \frac{M_R}{N-1} \right\rfloor\ vera \implies \bar{C}_N \geq \tilde{C}_R + 2\ vera$$

e la convergenza all'ottimo è dimostrata anche in quest'ultimo caso.

□

Capitolo 3

Tracking degli eventi

Il sistema di videosorveglianza considerato in questo progetto integra la funzione di pattugliamento perimetrale, illustrata nei capitoli precedenti, attraverso un'azione di *tracking*, eseguita in modo autonomo dalle telecamere, per riconoscere ed inseguire un target di interesse all'interno delle rispettive aree di competenza; in questo modo le telecamere possono individuare la presenza di un intruso e tenere traccia dei relativi spostamenti lungo il percorso sorvegliato. Nello specifico, in questo contesto non si approfondisce il problema dell'identificazione visiva dell'evento, trattato per esempio in [5], ma viene presentata una strategia specifica per inseguire il target con le telecamere senza mai perderlo di vista. Il sistema proposto si basa sull'utilizzo del filtro di Kalman e sull'adozione di uno schema distribuito per coordinare il passaggio del target da una camera all'altra, trovando un giusto compromesso tra le esigenze del patrolling e del tracking.

3.1 Modellizzazione dell'evento

Prima di illustrare nel dettaglio la soluzione proposta per il tracking si definisce l'evento che le telecamere hanno il compito di inseguire. In generale, per *evento* si intende qualcosa che ha una dinamica nello spazio e nel tempo di cui il sistema deve accorgersi il prima possibile; nello specifico, si suppone valgano le seguenti ipotesi:

1. l'evento è uniformemente distribuito nello spazio L, vale a dire la probabilità che si verifichi è la stessa in ogni punto del perimetro;
2. le dimensioni dell'evento non sono puntiformi ma tali da occupare almeno un quanto elementare del perimetro discretizzato;
3. la comparsa dell'evento è rara, nel senso che è improbabile che lungo il perimetro vi siano contemporaneamente due target da inseguire; nell'implemetazione del sistema e nelle simulazioni riguardanti il tracking, si suppone che la probabilità di avere più eventi nello stesso istante sia nulla.

Un approccio comune per il tracking è quello di modellizzare il movimento incognito del target considerandone la posizione e la velocità nel tempo; in particolare, sapendo di utilizzare il filtro di Kalman, il modello viene rappresentato in spazio di stato ottenendo il seguente sistema lineare del secondo ordine, pilotato da rumore bianco:

$$\begin{aligned} x(t+1) &= \begin{bmatrix} 1 & 1 \\ 0 & a \end{bmatrix} x(t) + \begin{bmatrix} 0 & 0 \\ b & 0 \end{bmatrix} \begin{bmatrix} n(t) \\ m(t) \end{bmatrix} \\ y(t) &= \begin{bmatrix} 1 & 0 \end{bmatrix} x(t) + \begin{bmatrix} 0 & d \end{bmatrix} \begin{bmatrix} n(t) \\ m(t) \end{bmatrix} \end{aligned}$$

Lo stato $x(t) = \begin{bmatrix} p(t) & v(t) \end{bmatrix}^T$ è rappresentato dalla posizione e dalla velocità dell'evento; la posizione futura è funzione della posizione e della velocità attuali:

$$p(t+1) = p(t) + v(t)$$

mentre la velocità viene disturbata da un rumore bianco gaussiano $n(t)$ che le conferisce un andamento aleatorio e le consente di cambiare anche di segno, facendo si che l'evento si diriga verso un estremo o l'altro del perimetro:

$$v(t+1) = a\,v(t) + b\,n(t)$$

Per finire, l'uscita $y(t)$ considera unicamente la posizione perché le telecamere non sono in grado di misurare la velocità dell'evento ma si limitano a stimarla. Il parametro $a \approx 1$ modella l'inerzia del target (a seconda che sia maggiore o minore di 1 l'evento viene accelerato o frenato), mentre b e d regolano l'intensità dei rumori di modello e di misura, rispettivamente $n(t)$ e $m(t)$ che sono supposti gaussiani, incorrelati e di varianza unitaria.

3.2 Filtro di Kalman

Una volta definito il modello dell'evento, si implementa il filtro di Kalman utilizzando le equazioni standard per gli stimatori a minima varianza d'errore dello stato:

- **Stima a priori** (*aggiornamento temporale*)

$$\hat{x}(t+1 \mid t) = A\,\hat{x}(t \mid t)$$
$$P(t+1 \mid t) = A\,P(t \mid t)\,A^T \;+\; Q$$

- **Stima a posteriori** (*aggiornamento rispetto alle misure*)

$$\hat{x}(t+1 \mid t+1) = \widehat{x}(t+1 \mid t) \;+\; L(t+1)\,[y(t+1) \;-\; C\,\hat{x}(t+1 \mid t)]$$
$$P(t+1 \mid t+1) = P(t+1 \mid t) \;-\; L(t+1)\,C\,P(t+1 \mid t)$$

dove si utilizza il *guadagno del filtro*:

$$L(t+1) \;\triangleq\; P(t+1 \mid t)\,C^T\,(C\,P(t+1 \mid t)\,C^T \;+\; R)^{-1}$$

- **Condizioni iniziali**

$$\hat{x}(0 \mid -1) \;=\; \mu_0$$
$$P(0 \mid -1) \;=\; P_0$$

Per quanto riguarda la definizione della condizione iniziale $\hat{x}(0 \mid -1)$, se da un lato alla variabile che descrive la posizione dell'evento $\hat{p}(0 \mid -1)$ è possibile assegnare il punto del perimetro corrispondente al primo avvistamento, dall'altro non si dispongono delle informazioni necessarie per stabilire un valore corretto della velocità iniziale $\hat{v}(0 \mid -1)$; in effetti le telecamere non sono in grado di valutare l'entità della velocità istantanea di un oggetto in movimento. Per questo motivo, viene adottata la scelta progettuale di utilizzare la velocità della telecamera in tracking come stima arbitraria della velocità iniziale e di associare ad essa una elevata incertezza attraverso una definizione appropriata della varianza; per esempio:

$$P(0 \mid -1) \;=\; E\begin{bmatrix}\widehat{p}(0 \mid -1)\\ \hat{v}(0 \mid -1)\end{bmatrix}\;\begin{bmatrix}\widehat{p}(0 \mid -1) & \hat{v}(0 \mid -1)\end{bmatrix} \;=\; \begin{bmatrix}10 & 0\\ 0 & 1000\end{bmatrix}$$

Naturalmente, la strategia adottata per bypassare il problema della velocità incognita introduce inevitabilmente delle ricadute negative sul problema di stima: per esempio, in corrispondenza dei primi campioni si ha un aumento dell'errore di predizione. Tuttavia, quando si supera il transitorio iniziale, gli errori diminuiscono e l'inseguimento del target procede con una precisione adeguata.

3.3 Strategia per il tracking

Come accennato in precedenza, il tracking viene progettato in modo tale da consentire al sistema di videosorveglianza di inseguire il target senza interrompere l'azione di pattugliamento, che è indispensabile per intercettare la comparsa degli eventi futuri. Un compromesso tra queste due attività, tra loro contrastanti, viene raggiunto coordinando in modo distribuito l'azione delle telecamere; per esempio, quando una telecamera entra in tracking ed interrompe l'attività di sorveglianza perché l'inseguimento dell'evento ha una priorità più alta, comanda alle telecamere adiacenti di ereditare il controllo della sua zona, estendendo fino ai limiti fisici i bordi delle relative regioni. La strategia adottata consiste nell'ottimizzare le prestazioni del sistema facendo collaborare tra loro le telecamere; nello specifico, si procede in questo modo:

1. ciascuna telecamera entra in modalità tracking ed interrompe l'azione di patrolling ogni qual volta si manifesti un evento sul tratto di perimetro sorvegliato (data l'ipotesi sulle dimensioni non puntiformi del target è sufficiente che una sua parte occupi il quanto controllato dalla telecamera);

2. quando una telecamera inizia ad inseguire un evento comunica la sua nuova attività ai vicini che reagiscono assumendosi l'onere di sorvegliare la regione non monitorata estendendo il loro raggio d'azione fino ai limiti fisici;

3. una telecamera in fase di inseguimento comunica ai vicini il tempo stimato nel quale l'evento raggiungerà i bordi fisici;

4. le telecamere adiacenti esaminano le informazioni ricevute e le confrontano con i tempi necessari per raggiungere il bordo, allo scopo di decidere se continuare il patrolling o interrompere l'attività di sorveglianza e raggiungere in tempo il confine per iniziare l'inseguimento del target;

5. le telecamere in tracking comunicano con i vicini solo in istanti aleatori, seguendo lo schema di comunicazione randomizzato definito all'inizio del progetto; nel capitolo dedicato alle simulazioni, viene implementato anche un processo di comunicazione deterministico che consente alle telecamere in tracking di comunicare in modo continuo con i vicini, migliorando ulteriormente le prestazioni del sistema.

La strategia adottata garantisce che l'evento non venga mai perso ed ottimizza il funzionamento complessivo del sistema perché le telecamere sospendono l'azione di pattugliamento solamente quando l'evento inizia ad avvicinarsi al bordo della loro regione. Lo schema proposto ha anche il merito di rendere consistente l'attività di tracking; per esempio, se si considera il caso sfavorevole in cui il movimento del target si sviluppa nei pressi dell'estremità comune a due regioni di sorveglianza, le regole prese impongono ad entrambe le telecamere di inseguire l'evento, riducendo in questo modo la probabilità di perdere il target qualora subisse uno spostamento improvviso verso la zona di una delle due telecamere.

Inoltre, l'aggiornamento delle sezioni di copertura prosegue in modo distribuito secondo la teoria sviluppata nei capitoli precedenti, garantendo così delle ottime prestazioni.

3.4 Opposizione di fase

Al fine di minimizzare la probabilità di perdere un evento o se non viene perso, il tempo impiegato per intercettarlo, si è deciso, verso la fine del progetto, di regolare il movimento delle telecamere sincronizzandole in opposizione di fase. In particolare, per non compromettere il corretto funzionamento del sistema, la messa in fase viene indotta con un meccanismo a priorità inferiore rispetto all'esecuzione delle altre attività, come ad esempio il tracking ed il pattugliamento del perimetro. Nello specifico, si è deciso di sincronizzare le telecamere agendo sulla loro velocità, rallentando quelle in anticipo di fase e lasciando inalterate le altre, visto che non è possibile aumentare oltre il limite massimo le velocità delle camere in ritardo. Naturalmente, un intervento di questo tipo produce inesorabilmente delle ricadute negative sul tempo di pattugliamento ottimo; in effetti, la diminuzione di velocità di alcune telecamere aumenta il tempo massimo di non visita ai punti del perimetro. Tuttavia, il peggioramento indotto sull'azione di patrolling non è problematico e coinvolge solo una piccola parte dell'intero percorso di videosorveglianza; in ogni caso, un rilassamento delle prestazioni ottime è ammissibile se si vuole scongiurare l'eventualità di perdere irrimediabilmente un evento. Nell'implementazione della messa in fase, vengono presi in considerazione i tempi di raggiungimento dei bordi fisici, ricavati in modo autonomo dalle camere stesse, utilizzando le conoscenze sulla loro velocità e sugli estremi di pattugliamento.

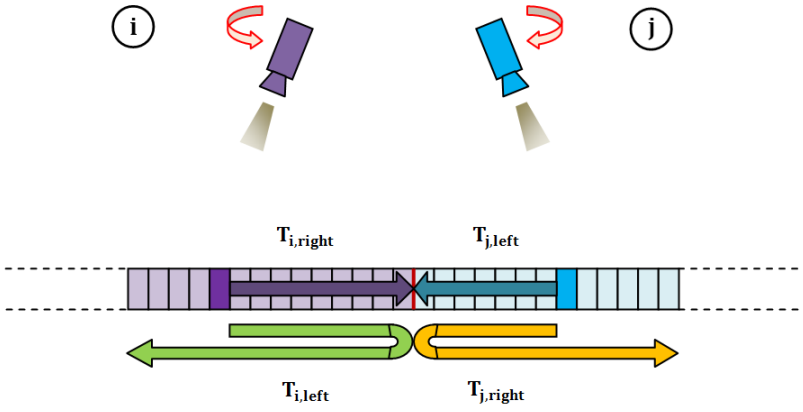


Figura 3.1: Tempi di raggiungimento dei bordi fisici come misura della fase

Dunque, la camera i-esima calcola dei valori di fase di questo tipo:

- se $V_i > 0$ cioè se $z_i(t+1) - z_i(t) > 0$

$$\begin{aligned} T_{i,right} &= \frac{q_{i,right}(t) - z_i(t)}{v_i} \\ T_{i,left} &= T_{i,right} + T_A \end{aligned}$$

- altrimenti, se $V_i < 0$ cioè se $z_i(t+1) - z_i(t) < 0$

$$\begin{aligned} T_{i,left} &= \frac{z_i(t) - q_{i,left}(t) + 1}{v_i} \\ T_{i,right} &= T_{i,left} + T_A \end{aligned}$$

dove $T_{i,left}$ e $T_{i,right}$ sono i tempi di arrivo della telecamera sui bordi sinistro e destro della sezione di copertura, mentre T_A è il tempo di percorrenza dell'intero segmento A_i:

$$T_A = \frac{q_{i,right}(t) - q_{i,left}(t) + 1}{v_i}$$

Si osserva che la fase è definita in modo tale da distinguere il caso in cui la telecamera stia andando in una direzione o nell'altra.

Le camere adiacenti si scambiano l'informazione sulla fase e se i tempi di raggiungimento dei bordi differiscono, decidono di applicare la regolazione di velocità. Per esempio, la camera i-esima in figura è in anticipo di fase rispetto alla j-esima quando sono verificate entrambe le disuguaglienze:

$$\begin{aligned} T_1 &= |T_{i,right} - T_{j,left}| > 0 \\ T_2 &= |T_{i,left} - T_{j,right}| > 0. \end{aligned}$$

In tal caso, la camera i-esima determina lo scostamento minore tra quelli esaminati, vale a dire:

$$T = min(T_1, T_2)$$

e rallenta la propria velocità di un fattore ad esso proporzionale. La scelta di correggere la velocità in modo proporzionale allo sfasamento e di utilizzare un piccolo fattore, permette al sistema di eseguire la messa in fase in modo graduale, evitando di penalizzare eccessivamente il patrolling. Inoltre, la strategia seguita consente di operare in modo corretto la messa in fase delle telecamere anche nel caso di comunicazione sporadica, come è possibile riscontrare nel paragrafo riservato alle simulazioni.

Capitolo 4

Implementazione

L'implementazione del sistema di videosorveglianza esaminato nei capitoli precedenti ed in particolare, degli algoritmi di comunicazione asincroni utilizzati dalle telecamere per eseguire in modo distribuito le attività di patrolling e di tracking, si basano sull'utilizzo della suite Mathworks MATLAB R2012b. Questo programma non rappresenta necessariamente l'ambiente di sviluppo migliore, soprattutto se si considera la parte grafica nella simulazione del sistema; sotto questo punto di vista, esistono delle librerie C complete e molto veloci che consentono di ottenere delle prestazioni superiori. Ciò nonostante Matlab è in grado di offrire una maggiore flessibilità ed immediatezza d'uso rispetto agli altri programmi e proprio queste sue caratteristiche hanno spinto i progettisti ad adottarlo, consapevoli del fatto che l'obiettivo principale del lavoro intrapreso non riguarda tanto l'aspetto grafico ma la dimostrazione della correttezza degli algoritmi proposti. Naturalmente, una decisione di questo tipo non vuole negare l'importanza di esporre in modo chiaro ed efficace i risultati ottenuti; a tal proposito, Matlab fornisce degli strumenti software specifici, come il pacchetto Mathworks Simulink, usato nella realizzazione di alcuni video in cui si simulano tutte le azioni delle telecamere, compreso l'inseguimento degli eventi.

La caratteristica peculiare del sistema di telecamere che si propone di realizzare è la capacità di calcolo distribuita e non centralizzata; per simulare questa indipendenza da ogni forma di controllo esterno ed accentrato, si è deciso di adottare per la stesura del codice il paradigma di programmazione orientato agli oggetti che si conforma in modo soddisfacente allo scenario considerato. Inoltre, l'orientamento agli oggetti permette di avere:

1. **riusabilità** del codice e facilità di manutenzione: le classi definite si possono riutilizzare, eventualmente estendendole per creare nuove classi (quindi è più facile apportare modifiche);

2. **robustezza**, cioè capacità di continuare l'esecuzione in presenza di errori. I casi anomali vengono gestiti con il meccanismo delle eccezioni così ogni qual volta si verifica un problema viene lanciata un'eccezione; il codice la intercetta e fornisce delle operazioni da eseguire in quel caso;

3. **affidabilità**: si può sempre verificare che siano soddisfatte le condizioni per cui un oggetto appartiene ad una classe. Per esempio, se una classe definisce una camera, si può verificare nel costruttore che la velocità istantanea sia sempre inferiore a quella massima.

Adottando la programmazione ad oggetti, ciascuna telecamera del sistema viene definita come l'istanza di una classe *Camera*, caratterizzata da un set di attributi che ne definisce lo stato, come ad esempio la velocità di patrolling e gli estremi del field of view e da alcuni metodi che ne determinano il comportamento; essi descrivono le operazioni che la telecamera può eseguire (in piena autonomia) dialogando anche con gli altri oggetti della classe di appartenenza. Nello specifico, queste funzioni sono formate da una serie di istruzioni in codice Matlab che, una volta eseguite, secondo la forma tradizionale della programmazione imperativa, consentono di implementare in modo distribuito le azioni di patrolling e di tracking esaminate nei paragrafi precedenti. Le principali procedure della classe *Camera* sono:

- $y = evoluzione(Camera\ C)$: determina l'evoluzione temporale della camera C tenendo conto dei vincoli fisici di copertura, degli aggiornamenti apportati al proprio fov e della sua eventuale entrata in modalità tracking. Il metodo consente anche di regolare la velocità della camera C nel caso in cui sia in anticipo di fase rispetto a quella precedente o seguente, per sincronizzarla in opposizione di fase. In entrambi i casi, il codice utilizza le informazioni fornite dal parametro in ingresso per modificare un singolo attributo della classe, rispettivamente la *posizione* e la *velocità*. Infine, restituisce la posizione del perimetro che la telecamera sta sorvegliando;

- $aggiorna(Camera\ C_1,\ Camera\ C_2)$:utilizza le regole dell'algoritmo di partizionamento ottimo per aggiornare la sezione di copertura della telecamera che riceve la comunicazione, tra quelle fornite in ingresso; in relazione a quanto esposto nel capitolo sul pattugliamento perimetrale, la funzione calcola il valore q* ed i tempi di patrolling delle due camere servendosi dei rispettivi attributi, in particolare considerando gli estremi di patrolling e le velocità. Inoltre, nel caso in cui i tempi considerati siano identici, l'aleatorietà nell'assegnazione del quanto viene indotta usando come argomento di una istruzione $if - else$ la condizione $rand(1) > 0,5$. La funzione interviene anche per regolare l'azione di tracking; nello specifico, se una delle due telecamere sta inseguendo un evento che si avvicina sempre più al confine con l'altra camera, agisce sulla velocità di quest'ultima per muoverla verso il bordo ed evitare di farle perdere il target;

- $[Ts\ Td] = delayTime(Camera\ C)$: calcola per la telecamera C i tempi di arrivo sui bordi sinistro e destro del segmento sorvegliato, prendendo in considerazione la velocità e la direzione del suo movimento; questi valori vengono usati dagli altri metodi della classe, come, ad esempio, la funzione *evoluzione* che li impiega per valutare la differenza di fase di due camere adiacenti;

Per completare l'implementazione del sistema di videosorveglianza, vengono definite anche le classi *KalmanFilter* ed *Evento*. La prima è costituita essenzialmente da un metodo (se non si considera anche il costruttore) che utilizza le equazioni standard del filtro di Kalman per restituire una stima della posizione e della velocità di un evento, nel futuro; ogni camera, per implementare l'azione di tracking memorizza in un attributo un'istanza di questa classe. La seconda, invece, implementa l'evento da inseguire, descritto dal modello di stato visto nel capitolo precedente.

Di seguito vengono presentate le classi e le relazioni tra le classi in notazione UML (Unified Modeling Language). Il simbolo di una classe è un rettangolo diviso in tre sezioni orrizzontali: la prima contiene il nome, quella centrale elenca gli attributi e la terza i metodi. Ogni metodo elenca i parametri in ingresso e se restituisce un valore viene seguito da : *Classe* per indicare la classe del valore di ritorno. Agli attributi ed ai metodi viene premesso un prefisso di visibilità che può essere pubblico (+) o privato (-) se la visibilità è limitata alla sola classe di appartenenza. Le relazioni tra le classi sono tutte delle *associazioni* che rappresentano un legame tra le istanze delle classi; l'associazione è rappresentata da una linea continua alle cui estremità vengono riportate le molteplicità con cui le classi partecipano alla relazione. Per esempio, una Camera può usare diversi filtri di Kalman, uno per ogni evento da inseguire (anche se in pratica ne usa solo uno per l'ipotesi sulla rarità dell'evento), pertanto si associa alla classe KalmanFilter con una molteplicità 1..* che significa *uno o più di uno*. Nel diagramma UML viene riportata la classe fittizia *Sistema di telecamere*, non utilizzata direttamente nel programma, per evidenziare come l'intero sistema di videosorveglianza sia ottenuto mettendo insieme le azioni di tutte le N camere; la relazione con la classe Camera è una *composizione* rappresentata da una linea di associazione con un rombo nero dal lato dell'oggetto composto.

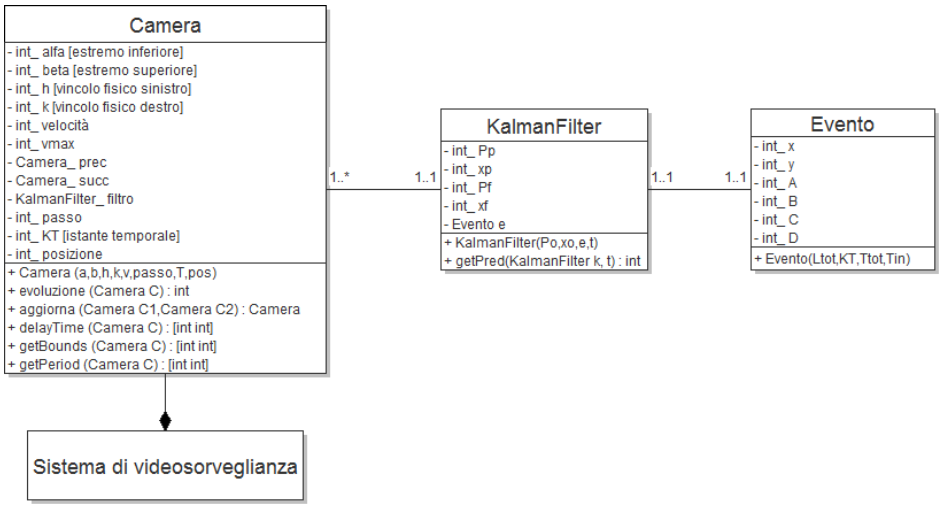


Figura 4.1: Diagramma UML del sistema di videosorveglianza

Capitolo 5

Simulazioni

In questo capitolo proponiamo alcune simulazioni per mostrare il comportamento del nostro sistema in diverse situazioni e al variare di alcuni parametri caratteristici. Quando non espressamente dichiarato, si utlizza $\mu = 5s$, che rappresenta la media dei tempi di interarrivo di ogni singola telecamera che sono modellati con una variabile aleatoria esponenziale negativa. Nel cd in allegato con la relazione sono presenti anche i video delle simulazioni tridimensionali relativi alle simulazioni qui riportate; le simulazioni sono state ottenute utilizzando il pacchetto di Simulink "Simulink 3D animation".

5.1 Variazione del rate di comunicazione

In questo esempio varieremo il parametro $\mu = \frac{1}{\lambda}$, che rappresenta la media dei tempi di interarrivo che sono modellati con una variabile aleatoria esponenziale negativa. Le immagini seguenti mostrano l'evoluzione dei periodi di patrolling delle varie videocamere.

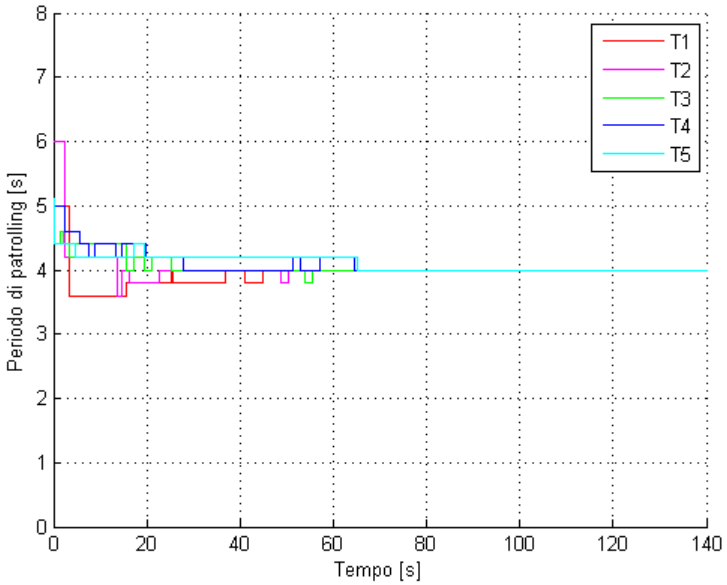


Figura 5.1: $\mu = 3s$

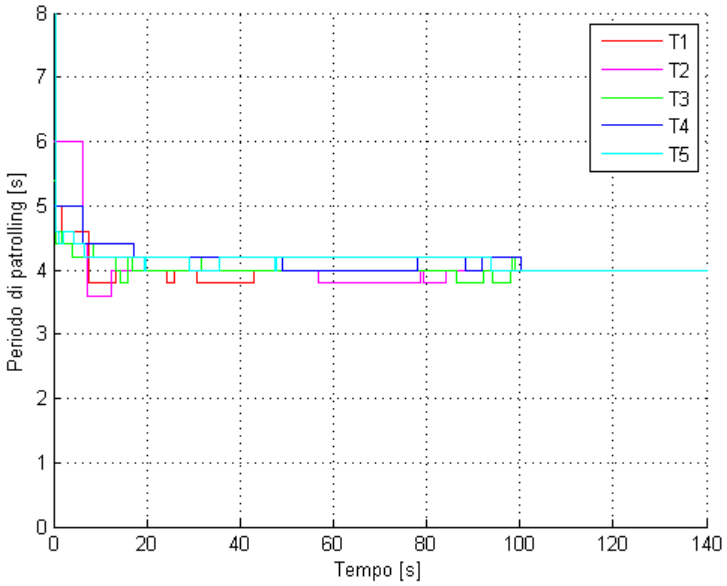


Figura 5.2: $\mu = 5s$

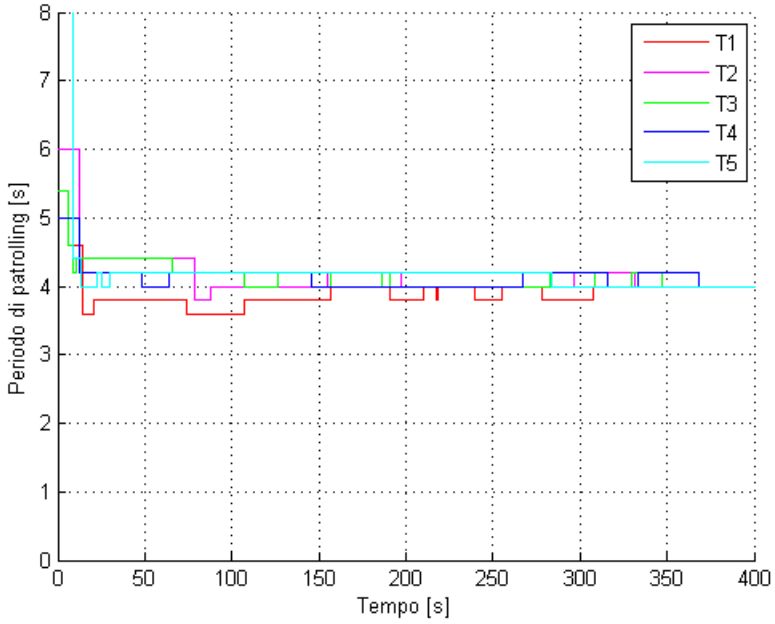


Figura 5.3: $\mu = 15s$

Si vede dalle figure che, nonostante l'aleatorietà del sistema, i tempi con cui l'algoritmo di partizionamento raggiunge l'ottimo decresce all'aumentare del rate di comunicazione come d'altra parte ci si poteva aspettare.

5.2 Perdita di pacchetti

In questa simulazione si vuole vedere l'effetto di una possibile perdita di pacchetti. Le possibili situazioni di perdita di pacchetti possono riguardare:

- Nessuna perdita di pachetti con probabilità p_1
- La videocamera a sinistra perde il pacchetto $\frac{1-p_1}{3}$
- La videocamera a destra perde il pacchetto $\frac{1-p_1}{3}$
- Entrambe le videocamere perdono il pacchetto. Questo caso, a differenza di quelli precedenti, in pratica è come non ci fosse stata nessuna comunicazione $\frac{1-p_1}{3}$.

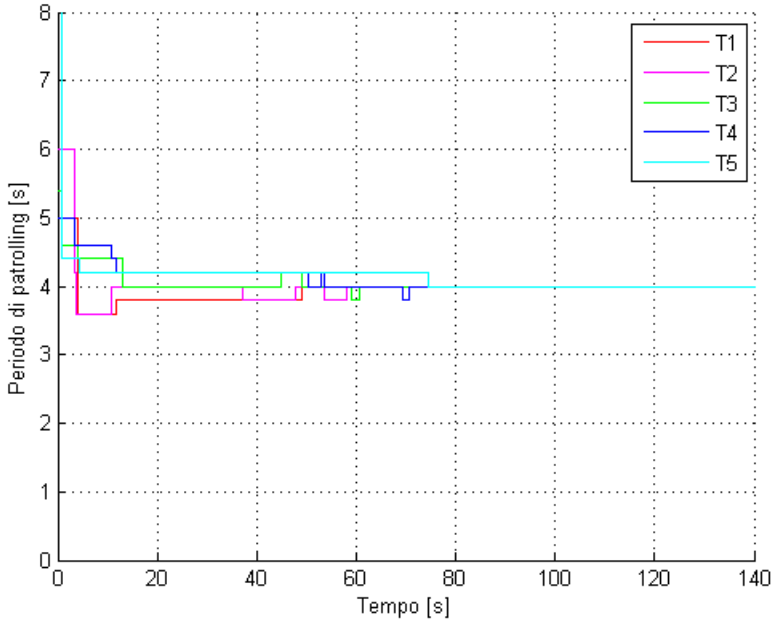


Figura 5.4: $p_1 = 0.8$

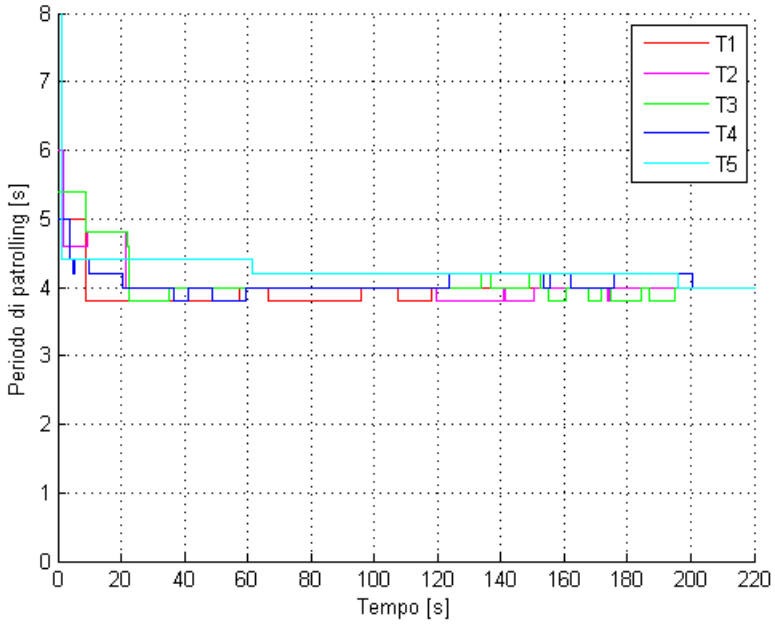


Figura 5.5: $p_1 = 0.5$

Queste simulazioni sono state effettuate con valori di probabilità di errore durante una comunicazione non verosimili per i protocolli di comunicazioni attuali che solitamente sono molto inferiori all' 1% . Siamo stati costretti a mettere tali valori per poter avere differenze apprezzabili. Si può osservare immediatamente che con una probabilità di perdita di pacchetto pari al 20% si ottengono prestazioni paragonabili al caso in cui non ci sono errori nella trasmissione (cfr. Figura 5.2).

5.3 Simulazione telecamere con velocità diverse presenza di vincoli fisici

In questa simulazione vogliamo far vedere come viene suddiviso il perimetro nel caso in cui le videocamere abbiano velocità massime nettamente diverse. La condizione iniziale è stata posta volutamente molto distante dall'ottimo in modo di poter analizzare in maniera più soddisfacente in che modo vengono modificati. Si può notare che alla videocamera 3, che ha la velocità maggiore, viene assegnata la porzione di perimetro più alta. Le velocità delle videocamere vengono inoltre gestite per raggiungere e mantenere l'opposizione di fase.

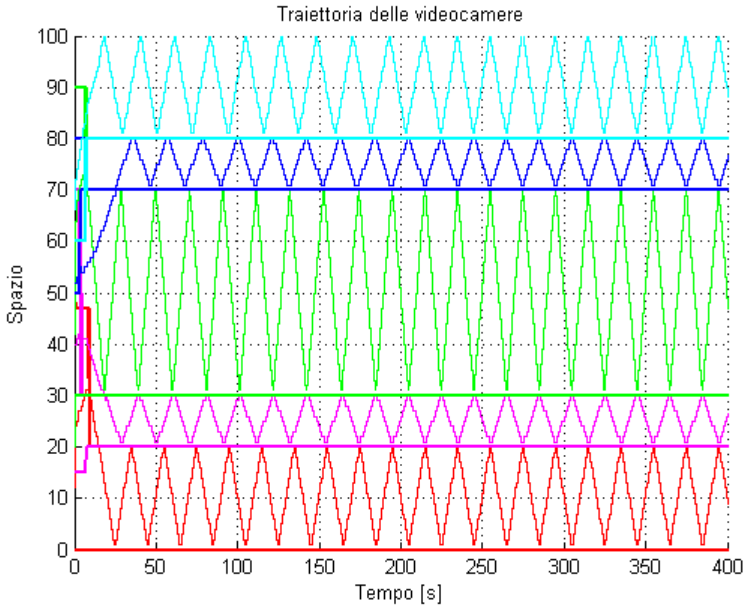


Figura 5.6: Dinamica delle videocamere con velocità diverse

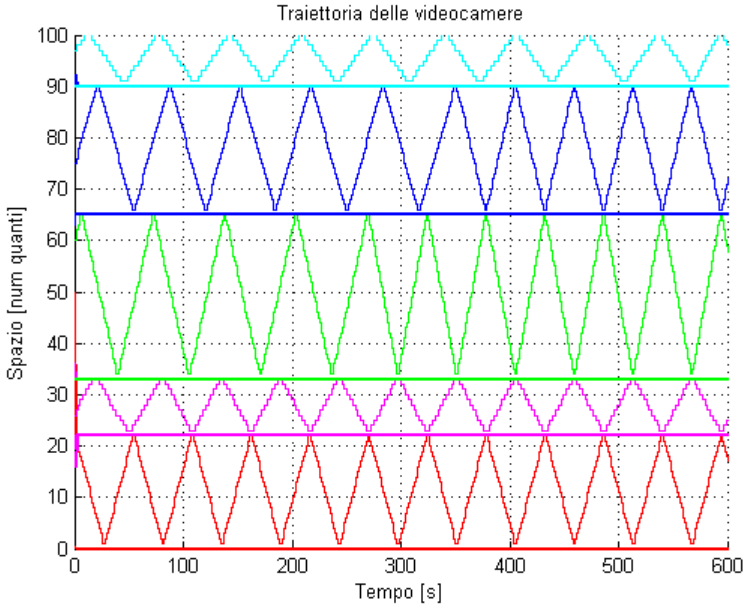


Figura 5.7: Dinamica delle videocamere in presenza di vincoli fisici e velocità diverse

Si nota che nel primo caso, dove c'è solo la presenza di videocamere con velocità diversa, pari a $v = [2, 1, 4, 1, 2]$ si ha che le videocamere più veloci si prendono una porzione di spazio maggiore in modo da minimizzare il periodo di patrolling.

Nell'altro caso invece sono stati introdotti vincoli fisici tra la seconda e terza telecamera e tra la terza e la quarta. Da come si può osservare immediatamente, in questo caso il consenso sulla partizione si ottiene in maniera molto più veloce perchè è come se dovessimo ottenere il consenso con 3 sistemi di telecamere separati: uno formato dalla 1 e 2, uno dalla 3 e infine uno da 3 e 4. Invece per quanto riguarda la messa in fase delle telecamere, che essendo un task secondario rispetto al patrolling e al tracking, ci mette di più in quanto, in tal caso il sistema è quello formato da tutte e cinque le telecamere.

5.4 Tracking di un evento

In questa sezione vogliamo mostrare come si comporta il sistema di telecamere in presenza di un evento da seguire; inizialmente avevamo pensato di far comunicare le telecamere in maniera randomizzarta cioè ogni telecamera parla in media ogni μ secondi. Questo tipo di comunicazione consente di ottenere lo stesso buone prestazioni del tracking da come si può vedere dalla figura seguente:

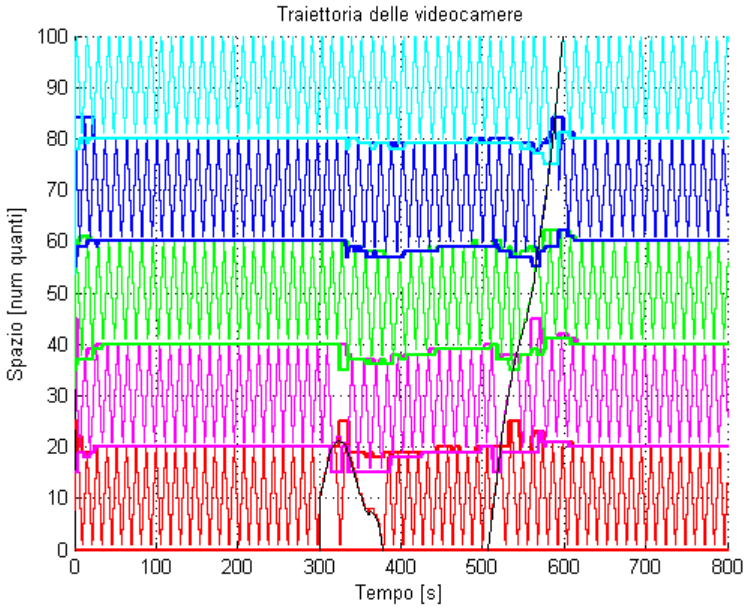


Figura 5.8: Dinamica delle videocamere in presenza di un evento, le telecamere comunicano in maniera randomizzata con $\mu = 5s$

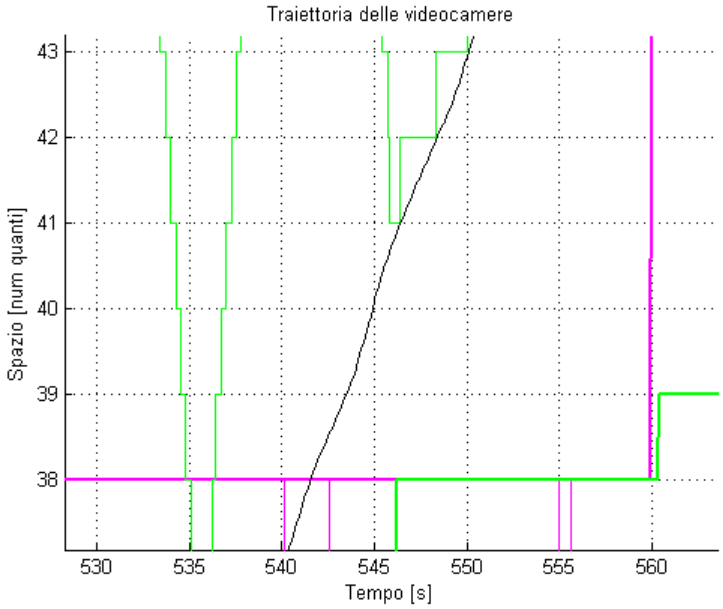


Figura 5.9: Particolare della realizzazione precedente

Infatti da come si può vedere, le prestazioni sono ancora molto buone: se si osserva la Figura 5.8 non si riesce quasi neanche a notare che l'evento ad un certo punto viene perso durante lo "scambio" da una telecamera a quella successiva a meno che non si osservi in dettaglio l'evoluzione come riportato in Figura 5.10 (si osservi che comunque l'evento viene perso per un brevissimo periodo di tempo pari a 5s circa).

Per ovviare a questo problema abbiamo reso "un po più deterministico" il modo in cui le telecamere comunicano, cioè abbiamo fatto in modo che la telecamera in tracking parli ad intervalli regolari, non aleatori e in maniera più frequente; questo è giustificato dal fatto che la fase di tracking è quella che ha velocità priorità massima rispetto a qualunque altro task. Infatti, da come si evince dalla figura sottostante ora l'evento(che è lo stesso non viene mai perso.

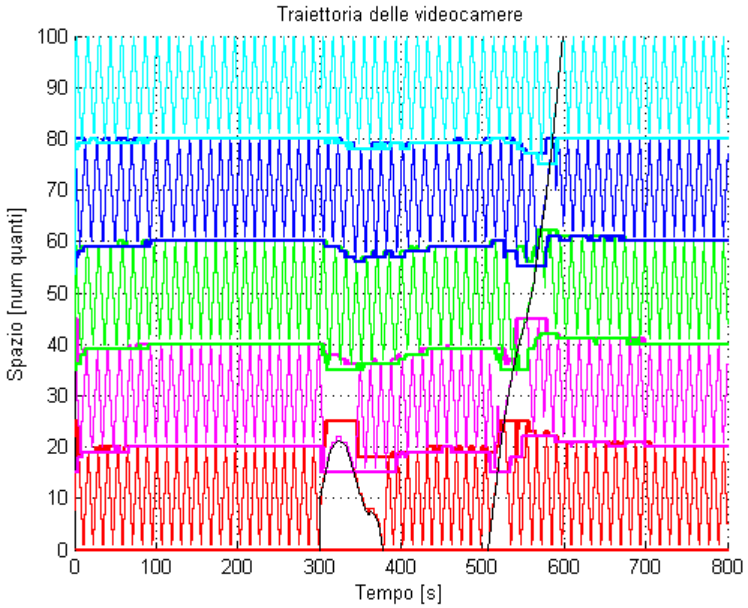


Figura 5.10: Dinamica delle videocamere in presenza di un evento, la telecamera in tracking non comunica in maniera randomizzata

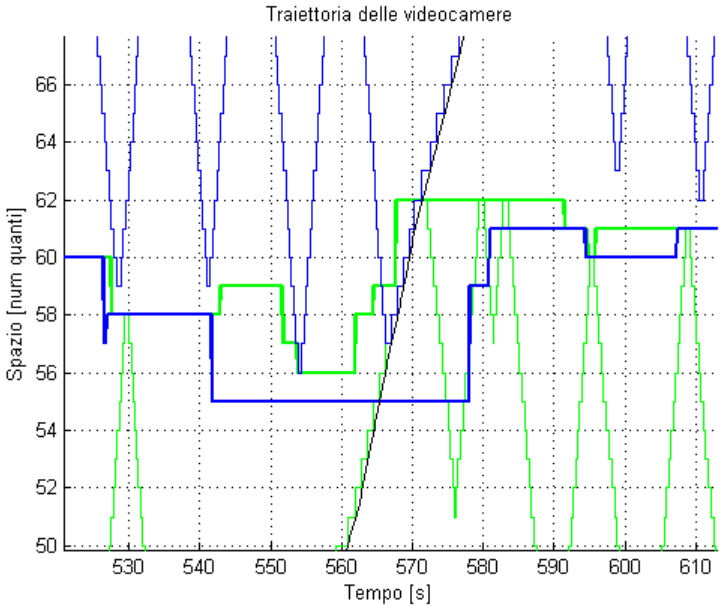


Figura 5.11: Particolare della realizzazione precedente

5.5 Guasto di una videocamera

In questa simulazione mostriamo il comportamento del sistema a seguito di un guasto ad una telecamera. Il guasto viene rilevato quando le telecamere adiacenti non ricevono più comunicazioni da quella che sta in mezzo per un determinato periodo di tempo. Quando si verifica questa situazione, le telecamere adiacenti a quella che avuto il guasto fanno coincidere i loro nuovi limiti con i vincoli fisici; la figura sottostante mostra quello che succede in seguito ad un guasto di una telecamera:

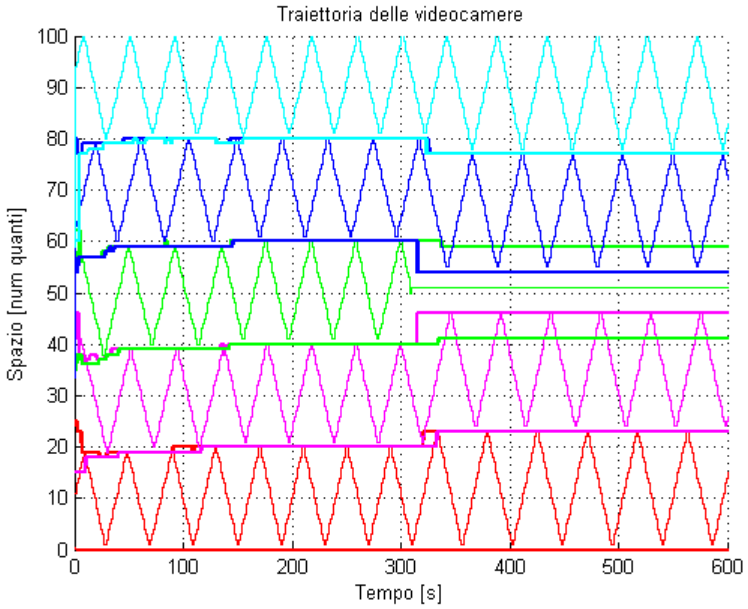


Figura 5.12: Guasto di una telecamera

Si evince che il sistema, dopo un gusto, ora è come fosse composto da due sistemi indipendenti, ciò si può notare dal grafico che mostra come una volta che la videocamera 3 si ferma, le altre due cercano di coprire più spazio possibile della partizione 3 e poi si risistemano i loro limiti in modo da avere ancora un partizionamento ottimo di quel che resta del sistema e cercano di rimettersi in fase.

Capitolo 6

Conclusioni

Il nostro progetto è il lavoro conclusivo di un percorso iniziato qualche anno fa. Come detto nell'introduzione la grande differenza rispetto ai lavori precedenti è stata la discretizzazione del perimetro per cercare di giungere alla dimostrazione matematica della convergenza all'ottimo.

Giunti alla conclusione del nostro percorso e valutando i dati ottenuti possiamo fare le seguenti riflessioni:

- la discretizzazzione del perimetro ha portato a complicare leggermente l'algoritmo di partizionamento del perimetro; in particolare si è reso necessario introdurre un'aleatorietà per l'assegnazione delle celle. Il beneficio è stato che effettivamente, la discretizzazione, come inizialmente ipotizzato, ha portato dei vantaggi per giungere alla dimostrazione matematica della convergenza all'ottimo anche se in maniera probabilistica.

- la discretizzazione del perimetro non ha compromesso irrimediabilmente il problema. Questo come fatto notare in precedenza succede se la misura del quanto è sufficientemente piccola e se si sceglie un numero di quanti in modo da non avere delle "oscillazioni" del tempo ottimo a regime. In particolare nel caso di videocamere con le medesime caratteristiche e quindi anche con la stessa velocità massima questo vincolo si traduce nel scegliere un numero di quanti che sia multiplo del numero delle videocamere.

- le simulazioni mostrano che l'algoritmo è robusto sia alla perdita di pacchetti durante la comunicazione sia alla perdita di una videocamera dovuta a possibili guasti.

- il protocollo di comunicazione utilizzato è il più generico e semplice possibile. Questo fatto secondo noi è molto importante perchè non pone vincoli in una realizzazione pratica del sistema e certamente la facilita. Un fattore molto importante è il rate di comunicazione. Come si è visto nelle simulazioni, soprattutto in caso di allarme, ovvero nel caso di presenza di un intruso, è bene dare una priorità maggiore alle videocamere che sono

in tracking. Questo si ottiene o aumentando il rate di comunicazione oppure facendo comunicare la videocamera in maniera determistica e con un intervallo di comunicazione adeguato.

Per quanto riguarda i lavori futuri possiamo dire che come detto in precedenza questo campo è già stato esplorato in amniera piuttosto approfondita. Come già accennato in alcuni lavori precedenti si potrebbero considerare delle aree 2D oppure strutture perimetriche più complesse.

Bibliografia

[1] Mauro Baseggio, Angelo Cenedese, Pierangelo Merlo, Mauro Pozzi, Luca Schenato *"Distributed perimeter patrolling and tracking for camera networks"* 2010

[2] Ruggero Carli Angelo Cenedese Luca Schenato *"Distributed Partitioning Strategies for Perimeter patrolling"* 2011

[3] Riccardo Alberton Ruggero Carli Angelo Cenedese Luca Schenato ,*"Multi-agent perimeter patrolling subject to mobility constraints"* 2012

[4] Baseggio Mauro, Merlo Pierangelo, Pozzi Mauro *"Coordinazione distribuita di telecamere per patrolling e tracking perimetrale"* 2010

[5] M. Valera, S.A. Velestian *"Intelligent distributed surveillance system: a review"* 2005

Elenco delle figure