



# UNIVERSITÀ DEGLI STUDI DI PADOVA

---

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE  
Corso di Laurea in Ingegneria dell'Automazione

## PATROLLING 2D

Progetto svolto nell'ambito del corso di "Progettazione di Sistemi di Controllo"

Progetto a cura di:  
GIORGIO PATTARELLO Matr. 626357  
DARIO VOLPATO Matr. 620359

Anno accademico 2010/2011

Febbraio 2011



# Indice

<b>Elenco delle figure</b>	<b>ii</b>
<b>Introduzione</b>	<b>v</b>
<b>1 Formulazione del problema</b>	<b>1</b>
1.1 Partizione di Voronoi . . . . .	2
1.1.1 Partizionamento di Voronoi . . . . .	3
1.1.2 Partizionamento di Voronoi centroidale . . . . .	3
1.1.3 Partizionamento discretizzato . . . . .	6
1.2 Soluzione distribuita al problema del partizionamento ottimo	8
<b>2 Soluzione proposta per il partizionamento</b>	<b>13</b>
2.1 Algoritmo di partizionamento ottimo sincrono (APSinc) . . .	14
2.2 Algoritmo di partizionamento ottimo con comunicazione sim- metrica di tipo gossip (APSimm) . . . . .	16
2.3 Algoritmo di partizionamento ottimo con comunicazione asim- metrica di tipo gossip (APASimm) . . . . .	17
<b>3 Soluzione proposta per il patrolling</b>	<b>19</b>
<b>4 Simulazioni</b>	<b>25</b>
4.1 Simulazione APSinc . . . . .	27
4.2 Simulazione APSimm . . . . .	30
4.3 Simulazione APASimm . . . . .	32
4.4 Simulazione con telecamera impegnata in un altro task (caso APASimm) . . . . .	35
4.5 Patrolling . . . . .	37
<b>5 Tipologie di partizionamento alternative</b>	<b>41</b>
5.1 Equitable partition . . . . .	41
<b>6 Conclusioni e possibili sviluppi futuri</b>	<b>43</b>
<b>Riferimenti bibliografici</b>	<b>45</b>



# Elenco delle figure

1.1	Esempio di partizione di Voronoi . . . . .	4
1.2	Due partizioni di Voronoi centroidali di un quadrato . . . . .	5
1.3	Sulla sinistra è rappresentato un partizionamento di Voronoi corrispondente a 10 punti disposti randomicamente su un quadrato; i puntini corrispondono ai punti selezionati mentre i cerchietti ai relativi centroidi. Sulla destra invece è rappresentata una partizione di Voronoi centroidale . . . . .	5
1.4	Due partizioni di Voronoi centroidali che derivano dalla convergenza dell'indice $H$ ad un minimo locale (a) o globale (b) a seconda delle condizioni iniziali . . . . .	6
1.5	Discretizzazione di una superficie . . . . .	6
1.6	Esempio di un grafo relativo ad una regione discretizzata . . . . .	7
1.7	Esempio: a) grafo non orientato e connesso b) grafo non orientato e non connesso . . . . .	10
3.1	Descrizione di una regione discretizzata . . . . .	20
3.2	(a) Regione; (b) Griglia e discretizzazione della regione (c) Matrice delle posizioni . . . . .	21
3.3	Possibile ciclo non ottimale in presenza di righe con almeno un valore 1 dispari nella matrice delle posizioni . . . . .	21
3.4	Illustrazione della strategia di patrolling utilizzata mediante la creazione di $M$ , matrice delle posizioni, e del vettore $v$ di attraversamento . . . . .	22
4.1	Regione discretizzata 20*20 . . . . .	25
4.2	Inizializzazione del sistema . . . . .	26
4.3	Grafo di comunicazione . . . . .	27
4.4	Risultati della simulazione dell'algorithmo APSinc . . . . .	28
4.5	Risultati della simulazione dell'algorithmo APSinc considerando le osservazioni 1 e 2 . . . . .	29
4.6	Andamento indice $H$ . . . . .	30
4.7	Risultati della simulazione dell'algorithmo APSimm . . . . .	31
4.8	Andamento indice $H$ . . . . .	32

4.9	Risultati della simulazione dell'algoritmo APASimm . . . . .	33
4.10	Andamento indice H . . . . .	34
4.11	Telecamera regione rosso ad un certo istante smette di perlustrare la sua regione . . . . .	35
4.12	Simulazione redistribuzione dell'area con porzione di area non controllata da nessuna telecamera . . . . .	36
4.13	Patrolling (prima parte) . . . . .	37
4.14	Patrolling (seconda parte) . . . . .	38
4.15	Costo del patrolling . . . . .	39

# Introduzione

Per monitorare scenari a larga scala attualmente si fa ricorso - e ancor più lo si farà in futuro - a sistemi di videosorveglianza in cui decine o centinaia di telecamere intelligenti Pan-Tilt-Zoom (PTZ) si coordinano tra loro per sorvegliare un'area, cooperando così a rilevare ed inseguire oggetti o persone, prendendo decisioni attraverso l'uso di algoritmi che analizzano dati video. A tale scopo, le telecamere sono utilizzate come sensori installati in un ambiente e connessi tra loro attraverso una rete di comunicazione.

Un tipico compito di tali sistemi è quello del patrolling, cioè perlustrare un'area per proteggerla e sorvegliarla, trovando un compromesso tra la complessità di installazione e le prestazioni di copertura. Una buona strategia di patrolling è quella di minimizzare il tempo che intercorre tra due visite dello stesso punto, assicurandosi che tutta l'area sia sorvegliata.

La maggior parte dei sistemi di sorveglianza al giorno d'oggi è centralizzata e human-controlled, tuttavia c'è la necessità di avere sistemi scalabili utilizzati per patrolling automatizzati e tracking (inseguimenti) senza intervento umano. Questo compito non è così semplice avendo come scopo ultimo quello di garantire un buon comportamento globale del sistema utilizzando solo informazioni locali e coordinando diverse telecamere con differenti risorse e vincoli (o *constraints*).

Quello che viene proposto in questo lavoro è una soluzione al problema del patrolling su una regione da parte di una rete di telecamere che comunicano tra loro in modo distribuito. Deve inoltre essere garantita la stabilità del sistema di sorveglianza in caso di malfunzionamenti di una o più telecamere (o assegnazione di altri task con priorità maggiore).

La maggior parte della letteratura a riguardo studia soprattutto il problema di sistemi costituiti da robot intelligenti e non da telecamere. Comunque molti dei concetti fondanti possono essere presi in considerazione anche nel problema in esame.

Per quanto riguarda la sorveglianza di perimetri, o di regioni a cui si può associare una struttura unidimensionale (come ad esempio corridoi), la soluzione ottima è già nota in letteratura [1]. Se l'ambiente che si va a considerare è invece di tipo bidimensionale (ad esempio un aeroporto) questo problema non è ancora stato risolto appieno.

In letteratura esistono varie soluzioni quando la comunicazione tra le teleca-

mere avviene tramite comunicazioni di tipo sincrono o asincrono simmetrico. In questo progetto ci si propone di risolvere il problema quando la comunicazione avviene in modo asincrono asimmetrico, dando quindi una valida ed efficace soluzione in sistemi distribuiti.

**Stato dell'arte** Come già anticipato, esiste una vasta letteratura che tratta le differenti tematiche che si devono affrontare per risolvere il problema che ci si pone in questo lavoro nella sua interezza. Prima di tutto si deve affrontare il problema del partizionamento della regione. Problemi di partizionamento sono affrontati nel lavoro classico di Lloyd [2] che utilizza le *partizioni di Voronoi*. Un altro tipo di partizionamento che potrebbe essere utilizzato è quello delle *equitable partition* trattato in [3]. Per sviluppare i concetti di controllo di partizionamento e copertura sono utilizzabili degli *gradient descent algorithm* come in [4] ed in [5]. In [6] si descrivono algoritmi che risolvono i problemi di partizionamento in caso di comunicazioni sincrone e regioni discretizzate. Quando invece si parla di comunicazioni asincrone di tipo gossip, sempre a regioni discretizzate, si fa soprattutto riferimento a [7]. Per quanto riguarda la strategia di patrolling ci si rifà a delle strategie utilizzate ad esempio nell'agricoltura o nella tessitura. Esse possono essere definite a *solco di bue* o ad andamento *bustrofedico* - termine che si prende da un tipo di antica scrittura in cui la sequenza dei segni procedeva, riga dopo riga, alternativamente da sinistra a destra e da destra a sinistra, seguendo un percorso simile a quello tracciato sul campo dall'aratro; andamento inoltre che ricorda quello della spola e della mano che la guida.



# Capitolo 1

## Formulazione del problema

Il progetto prevede la perlustrazione di una data area mediante telecamere. I dati iniziali di cui si dispone sono:

- regione  $R$  chiusa e connessa, sulla quale si eseguirà il patrolling
- $N$  telecamere dotate di capacità computazionali e di comunicazione di dati
- insieme  $S$  di strategie di patrolling ammissibili
- constraint fisici  $C_i$ ,  $i = 1, \dots, N$ , che individuano le massime aree visibili da ciascuna telecamera

Il primo obiettivo di questo progetto è quello di suddividere la regione  $R$  in  $N$  sottoregioni  $R_1, \dots, R_N$ , in modo tale che

$$\bigcup_{i=1}^N R_i = R$$

Ogni telecamera sarà responsabile del patrolling su una sottoregione  $R_i$  assegnata.

Si dovrà scegliere quindi una strategia di patrolling ed una suddivisione in regioni efficienti in maniera tale da minimizzare un funzionale costo desiderato.

Si procede ora nella trattazione con un approccio più matematico. Si definisce inizialmente il funzionale costo che si è preso in considerazione per giungere alla soluzione del problema.

Sia  $\Omega$  una regione,  $q$  un punto interno ad essa, ed  $s$  una delle possibili strategie di patrolling appartenenti all'insieme  $S$ .

Si definisce il funzionale costo  $Ind_{\Omega}(q; s)$  come *il tempo massimo sui punti di  $\Omega$  che intercorre tra due visite successive del punto  $q$  utilizzando la strategia  $s$ .*

Si considera ora il funzionale costo  $Ind(\Omega; s)$  definito come *il tempo massimo*

che intercorre tra due visite successive di un punto utilizzando la strategia  $s$ . In termini più matematici

$$Ind(\Omega; s) = \max_{q \in \Omega} Ind_{\Omega}(q; s)$$

Si introduce inoltre la nozione di ricoprimento N-dimensionale di una regione  $R$  come l'insieme

$$\mathbf{R}_c(R) = \left\{ \{R_i\}_{i=1}^N : \bigcup R_i = R \quad e \quad R_i \subseteq C_i \right\}$$

L'obiettivo è quello di trovare un ricoprimento N-dimensionale e la migliore strategia tale da minimizzare il tempo tra due visite successive di uno stesso punto, cioè

$$\min_{\{R_i\}_{i=1}^N \in \mathbf{R}_c} \min_{\{s_i\}_{i=1}^N \in S^N} \max_{1 \leq i \leq N} Ind(R_i; s_i) \quad (1.1)$$

Risolvere il problema formulato in (1.1) risulta molto difficile in quanto occorrerebbe minimizzare contemporaneamente sia su  $S$  che su  $\mathbf{R}_c$ .

Per rendere il problema più facilmente gestibile ci si concentra su una soluzione subottima: si assume che tutte le telecamere adottino la stessa strategia  $s$  di patrolling, ritenuta quella ragionevolmente più efficiente tra tutte le  $s_i \in S$ . Dunque, ci si concentra maggiormente sul problema del partizionamento ottimo.

Il problema semplificato diventa

$$\min_{\{R_i\}_{i=1}^N \in \mathbf{R}_p} \max_{1 \leq i \leq N} Ind(R_i; s) \quad (1.2)$$

dove  $s$  è assegnata a priori.

**Osservazione 1** Nella suddivisione in sottoregioni è intuitivo che la soluzione ottima si ottenga con un partizionamento, ovvero un particolare ricoprimento. Un il partizionamento è definito come l'insieme

$$\mathbf{R}_p = \left\{ \{R_i\}_{i=1}^N : \bigcup_{i=1}^N R_i = R \quad , \quad R_i \neq \emptyset \quad , \quad R_i \cap R_j = \emptyset \quad , \quad R_i \subseteq C_i \right\}$$

L'utilizzo di un partizionamento permetterebbe di ottenere delle sottoregioni senza overlap, cioè senza sovrapposizioni tra le sottoregioni.

## 1.1 Partizione di Voronoi

Per eseguire un partizionamento ottimo ci si rifà a quanto è considerato buono in letteratura, nello specifico si considera il partizionamento di Voronoi.

### 1.1.1 Partizionamento di Voronoi

La trattazione qui esposta viene svolta ad un livello generale d-dimensionale.

Nel nostro caso specifico sarà ricondotta al caso bidimensionale.

Sia  $A \subseteq \mathbb{R}^d$  regione che deve essere partizionata secondo Voronoi.

Sia  $G$  un insieme di punti distinti appartenete ad  $A$

$$G = \{(g_1, g_2, \dots, g_N) : g_i \in A \quad \forall i\}$$

Si definisce per  $A$  la partizione di Voronoi  $V(G) = (V_1(G), \dots, V_N(G))$  generata dai punti  $(g_1, g_2, \dots, g_N)$  come

$$V_i(G) = \{x \in A : \|x - g_i\| \leq \|x - g_j\| \quad \forall g_j \neq g_i\} \quad (1.3)$$

Ci si riferisce a  $G$  come l'insieme dei *generatori* di  $V(G)$  e a  $V_i(G)$  come le regione di Voronoi relativa all' $i$ -esimo generatore  $g_i$ .

Si definisce il *bisector* tra  $g_i$  e  $g_j$ , con  $g_i, g_j \in G$  e  $i \neq j$ , come

$$b(g_i, g_j) = \{x \in A : \|x - g_i\| = \|x - g_j\|\}$$

Il piano  $b(g_i, g_j)$  è ortogonale al segmento che congiunge  $g_i$  con  $g_j$  e lo divide in due parti. Il *bisector* divide quindi  $A$  in due sottoinsiemi portando alla definizione dell'insieme

$$D(g_i, g_j) = \{\|x - g_i\| \leq \|x - g_j\|\}$$

regione di dominio di  $g_i$  rispetto a  $g_j$ .

Infine la partizione di Voronoi  $V(G)$  può essere equivalentemente definita come

$$V_i(G) = \bigcap_{j \in I_m - \{i\}} D(g_i, g_j) \quad I_m = 1, 2, \dots, N$$

cioè l'intersezione di tutte le regioni di dominio di  $g_i$  rispetto ai  $g_j$  con  $g_i \neq g_j$ .

Questa seconda definizione mostra chiaramente che, se  $A$  è convessa, ciascuna regione di Voronoi relativa all' $i$ -esimo generatore  $g_i$  è un insieme convesso. Si può definire la partizione di Voronoi così ottenuta come una partizione convessa (vedi Figura 1.1).

### 1.1.2 Partizionamento di Voronoi centroidale

Il partizionamento di Voronoi centroidale è un tipo speciale di partizionamento di Voronoi. Un partizionamento di Voronoi è definito centroidale quando i punti generatori di ciascuna regione corrispondono al loro centro

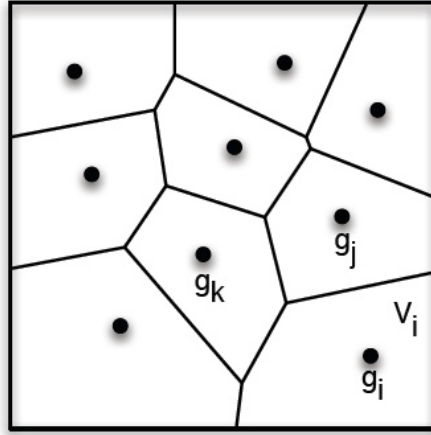


Figura 1.1: Esempio di partizione di Voronoi

di massa.

Data una regione  $V \subseteq \mathbb{R}^d$  e una funzione di densità  $\rho$ , definita in  $V$ , il centro di massa, o centroide,  $Cd(V)$  di  $V$  è definito come:

$$Cd(V) = \frac{\int_V y \rho(y) dy}{\int_V \rho(y) dy} \quad (1.4)$$

Per semplicità si considerano le funzioni di densità come costanti unitarie e quindi la (1.4) si semplifica in

$$Cd(V) = \frac{\int_V y dy}{|V|} \quad (1.5)$$

dove  $|V|$  è la misura di Lebesgue dell'insieme  $V$ . Quest'ultima risulta essere l'area della regione  $V$ .

Dati  $k$  punti  $Cd_i(V)$ ,  $i = 1, \dots, k$ , si possono definire le regioni di Voronoi  $\hat{V}_i$ ,  $i = 1, \dots, k$ ; d'altra parte date le regioni  $\hat{V}_i$ ,  $i = 1, \dots, k$  si possono definire i loro centroidi  $Cd_i(V)^*$ ,  $i = 1, \dots, k$ .

Qui si è interessati al caso in cui

$$Cd_i(V) = Cd_i(V)^* \quad i = 1, \dots, k$$

ossia una situazione nella quale i punti che servono come generatori per il partizionamento di Voronoi sono essi stessi i centroidi di queste regioni.

La soluzione di questo problema non è unica: per esempio, come mostrato in Figura 1.2, considerando il caso in cui  $d = 2$ ,  $\Omega \subset \mathbb{R}^2$  è un quadrato e  $\rho$  uniforme, è possibile ottenere più di un partizionamento di Voronoi centroidale.

La possibilità di ottenere regioni di Voronoi centroidali è un caso abbastanza

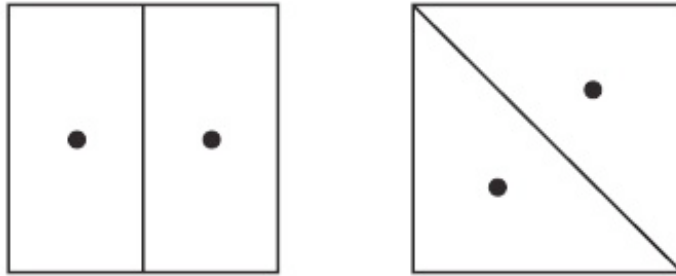


Figura 1.2: Due partizioni di Voronoi centroidali di un quadrato

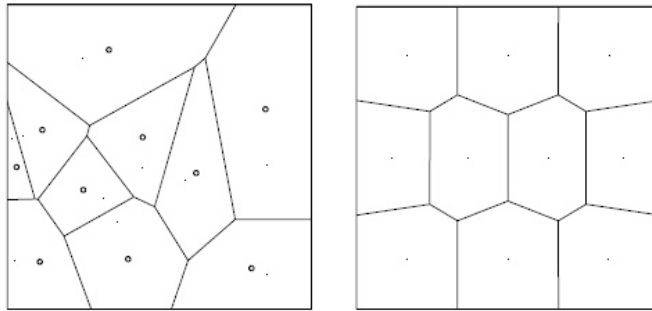


Figura 1.3: Sulla sinistra è rappresentato un partizionamento di Voronoi corrispondente a 10 punti disposti randomicamente su un quadrato; i puntini corrispondono ai punti selezionati mentre i cerchietti ai relativi centroidi. Sulla destra invece è rappresentata una partizione di Voronoi centroidale

speciale. In generale infatti scelti i punti in  $\mathbb{R}^d$  questi non risultano essere i centroidi relativi alle regioni di Voronoi (Figura 1.3).

Le partizioni di Voronoi centroidali coincidono con i punti critici di

$$H(\{V_i\}_{i=1}^N, \{C_d(V_i)\}_{i=1}^N) = \sum_{i=1}^N \int_{x \in V_i} \|x - c_i\|^2 dx \quad (1.6)$$

In letteratura esistono molti algoritmi iterativi che risolvono il problema di minimizzazione dell'indice  $H$ . Questi algoritmi sono basati su un *gradient descent* che converge ad un minimo locale di  $H$ . Purtroppo non sempre converge al minimo globale in quanto questo dipende dalle condizioni iniziali (vedi Figura 1.4).

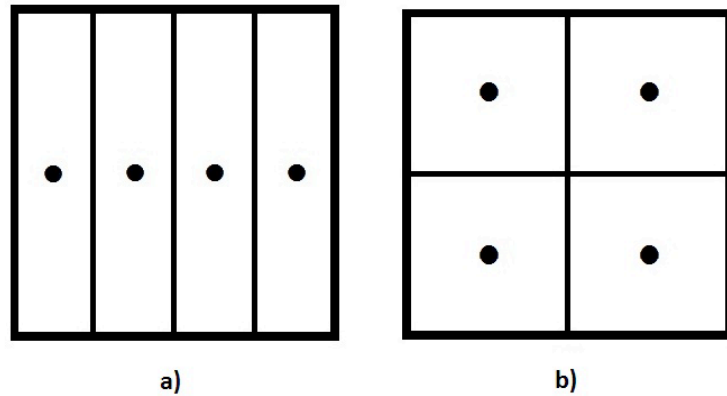


Figura 1.4: Due partizioni di Voronoi centroidali che derivano dalla convergenza dell'indice  $H$  ad un minimo locale (a) o globale (b) a seconda delle condizioni iniziali

### 1.1.3 Partizionamento discretizzato

L'implementazione del partizionamento di Voronoi in ambiente continuo ha una difficoltà computazionale molto elevata e si potrebbero arrivare a dover gestire situazioni nelle quali non esistono euristiche soddisfacenti nel caso asincrono (che definiremo più avanti).

Per questo si è vista la necessità di ripensare al partizionamento di Voronoi attuandolo in un ambiente discretizzato. La regione  $R$  viene mappata con una griglia. A ciascuna cella della griglia è associato un punto (Figura 1.5). Così facendo le regioni sono caratterizzate da un insieme finito di punti.

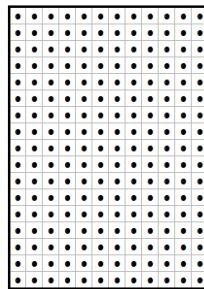


Figura 1.5: Discretizzazione di una superficie

La definizione di partizione di Voronoi rimane invariata. Considerata una regione discretizzata  $A$  (insieme finito e numerabile)

$$A = \{(x_1, x_2, \dots, x_n) : (x_1, x_2, \dots, x_n) \in \mathbb{R}^N\}$$

e  $G$  un insieme di punti appartenente ad  $A$

$$G = \{(g_1, g_2, \dots, g_N) : g_i \in A \quad \forall i\}$$

si definisce per  $A$  la partizione di Voronoi  $V(G) = (V_1(G), \dots, V_N(G))$  generata dai punti  $(g_1, g_2, \dots, g_N)$  come

$$V_i(G) = \{x \in A : d(x, g_i) \leq d(x, g_j) \quad \forall g_j \neq g_i\}$$

dove  $d(x, g_i)$  è la distanza dal punto  $x$  al punto generatore  $g_i$ .

Il concetto di distanza in questo approccio è generico e deve essere chiarito: questa potrebbe essere sia la distanza euclidea che la distanza sul grafo. Il fatto di aver discretizzato la regione non permette di usare liberamente la definizione di distanza euclidea in presenza di regioni non convesse o regioni con buchi o ostacoli. Sarebbe più opportuno associare alla discretizzazione un grafo  $G(V, E)$  dove i vertici sono i punti appartenenti ad  $A$ , quindi  $V \equiv A$ , mentre l'insieme  $E$  è formato da tutti gli archi  $(x, y)$  tale che i punti  $x, y \in V \equiv A$  e le relative celle, associate ad  $x$  e  $y$ , hanno un lato coincidente.

Come distanza si potrebbe utilizzare il concetto di cammino minimo del grafo tra due vertici, calcolabile attraverso l'algoritmo di Dijkstra. Utilizzando questo approccio si possono arrivare a considerare anche regioni convesse o con buchi.

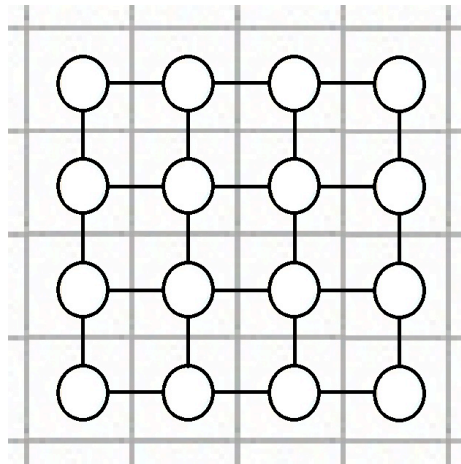


Figura 1.6: Esempio di un grafo relativo ad una regione discretizzata

Per semplificare l'implementazione di questo progetto si considerano regioni convesse e senza buchi. Di conseguenza è accettabile utilizzare come distanza la distanza euclidea, vista anche la minore onerosità computazionale che richiede. Perciò la definizione di partizione di Voronoi resta quindi invariata rispetto alla (1.3).

Dopo la discretizzazione, per utilizzare la partizione di Voronoi centroidale,

occorre dare una definizione opportuna di centroide. Non è detto infatti che, usando la definizione naturale, il centroide  $z^*$  appartenga alla regione considerata, in quanto ora  $A$  è definito un insieme finito di punti.

Si definisce quindi il centroide di una regione discreta  $\Omega \subset \mathbb{R}^N$  come

$$x^* \in \arg \min_{x \in \Omega} \left\{ \sum_{y \in \Omega} d(x, y) \right\} \quad (1.7)$$

Con questa metodologia potrebbe essere trovato più di un centroide. Occorre quindi fare una decisione. Per scelta progettuale sarà indifferente quale centroide scegliere tra i papabili. Si suppone che si definisca in  $R$  una relazione di ordinamento totale ' $<$ '. Se si considerasse la lista ordinata dei punti che contraddistinguono una determinata regione  $R$ , potremmo scrivere  $R = \{1, 2, \dots, |R|\}$ .

Con questa assunzione, un criterio accettabile è quello di scegliere il minore, ovvero d'ora in poi assumeremo per una certa regione  $\Omega$ :

$$Cd(\Omega) = \min \left\{ \arg \min_{x \in \Omega} \left\{ \sum_{y \in \Omega} d(x, y) \right\} \right\}$$

La definizione appena presentata non calcola il vero centroide della regione  $\Omega$ , ma ne trova un'approssimazione considerata accettabile.

Per chiarezza si termina la sezione focalizzando l'attenzione su quale sia l'obiettivo definitivo: trovare una partizione  $\{V_i\}_{i=1}^N$  tale da minimizzare il funzionale costo

$$H \left( \{V_i\}_{i=1}^N, \{C_d(V_i)\}_{i=1}^N \right) = \sum_{i=1}^N \sum_{q \in V_i} d(q, C_d(V_i)) \quad (1.8)$$

## 1.2 Soluzione distribuita al problema del partizionamento ottimo

Si procede ad una partizione considerando uno scenario di tipo distribuito. Un sistema distribuito è caratterizzato da tanti processi autonomi che interagiscono tra loro comunicando attraverso una rete, al fine di raggiungere un obiettivo comune.

Un sistema distribuito è caratterizzato da:

- una condizione iniziale
- aggiornamento dello stato del processo
- grafo relativo alla rete di comunicazione



## 1.2 Soluzione distribuita al problema del partizionamento ottimo

La condizione iniziale prevede che ad ogni telecamera sia associata una certa regione iniziale  $R_i(0)$ . L'inizializzazione del sistema deve soddisfare determinate condizioni:

- $R_i(0) \neq \emptyset \forall i$
- $\bigcup_i R_i = R$  dove  $R$  è l'area totale da perlustrare
- $R_i(0) \subseteq C_i$

Per praticità si considera  $R_i(0) = C_i$ . Questa inizializzazione rispetta le proprietà precedenti.

L'attività della rete è suddivisa temporalmente in periodi. In ognuno di questi periodi una o più telecamere invia nella rete delle informazioni relative al proprio stato, cioè relative alla propria regione su cui viene svolto il patrolling. Queste informazioni vengono ricevute da una o più telecamere che le utilizzano per aggiornare la propria regione, in modo tale che il sistema raggiunga, in un tempo finito, un partizionamento ottimo (come descritto in (1.2)).

Il flusso di informazioni è regolato da un grafo non orientato, connesso e tempo invariante  $G(V, E)$ , dove  $V$  è l'insieme dei vertici del grafo

$$V = \{1, 2, \dots, N\}$$

mentre  $E$  rappresenta l'insieme degli archi che connettono i vari vertici (le connessioni tra le varie telecamere in grado di comunicare tra loro)

$$E = \{(i, j) : i, j \in V\}$$

In un grafo non orientato se l'arco  $(i, j) \in E$  anche  $(j, i) \in E$ . Esiste dunque una bidirezionalità della comunicazione, e questo viene rappresentato graficamente con una linea che collega il vertice  $i$  con il vertice  $j$ .

Se  $(i, j) \in E$ , si dice che il vertice  $i$  è adiacente a  $j$  o che l'arco  $(i, j)$  connette i vertici  $i$  e  $j$  oppure che le telecamere  $i$  e  $j$  sono vicine e che sono quindi in grado di comunicare.

Si definisce allora  $\forall i \in V$  l'insieme

$$N_i = \{j \in V : (i, j) \in E\}$$

come l'insieme dei vicini di  $i$ .

Si può parlare di tempo invarianza quando l'insieme dei vertici e degli archi caratterizzanti il grafo non viene modificato al variare del tempo. Inoltre si parla di grafo connesso quando per ogni coppia di vertici esiste un cammino (una sequenza di archi consecutivi) che li collega; matematicamente

$$\forall i, j \in V \quad \exists e_1, e_2, \dots, e_k \in E$$

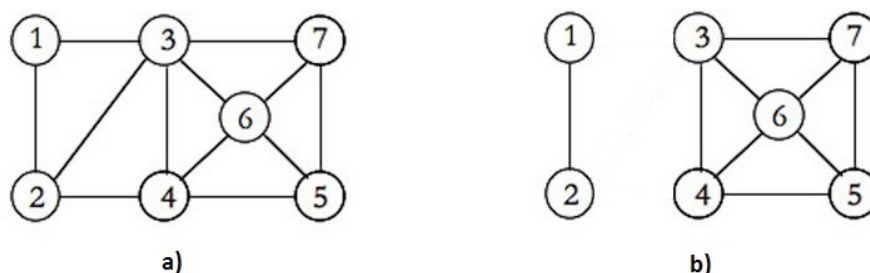


Figura 1.7: Esempio: a) grafo non orientato e connesso b) grafo non orientato e non connesso

tali che

$$e_1 = (i, n), e_2 = (n, m), \dots, e_{k-1} = (o, p), e_k = (p, j)$$

Nel caso in esame l'insieme dei vertici del grafo è rappresentato dall'insieme delle  $N$  telecamere:

$$V = \{1, 2, 3, \dots, N\}$$

L'insieme degli archi è formato da tutte le coppie di indici  $(i, j)$  che soddisfano la seguente proprietà:

$$(i, j) \in E \Leftrightarrow C_i \cap C_j \neq \emptyset$$

Costruito il grafo che regola il flusso di informazioni della rete di telecamere, si considereranno tre differenti scenari che dipendono dal protocollo di comunicazione adottato dall'insieme di telecamere, per scambiarsi le informazioni. I tipi di comunicazione sono:

- comunicazione sincrona
- comunicazione simmetrica di tipo gossip
- comunicazione asimmetrica di tipo gossip

Nel caso di una *comunicazione sincrona* le telecamere parlano tra di loro in maniera sincrona, cioè, a ciascun round di comunicazione ogni telecamera trasmette ai propri vicini le informazioni relative al proprio stato, (regione), al tempo corrente.

Invece in una comunicazione simmetrica di tipo gossip, ad ogni iterazione di trasmissione, solo una coppia di telecamere vicine comunica mentre le altre telecamere sono in attesa (né trasmettono, né ricevono alcuna informazione). Infine nel protocollo asimmetrico di tipo gossip, a differenza del gossip simmetrico non vi è bidirezionalità nella comunicazione: considerando due nodi

## **1.2 Soluzione distribuita al problema del partizionamento ottimo**

$i$  e  $j$  o  $i$  trasmette a  $j$ , e quindi solo  $j$  aggiorna la propria regione, o  $j$  trasmette a  $i$ .

Verranno in seguito presentati tre algoritmi differenti che realizzino il partizionamento ottimo (soddisfacente (1.2)) utilizzando queste tre differenti tipologie di protocolli di comunicazione.



## Capitolo 2

# Soluzione proposta per il partizionamento

Si presentano ora tre algoritmi che risolvono il problema del partizionamento di un'area attraverso l'uso di telecamere, che comunicano tra di loro scambiandosi informazioni, considerando i tre protocolli di comunicazione visti in precedenza: la comunicazione sincrona, la comunicazione simmetrica e asimmetrica di tipo gossip.

I tre criteri di comunicazione, a causa della loro diversità, non danno la possibilità di utilizzare un unico algoritmo, quindi si è vista la necessità di analizzarli separatamente, definendo quindi:

- l'algoritmo di partizionamento ottimo sincrono (APSinc)
- l'algoritmo di partizionamento ottimo simmetrico gossip (APSimm)
- l'algoritmo di partizionamento ottimo asimmetrico gossip (APASimm)

La situazione di partenza è quella di supporre noto il grafo  $G(V, E)$  relativo alla rete di comunicazione, ogni telecamera quindi conosce i vicini a cui inviare le proprie informazioni.

È nota inoltre la regione discretizzata  $R \subset \mathbb{R}^2$  che si vuole partizionare in  $N$  sotto-regioni  $R_1, R_2, \dots, R_N$  tali da soddisfare, ad ogni round di aggiornamento, le proprietà

- $R_i \neq \emptyset$
- $\bigcup_i R_i = R$

Dapprima si procederà alla descrizione degli algoritmi senza considerare i constraints fisici delle telecamere, cioè il caso in cui ciascuna telecamera sarebbe in grado di sorvegliare l'intera regione  $R$ .

Si osserverà successivamente come modificare gli algoritmi nel caso in cui si

debbano considerare i constraints, a causa della non totale visibilità dell'intera regione  $R$ , da parte di qualche telecamera, determinando così un limite massimo sulla area che tali telecamere possono perlustrare.

## 2.1 Algoritmo di partizionamento ottimo sincrono (APSinc)

In questo paragrafo, si introduce l'algoritmo di partizionamento ottimo sincrono, denotato come APSinc, questo é formalmente descritto nel modo seguente:

**Stati del processore:** Per ogni  $i \in \{1, \dots, N\}$  la telecamera  $i$  tiene in memoria la sua regione di appartenenza  $R_i$ , e il relativo centroide  $Cd(R_i)$ , visto in (1.7);

**Inizializzazione:** Per ogni  $i \in \{1, \dots, N\}$  alla telecamera  $i$  è associato un valore iniziale della regione da perlustrare,  $R_1(0), R_2(0), \dots, R_N(0)$  tali da verificare la definizione di partizionamento di  $R$  e in modo che i centroidi  $Cd(R_1(0)), Cd(R_2(0)), \dots, Cd(R_N(0))$  associati alle  $R_1(0), R_2(0), \dots, R_N(0)$  non siano coincidenti;

**Trasmissione informazioni:** A ciascun istante  $t \in \mathbb{N}$ , per ogni  $i \in \{1, \dots, N\}$  la telecamera  $i$  trasmette ai propri vicini la regione  $R_i(t)$  e il centroide relativo  $Cd(R_i(t))$ ;

**Update dello stato:** A ciascun istante  $t$ , per ogni  $i \in \{1, \dots, N\}$  la telecamera  $i$ , dopo aver ricevuto le  $R_j(t)$  e  $Cd(R_j(t))$  con  $j \in N_i$ , dove  $N_i = \{j \in V : (i, j) \in E\}$  è l'insieme dei vicini di  $i$ , compie le seguenti operazioni:

1. computa  $\forall j \in \{1, \dots, m\}$  gli insiemi:

$$W_{i \rightarrow j} := \{x \in R_i(t) : d(x, Cd(R_j(t))) < d(x, Cd(R_i(t)))\}$$

$$W_{j \rightarrow i} := \{x \in R_j(t) : d(x, Cd(R_i(t))) < d(x, Cd(R_j(t)))\}$$

$$W_{j \cong i} := \{x \in R_i(t) \cup R_j(t) : d(x, Cd(R_i(t))) = d(x, Cd(R_j(t)))\}$$

2. aggiorna la sua regione ed il relativo centroide:

$$R_i(t+1) := \left( \left( R_i(t) - \bigcup_j W_{i \rightarrow j} \right) \cup \left( \bigcup_j W_{j \rightarrow i} \right) \right) \cup \left( \bigcup_j W_{j \cong i} \right)$$

$$Cd(R_i(t+1)) = \min \{ \arg \min \{ \sum_{x \in R_i(t+1)} d(x, y) \} \}$$

**Osservazione 1** Come si può facilmente vedere dall'equazione di update dello stato, le regioni non rispettano la condizione di non overlap, in quanto l'insieme  $W_{j\cong i} \equiv W_{i\cong j}$  quindi i punti appartenenti a questi insiemi coincidono, stanno sia in  $R_i(t)$  che in  $R_j(t)$ .

Per evitare l'overlap, si può modificare l'equazione di aggiornamento nel modo seguente:

$$R_i(t+1) := \left( \left( R_i(t) - \bigcup_j W_{i \rightarrow j} \right) \cup \left( \bigcup_j W_{j \rightarrow i} \right) \right) \cup \left( \bigcup_{j>i} W_{j \cong i} \right) - \left( \bigcup_{i>j} W_{j \cong i} \cap R_j(t) \right)$$

cioè assegnando i punti che sono ad uguale distanza sia da  $Cd(R_i(t))$  che da  $Cd(R_j(t))$ , alla telecamera con indice minore.

**Osservazione 2** Nell'illustrare questo algoritmo si è considerato irrilevante la presenza dei constraints fisici, in quanto si è supposto che  $\forall i$  con  $i \in \{1, \dots, N\}$  sia soddisfatta la condizione  $C_i = R$ .

Nella realtà è difficile trovare una situazione dove tutte le telecamere siano in grado di osservare tutta l'area da pattugliare, quindi si deve rivedere l'algoritmo tenendo conto di questa limitazione.

Supponiamo quindi di essere nel caso in cui  $\forall i$  con  $i \in \{1, \dots, N\}$  vale  $C_i \subset R$ ; si devono portare allora le seguenti modifiche all'algoritmo:

- **Stati del processore:** Le telecamere  $i$  oltre che a tenersi in memoria  $R_i$  e  $Cd(R_i)$  deve conoscere i constraints fisici  $C_i$  delle telecamere dei propri vicini;
- **Nell'inizializzazione:** Le regioni iniziali,  $R_1(0), R_2(0), \dots, R_N(0)$  oltre a verificare la condizione di centroidi non coincidenti, devono  $\forall i$  con  $i \in \{1, \dots, N\}$  essere  $R_i(0) \subseteq C_i$ .
- **Nell'update dello stato:** i tre insiemi  $W_{i \rightarrow j}$ ,  $W_{j \rightarrow i}$  e  $W_{j \cong i}$  si modificano in questo modo:

$$W_{i \rightarrow j} := \{x \in R_i(t) : d(x, Cd(R_j(t))) < d(x, Cd(R_i(t))) \quad e \quad x \in C_j\}$$

$$W_{j \rightarrow i} := \{x \in R_j(t) : d(x, Cd(R_i(t))) < d(x, Cd(R_j(t))) \quad e \quad x \in C_i\}$$

$$W_{j \cong i} := \{x \in R_i(t) \cup R_j(t) : d(x, Cd(R_i(t))) = d(x, Cd(R_j(t))) \quad e \quad x \in C_i\}$$

In queste condizioni non si convergerà ad una partizione di Voronoi certoidale e quindi alla partizione ottima in assoluto, ma si otterrà una soluzione sub-ottima del problema, il meglio ottenibile.

Per questo Algoritmo sono valide le seguenti proposizioni:

**Proposizione 1**  $\forall t \in \mathbb{N}$ ,  $\{R_i(t)\}_{i=1}^N$  è un ricoprimento

**Proposizione 2** *L'indice  $H(\cdot)$  definito in (1.6) sulle le partizioni  $\{R_i(t)\}_{i=1}^N$  e i relativi centroidi  $Cd(R_i(t))$  è non crescente, cioè*

$$H(\{R_i(t+1)\}_{i=1}^N, \{Cd(R_i(t+1))\}_{i=1}^N) \leq H(\{R_i(t)\}_{i=1}^N, \{Cd(R_i(t))\}_{i=1}^N)$$

dove vale la disuguaglianza stretta se

$$W_{i \rightarrow j} \cup W_{j \rightarrow i} \neq \emptyset$$

## 2.2 Algoritmo di partizionamento ottimo con comunicazione simmetrica di tipo gossip (APSimm)

In questa sezione si introduce l'algoritmo di partizionamento ottimo con comunicazione simmetrica di tipo gossip.

Questo algoritmo è basato sul sistema di comunicazione simmetrico di tipo gossip, dove a ciascun round di comunicazione solo una coppia di vicini si scambia informazioni mentre le rimanenti telecamere non trasmettono né ricevono nulla.

L'algoritmo è implementato nel modo seguente:

Rispetto all'APsinc, **gli stati del processore e l'inizializzazione** rimangono invariati

**Trasmissione informazioni:** Per  $t \in \mathbb{N}$ , durante la  $t$ -esima iterazione dell'algoritmo, solo una coppia di telecamere vicine comunicano tra di loro, lo scambio dell'informazione avviene bi-direzionalmente, cioè se per esempio  $i$  e  $j$  comunicano allora  $i$  trasmette a  $j$  ma anche  $j$  trasmette a  $i$ ;

**Update dello stato:** A ciascun istante  $t$  c'è sempre una coppia di telecamere  $(i, j)$  che trasmette informazioni, la telecamera  $i$  riceve quindi  $R_j(t)$  e  $Cd(R_j(t))$  mentre  $j$  riceve  $R_i(t)$  e  $Cd(R_i(t))$ . Si compiono quindi le seguenti operazioni:

1. le telecamere computano gli insiemi:

$$W_{i \rightarrow j} := \{x \in R_i(t) : d(x, Cd(R_j(t))) < d(x, Cd(R_i(t)))\}$$

$$W_{j \rightarrow i} := \{x \in R_j(t) : d(x, Cd(R_i(t))) < d(x, Cd(R_j(t)))\}$$

$$W_{j \cong i} := \{x \in R_i(t) \cup R_j(t) : d(x, Cd(R_i(t))) = d(x, Cd(R_j(t)))\}$$

2. le telecamere aggiornano la loro regione e i relativi centroidi:

$$R_i(t+1) := ((R_i(t) - W_{i \rightarrow j}) \cup W_{j \rightarrow i}) \cup W_{j \cong i}$$

$$Cd(R_i(t+1)) = \min_{x \in R_i(t+1)} \{ \arg \min_{y \in R_i(t+1)} \sum d(x, y) \}$$



$$R_j(t+1) := ((R_j(t) - W_{j \rightarrow i}) \cup W_{j \rightarrow j}) - (R_j W_{j \cong i})$$

$$Cd(R_j(t+1)) = \min\{ \arg \min \{ \sum_{x \in R_j(t+1)} d(x, y) \} \}$$

In questo algoritmo è valida solo l'Osservazione 2 fatta nel paragrafo precedente. L'Osservazione 1 in questo caso non è vera perché  $R_i(t+1) \cap R_j(t+1) = \emptyset$ .

In APASimm sono valide anche le proposizione 1 e 2 del paragrafo precedente.

### 2.3 Algoritmo di partizionamento ottimo con comunicazione asimmetrica di tipo gossip (APASimm)

In questa sezione infine si introduce l'algoritmo di partizionamento ottimo con comunicazione asimmetrica di tipo gossip.

Questo algoritmo è basato sul sistema di comunicazione asimmetrico di tipo gossip, dove a ciascun round di comunicazione solo una telecamera manda informazioni ad uno dei suoi vicini.

L'algoritmo è implementato nel modo seguente:

Rispetto all'APSinc e all'APASimm, **gli stati del processore e l'inizializzazione** rimangono invariati

**Trasmissione informazioni:** Per  $t \in \mathbb{N}$ , c'è solo una telecamera che trasmette informazioni ad un suo vicino, per esempio la telecamera  $j$  trasmette le sue informazioni ad  $i$ ;

**Update dello stato:** A ciascun istante  $t$  la telecamera  $i$  riceve quindi  $R_j(t)$  e  $Cd(R_j(t))$  e compie le seguenti operazioni:

1. la telecamera computa gli insiemi:

$$W_{i \rightarrow j} := \{x \in R_i(t) \cap R_j(t) : d(x, Cd(R_j(t))) < d(x, Cd(R_i(t)))\}$$

$$W_{j \rightarrow i} := \{x \in R_j(t) : d(x, Cd(R_i(t))) < d(x, Cd(R_j(t)))\}$$

$$W_{j \cong i} := \{x \in R_i(t) \cup R_j(t) : d(x, Cd(R_i(t))) = d(x, Cd(R_j(t)))\}$$

2. la telecamera aggiorna la sua regione e il relativo centroide:

$$R_i(t+1) := ((R_i(t) - W_{i \rightarrow j}) \cup W_{j \rightarrow i}) \cup W_{j \cong i}$$

$$Cd(R_i(t+1)) = \min\{ \arg \min \{ \sum_{x \in R_i(t+1)} d(x, y) \} \}$$

**Osservazione 3** *Nell'APASimm si converge a regione con overlap, come nell'APsinc, quindi occorre modificare le equazioni di update se si vuol ottenere regioni senza overlap.*

*Alle  $N$  telecamere è associato un indice  $i \in 1, \dots, N$ , l'algoritmo si modifica nel seguente modo:*

- *se  $i$  riceve informazioni da una telecamera  $j$  e  $i < j$  allora*

$$R_i(t+1) := ((R_i(t) - W_{i \rightarrow j}) \cup W_{j \rightarrow i}) \cup W_{j \cong i}$$

- *altrimenti se  $j < i$*

$$R_i(t+1) := ((R_i(t) - W_{i \rightarrow j}) \cup W_{j \rightarrow i}) - (W_{j \cong i} \cap (R_i(t) \cap R_j(t)))$$

**Osservazione 4** *L'osservazione 2 è valida anche in questo caso con la sola eccezione che nell'insieme  $W_{i \rightarrow j}$  non serve controllare se l' $x$ , che soddisfa la disuguaglianza  $d(x, Cd(R_j(t))) < d(x, Cd(R_i(t)))$ , appartiene ad  $C_j$ . Infatti in questo caso, i punti che si considerano in  $W_{i \rightarrow j}$ , sono gli  $x \in R_i(t) \cap R_j(t)$  che già appartengono a  $C_j$ .*

In APASimm è verificata solamente la proposizione 1.

## Capitolo 3

# Soluzione proposta per il patrolling

Come si è già detto precedentemente lo scopo di questa parte del progetto è quella di trovare una soluzione ragionevole al problema del patrolling. Esistono numerose tecniche con cui è possibile svolgere la perlustrazione di un'area. Qui ci si limita a descrivere quella che è stata adottata e che permetta di perlustrare qualsiasi tipo di area, purchè sia connessa.

Innanzitutto bisogna premettere che l'area che ogni telecamera deve pattugliare è fornita in modo discreto (come descritto nel capitolo precedente). Dunque questa è descritta da un insieme finito di punti. Ogni punto è il centro del quadrato costituente la cella base della griglia utilizzata per suddividere la regione ed è distintivo di quella parte di area.

È necessario inoltre tenere in considerazione il campo visivo di ogni telecamera. In questa elaborazione viene considerato il campo visivo come un quadrato di lato  $dq$  approssimativamente simile all'area che viene visualizzata sullo schermo in fase di sorveglianza. Il lato  $dq$  è eventualmente ridimensionabile, vista la proprietà di zoom delle telecamere PTZ in uso, ma dovrà essere comunque scelto di dimensione adeguata allo scopo.

Per evitare problemi che potrebbero venirsi a creare nell'algoritmo di perlustrazione è stata adottata una convenzione: la griglia con cui si divide la regione iniziale  $R$ , e quindi strettamente correlata alla forma della regione  $R_i$  oggetto di pattugliamento da parte dell' $i$ -esima telecamera, deve avere la cella base con il lato del quadrato di circa il doppio di  $dq$ . Non si tratta assolutamente di una condizione gravosa e non vincola in alcun modo un qualsiasi altro aspetto del progetto.

Ogni telecamera è dotata di un processore in grado di elaborare dati. Quello che ci si propone è definire l'algoritmo che ogni telecamera deve utilizzare per compiere la perlustrazione completa di tutta la regione che le è stata assegnata. L'algoritmo deve essere tale da minimizzare ragionevolmente il gap temporale tra due visite successive di un punto appartenente alla regione.

Intuitivamente questo avviene se la traiettoria elaborata è ciclica.

**Dati in ingresso:** I dati da processare per arrivare alla definizione di una traiettoria sono quelli che descrivono una regione. Arrivando da una discretizzazione della regione  $R$ , il partizionamento ottimo porta alla creazione di sottoregioni  $R_i$  composte da affiancamenti di quadrati con almeno un lato in comune.

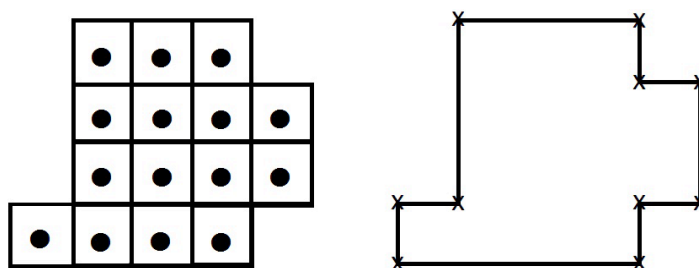


Figura 3.1: Descrizione di una regione discretizzata

Una regione quindi, non potendo avere lati curvilinei, può essere descritta elencando in ordine i vertici del poligono da cui è delimitata (vedi Figura 3.1). L'ordine può essere quello orario o quello antiorario, a seconda della convenzione che si decide di utilizzare.

**Elaborazione della regione:** La traiettoria che deve compiere il campo visivo è descritta da un insieme di punti sul piano da cui deve passare il centro del quadrato di lato  $dq$ . Questi punti devono essere tali da far sì che il passaggio su di essi garantisca la copertura dell'intera area. Diventa quindi importante come scegliere tali punti.

La regione  $R_i$  viene discretizzata in modo simile a quanto già fatto per l'intera regione  $R$ . Per rendere più chiara l'esposizione dell'algoritmo si può pensare che venga creata una *matrice delle posizioni* in cui la riga corrisponda all'ordinata del punto nel sistema dei due assi coordinati mentre la colonna rappresenti l'ascissa. Per esempio alla riga 5 si trovano tutti i punti  $(x, y)$  con  $y = k_y$  ed  $x = k_x + a * dq$   $a = 1, 2, 3, \dots, l$  dove  $l$  è la "larghezza discretizzata" della regione a quella latitudine. Alla regione  $R_i$  viene sovrapposta una griglia con celle quadrate di lato  $dq$ . Le rette componenti la griglia sono costruite parallele agli assi coordinati con cui sono descritte tutte le regioni. Se una cella ha al suo interno parte della regione  $R_i$  allora per il punto identificativo di quella cella, nella *matrice delle posizioni*, viene associato il valore 1 (e quindi la sua presenza). In caso contrario viene associato al punto il valore 0 (e quindi non viene considerato).

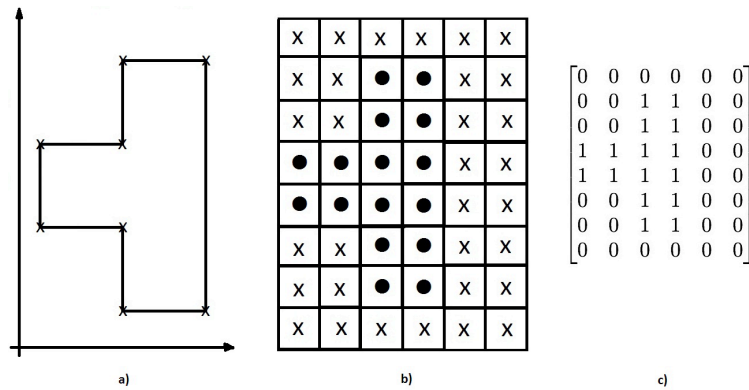


Figura 3.2: (a) Regione; (b) Griglia e discretizzazione della regione (c) Matrice delle posizioni

Come in precedenza, i punti identificativi della regione sono i centri dei quadrati delle celle base costituenti la griglia (Figura 3.2). Si viene a creare quindi una matrice di 1 e 0 che rimanda direttamente alla posizione dei punti su cui deve passare, o non serve che passi, il quadrato del campo visivo. Per costruzione, se in un ciclo la traiettoria passasse per tutti i punti con valore associato 1 allora si sarebbe certi della copertura totale della regione.

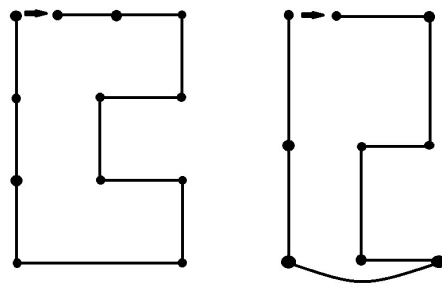


Figura 3.3: Possibile ciclo non ottimale in presenza di righe con almeno un valore 1 dispari nella matrice delle posizioni

**Osservazione 1** *La griglia va disposta in modo oculato: il numero di righe nella matrice delle posizioni in cui compare almeno un 1 deve essere pari in modo da evitare problemi di generazione di un ciclo non ottimale (vedi Figura 3.3). Scegliendo opportunamente le griglie questo problema non sussiste.*

Ora si dispone effettivamente dei punti su cui basare la traiettoria da seguire.

**Elaborazione della traiettoria:** Ciò che ora si vuole ottenere è un vettore che descriva l'“attraversamento” delle celle della *matrice delle posizioni* a valore 1. Viene creato un vettore  $v$  i cui elementi puntano alle celle della *matrice delle posizioni*  $M \in \mathbb{R}^{n \times m}$ . Per  $\frac{n}{2}$  volte, per tutte le righe dispari, partendo dalla riga  $i = 1$  della matrice  $M$  fino alla  $i = n - 1$ , inserisco in  $v$  gli la posizione degli elementi con valore 1 dal secondo all'ultimo dell' $i$ -esima riga e dall'ultimo al secondo della  $(i + 1)$ -esima riga. Si inseriscono ora in  $v$  tutti i primi elementi a valore 1 di ogni riga partendo dall'ultima fino alla prima.

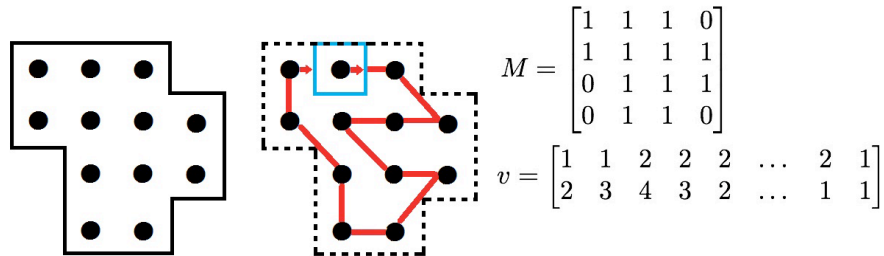


Figura 3.4: Illustrazione della strategia di patrolling utilizzata mediante la creazione di  $M$ , matrice delle posizioni, e del vettore  $v$  di attraversamento

Il vettore così creato descrive l'ordine in cui attraversare tutti i punti a valore 1 della *matrice delle posizioni*. La traiettoria è la spezzata originata dall'unione dei segmenti che collegano punti successivi nel vettore  $v$ . Per chiudere il ciclo bisogna considerare anche il segmento che unisce l'ultimo elemento di  $v$  al primo. (Figura 3.4)

La bontà o meno di un patrolling, considerando che tutte le telecamere muovano il campo visivo alla stessa velocità  $v_t$  con moto uniforme, si può osservare verificando la lunghezza della traiettoria  $l_t$ . Considerando una  $v_t = 1$  (indipendente dall'unità di misura), il *tempo di patrolling* che ne deriva è  $t_t = l_t$ .

**Osservazione 2** Questo algoritmo di attraversamento si basa su regioni connesse e senza buchi. L'algoritmo sarebbe da ripensare se una regione non fosse tale da verificare queste proprietà in quanto non sarebbe più una soluzione ragionevolmente ottimale.

**Osservazione 3** Due regioni con stessa area non è detto che abbiano la stesso tempo di patrolling. Questo dipende infatti da come si sviluppa la regione. Anche regioni simmetriche rispetto ad un asse potrebbero avere tempi di patrolling che differiscono leggermente.

**Osservazione 4** *Se non considerassi regioni discretizzate si potrebbe pensare di disporre la griglia in modo più accurato seguendo lo sviluppo della regione senza farsi condizionare dalla disposizione del sistema di assi cartesiani comuni a tutti. In pratica si dovrebbero trovare gli assi principali d'inerzia e creando una griglia in accordo con essi.*





## Capitolo 4

# Simulazioni

In questo capitolo verranno presentate le simulazioni che sono state effettuate, implementando i tre algoritmi appena presentati.

Nelle simulazioni che si presentano sono state prese in considerazione 5 telecamere che devono fare il patrolling su una regione quadrata discretizzata grande 20x20, vedi figura 4.1.

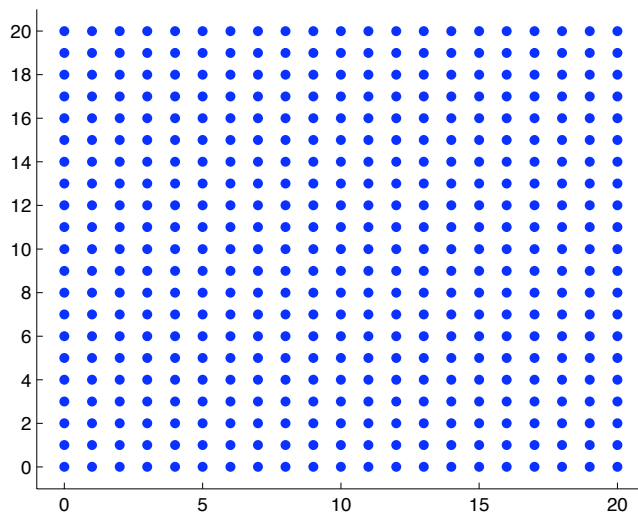


Figura 4.1: Regione discretizzata 20\*20

L'inizializzazione nei tre casi è comune. Si è deciso infatti di porre le  $R_i(0) = C_i \forall i \in \{1, \dots, 5\}$ . Le regioni iniziali associate ad ogni telecamera coincidono con il proprio constraint fisico (regione massima di possibile visibilità). Si sono quindi computati i centroidi iniziali  $Cd(R_i(0))$ .

Si è ottenuto il seguente ricoprimento iniziale, vedi Figura 4.2

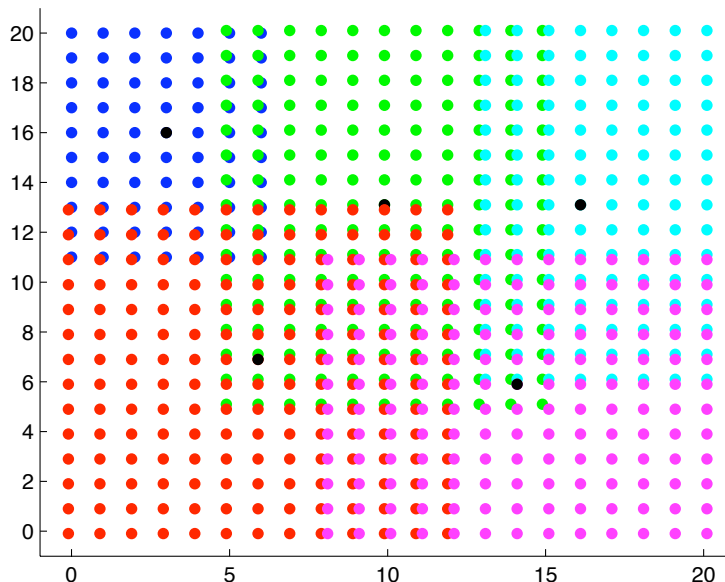


Figura 4.2: Inizializzazione del sistema

Come visto in precedenza, ogni telecamera trasmette e riceve informazioni. Il flusso di dati è regolato da un grafo  $G(V, E)$ . L'insieme dei vertici è:

$$V = \{i : i = 1, \dots, 5\}$$

l'insieme degli archi è:

$$E = \{(i, j) : i, j \in V \text{ e } C_i \cap C_j \neq \emptyset\}$$

cioè l'insieme delle coppie  $(i, j)$  tale che i constraints delle telecamere  $i$  e  $j$  condividono una porzione di regione.

Nel caso in esame, come osservabile in Figura 4.2, si considera:

- la regione blu  $\implies$  constraints telecamera 1
- la regione verde  $\implies$  constraints telecamera 2
- la regione azzurra  $\implies$  constraints telecamera 3
- la regione rossa  $\implies$  constraints telecamera 4
- la regione viola  $\implies$  constraints telecamera 5

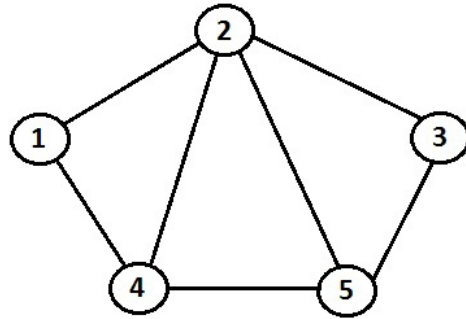


Figura 4.3: Grafo di comunicazione

Il grafo di comunicazione equivalente risulta quello in Figura 4.3

La rete di comunicazione, e quindi il grafo, è tempo invariante e comune per gli algoritmi di partizionamento; si considera inoltre la stessa condizione iniziale di partenza vista il figura 4.2.

Si passa ad illustrare i risultati ottenuti nei vari protocolli di comunicazione.

## 4.1 Simulazione APSinc

In questa sezione si presentano le simulazioni dell'algoritmo descritto nel paragrafo 2.1. Con riferimento alle osservazioni poste nel suddetto paragrafo si considera inizialmente l'algoritmo base e successivamente si implementano l'osservazione 1 e 2 rispettivamente per la condizione di non overlap e per il caso di constraints fisici.

Si sono simulate 15 iterazioni dell'algoritmo. Dalla terza iterazione l'APSinc converge e non modifica più le regioni delle 5 telecamere.

Da questa simulazione si possono fare tre importanti considerazioni:

1. si ottiene una partizione con overlap;
2. durante la dinamica dell'algoritmo si possono presentare delle partizioni con regioni sconnesse, ma l'algoritmo converge ad una partizione con regioni connesse;
3. dopo tre iterazioni l'algoritmo non modifica più il partizionamento, e quindi l'APSinc converge (per queste condizioni iniziali).

In figura 4.5 si presentano i risultati delle simulazioni effettuate considerando ora le osservazioni 1 e l'osservazione 2.

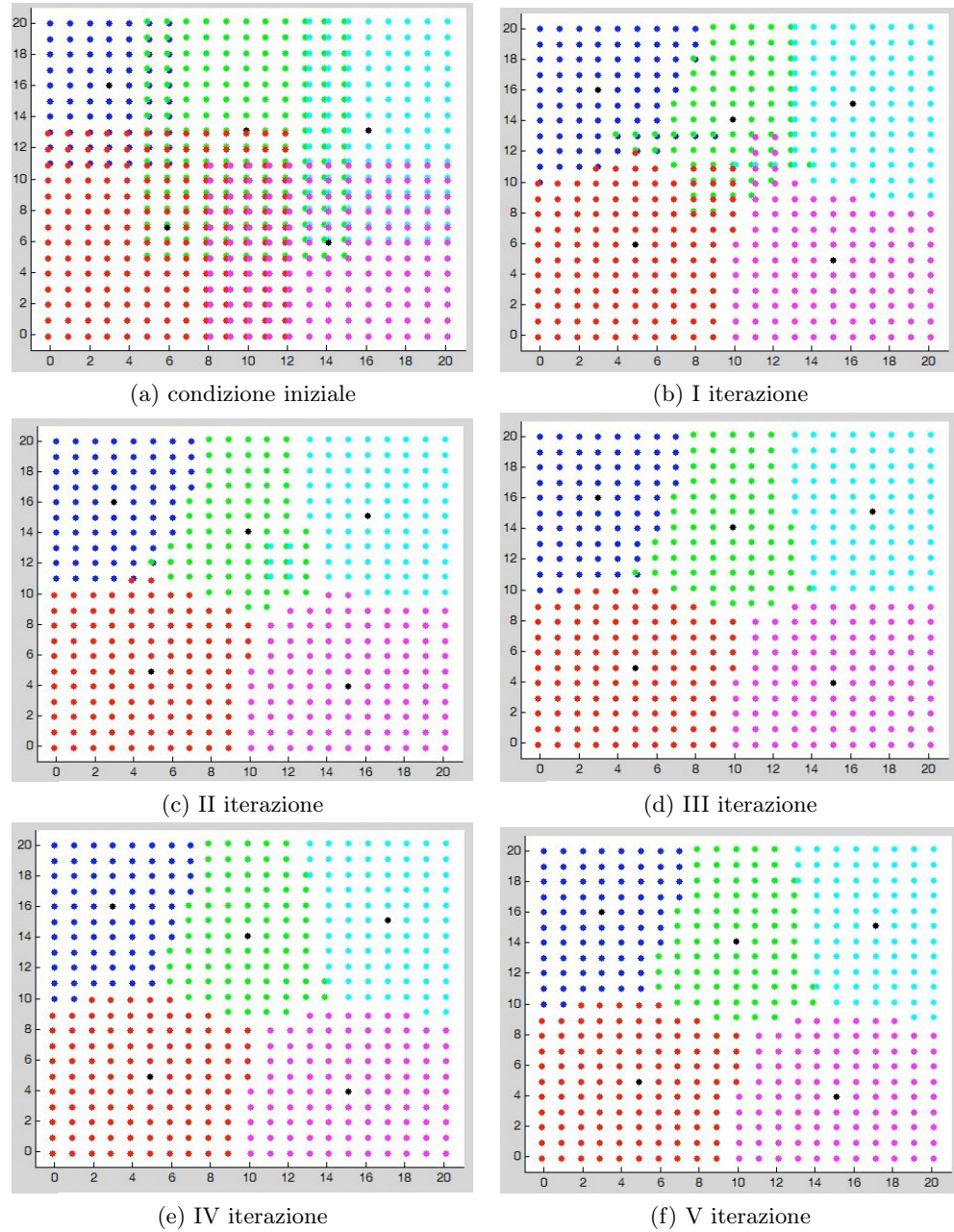


Figura 4.4: Risultati della simulazione dell' algoritmo APSinc

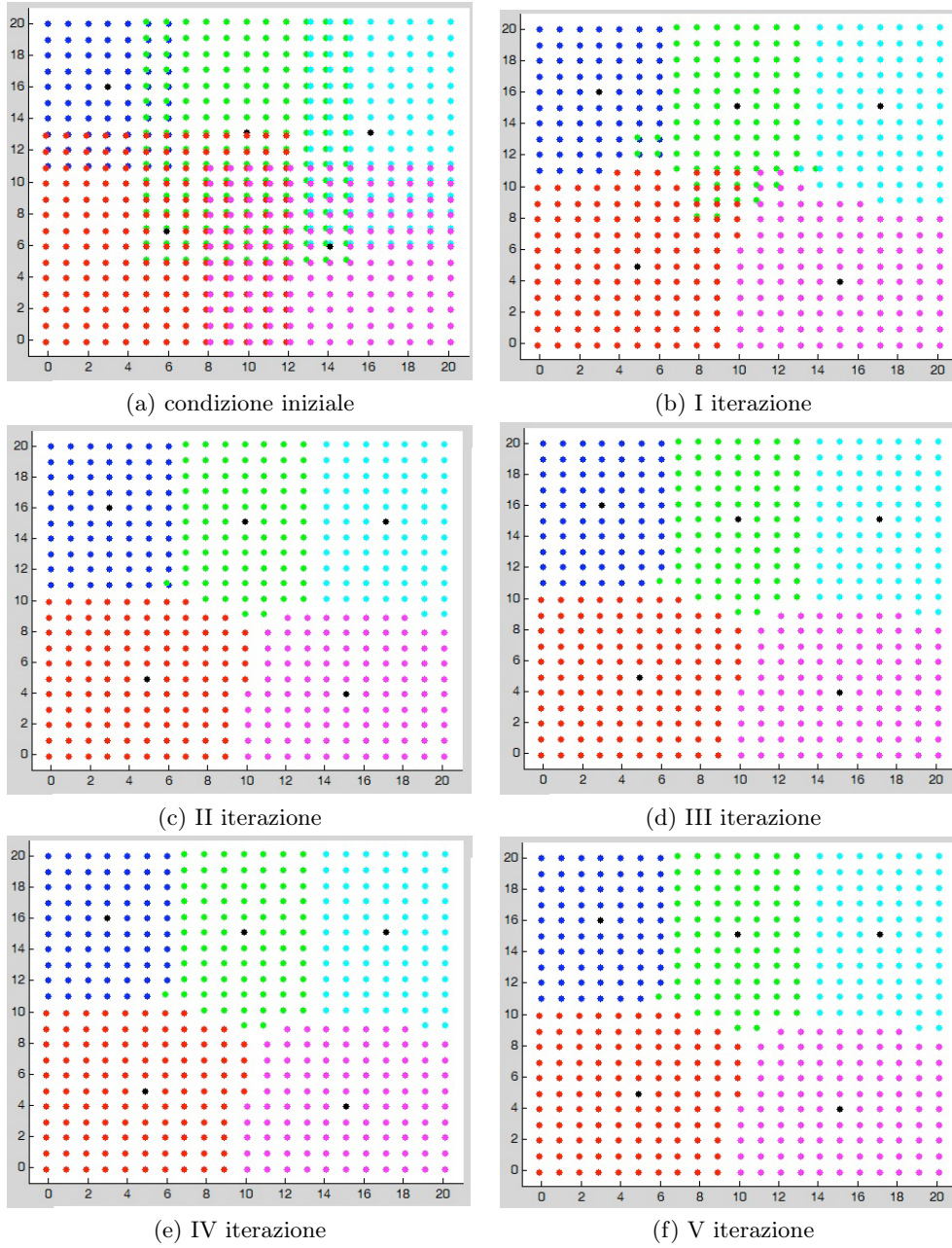


Figura 4.5: Risultati della simulazione dell'algoritmo APSinc considerando le osservazioni 1 e 2



Come nel precedente caso dopo la terza iterazione l'algoritmo computa sempre la stessa partizione.

Si può osservare che:

- L'APSync converge ad una partizione senza overlap e che rispetta i constraint fisici delle telecamere.
- La velocità di convergenza resta la stessa, rispetto alla precedente simulazione.

Come si può osservare dalla figura 4.6 che l'indice  $H$  definito in 1.6 è non crescente (come annunciato nella proposizione 2 del paragrafo 2.1).

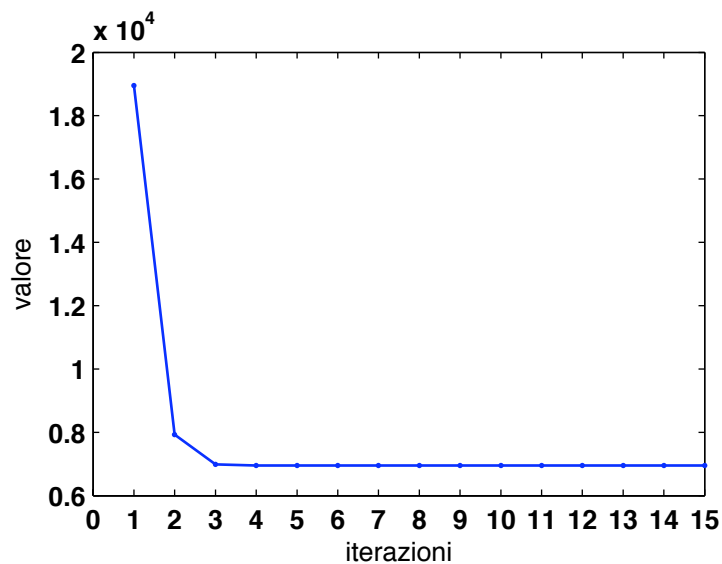


Figura 4.6: Andamento indice H

## 4.2 Simulazione APSimm

In questa sezione si analizza la simulazione dell'APSimm, in questo caso si sono prese in considerazione più iterazione dell'algoritmo (precisamente 40) in quanto la quantità di informazioni che circola sulla rete ad ogni round iterativo dell'algoritmo è più ridotto rispetto al caso sincrono. Si impiega infatti più tempo per partizionare la regione. Partendo dalla condizione iniziale di Figura 4.2 si sono ottenuti i risultati di Figura 4.7, sono stati riportati i partizionamenti più significativi dell'evoluzione dell'algoritmo.

La linea nera, all' $i$ -esima iterazione, individua la coppia di telecamere che si scambiano le informazioni e quindi le telecamere che aggiornano la propria regione.

In Figura 4.7 sono riportati i risultati ottenuti dall'APSimm già conside-

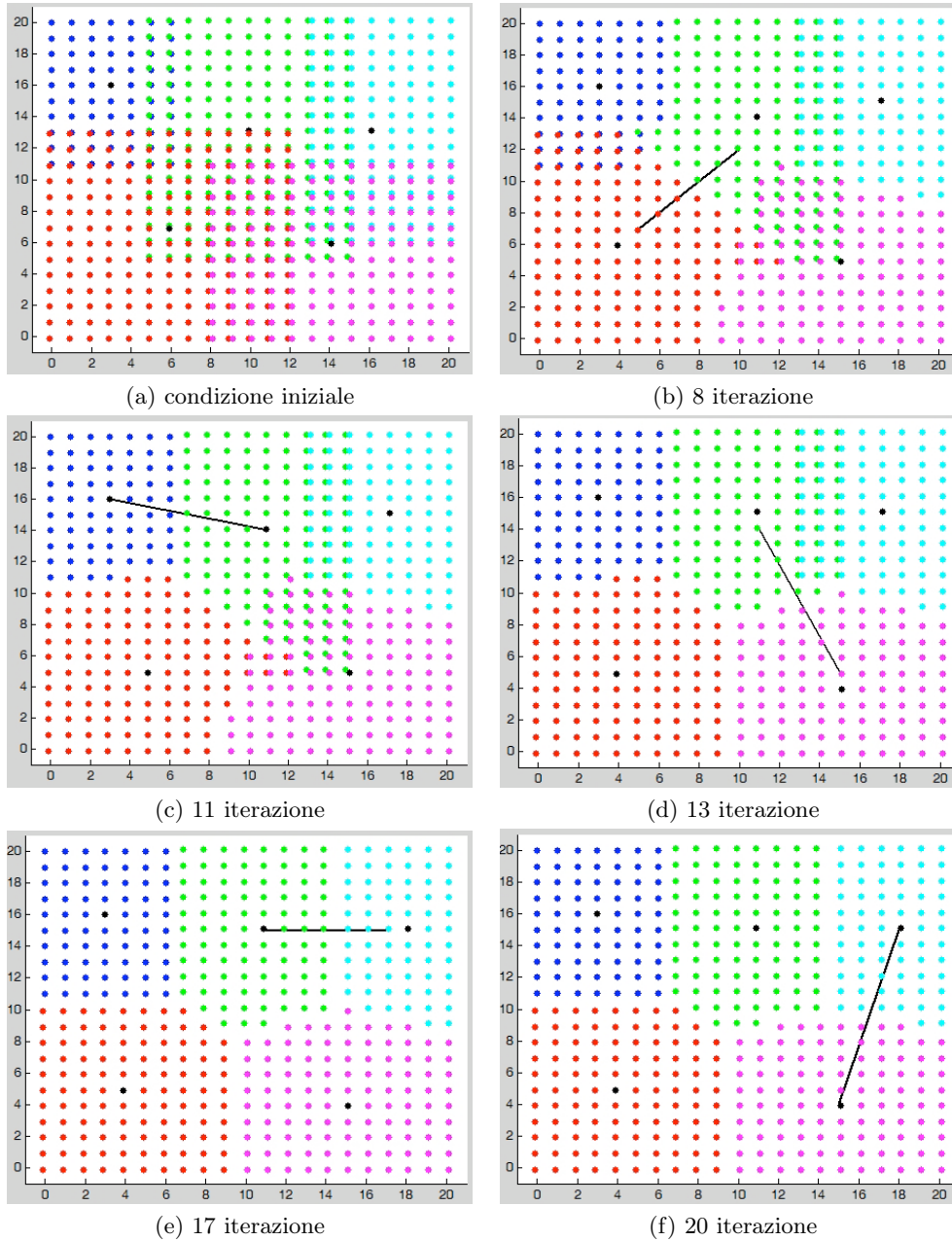


Figura 4.7: Risultati della simulazione dell'algoritmo APSimm

rando i constraints fisici delle telecamere.

Si può osservare che:

- il partizionamento è senza overlap;
- l'algoritmo converge dopo 20 iterazioni, valore non deterministico ma dipendente dall'aleatorietà della trasmissione (per assurdo potrebbe accadere che due telecamere vicine non scabino mai informazioni tra di loro);
- velocità di convergenza maggiore rispetto all'APSinc.
- l'aleatorietà delle trasmissioni potrebbe portare a convergere a partizionamenti differenti.

Come spiegato nel paragrafo 2.2, anche con questa tipologia di comunicazione, l'algoritmo presenta un andamento non crescente dell'indice H, come si vede in figura 4.8.

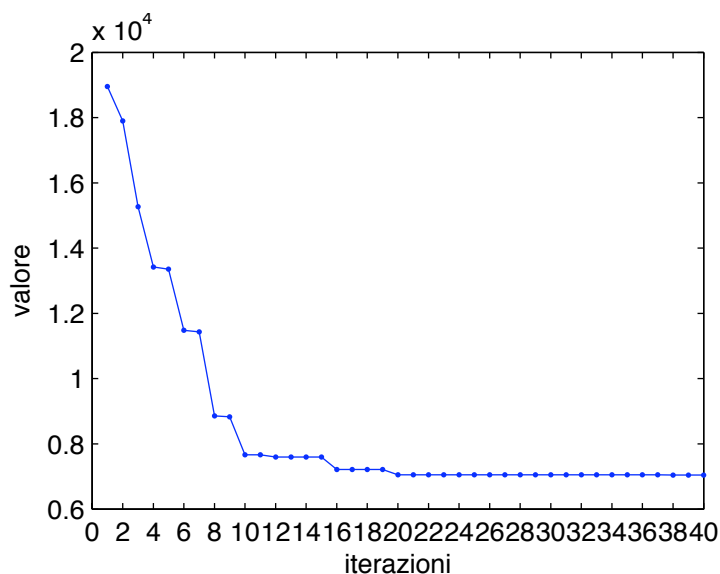


Figura 4.8: Andamento indice H

### 4.3 Simulazione APASimm

In questa sezione si presentano i risultati ottenuti nell'implementazione e simulazione dell'APASimm.

La comunicazione tra una coppia di vicini si differenzia da quella presentata in 4.2 per il fatto che la trasmissione avviene in una sola direzione. A causa di questa minor circolazione d'informazioni, nella simulazione si sono considerate 100 iterazioni dell'APASimm.



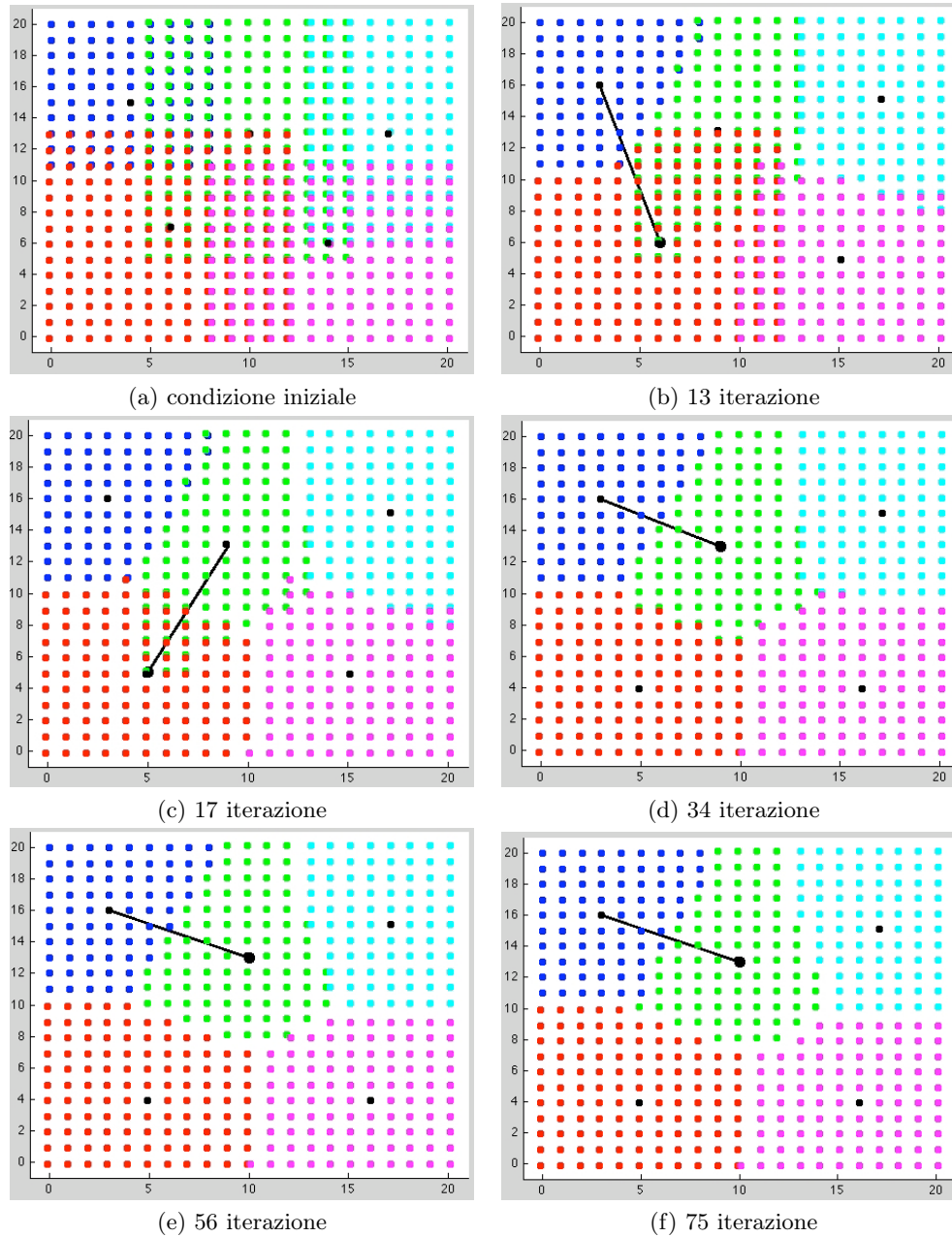


Figura 4.9: Risultati della simulazione dell'algoritmo APASimm

Anche in questo caso si riportino in Figura 4.9 le parti più significative dell'evoluzione del partizionamenti.

La linea nera in figura identifica la coppia di vicini coinvolta all' $i$ -esima iterazione ed il punto grosso indica chi, dei due nodi, riceve l'informazione aggiornando la propria area.

In questo caso si può vedere che:

- la velocità di convergenza è più bassa rispetto all'APSinc e all'AP-Simm;
- come nel caso dell'APSimm, l'aleatorietà con cui avviene la comunicazione non dà una certezza sul numero d'iterazioni per la convergenza. È chiaro che questo valore risulterà maggiore rispetto ai due casi precedenti. Nel caso in esame alla settantacinquesima iterazione l'algoritmo converge;
- l'aleatorietà delle trasmissioni potrebbe portare a convergere a partizionamenti differenti.

Infine si può notare dalla Figura 4.10 che l'indice  $H$  l'APASimm è localmente non crescente, ma non globalmente.

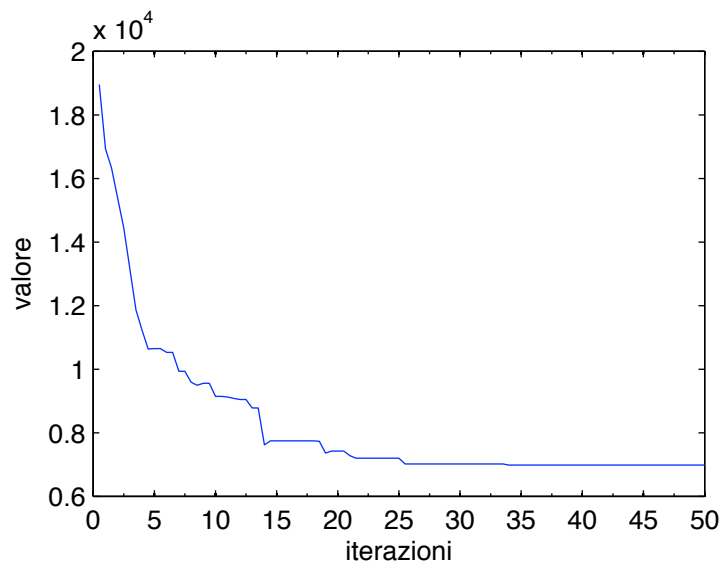
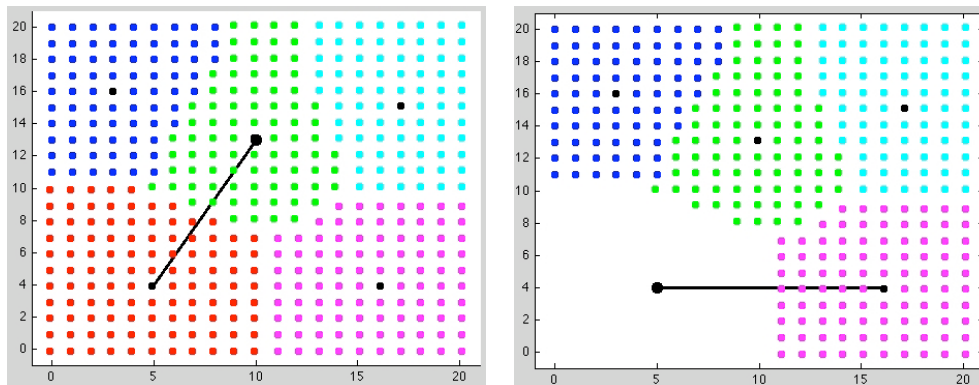


Figura 4.10: Andamento indice  $H$

## 4.4 Simulazione con telecamera impegnata in un altro task (caso APASimm)

Si è simulata la situazione in cui ad un certo istante  $t$ , una telecamera smette di perlustrare la sua regione in quanto gli è stata assegnata un'azione differente da eseguire, per esempio il tracking di un oggetto.

Si viene a creare una porzione di regione non perlustrata da nessuna telecamera (come si vede in Figura 4.11).



(a) tutte le telecamere eseguono il patrolling (b) telecamera impegnata in un altro task

Figura 4.11: Telecamera regione rosso ad un certo istante smette di perlustrare la sua regione

In questo caso si deve ridistribuire la parte non perlustrata tra le altre telecamere rispettando i constraint fisici. Può essere che non si riesca a ridistribuire l'intera porzione. In questo caso si avrà una zona non controllabile.

Per ridistribuire la regione non controllata, basta reinizializzare l'algoritmo considerando per le  $N - 1$  telecamere  $R_i(0) = C_i$ . Si computa ora nuovamente il partizionamento tramite l'APASimm.

I risultati riportati in Figura 4.12 sono le parti più significative dell'evoluzione del nuovo partizionamento della regione.

Si nota che il partizionamento si svolge in maniera regolare ed efficace.

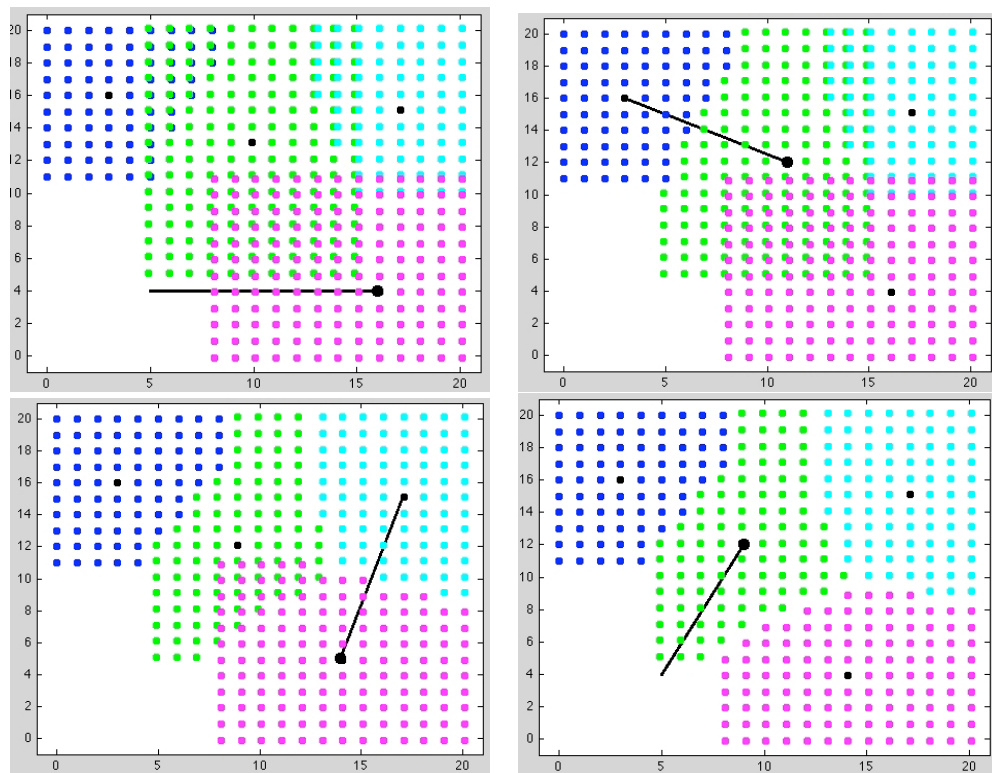


Figura 4.12: Simulazione redistribuzione dell'area con porzione di area non controllata da nessuna telecamera

## 4.5 Patrolling

Il patrolling viene effettuato da ogni telecamera su ogni singola sottoregione assegnatale come descritto nel capitolo 3. Si illustra nelle figure successive come viene eseguito il patrolling nelle simulazioni. Nel caso specifico lo si mostra quando sono state già ottimizzate le sottoregioni mediante *APA-Simm*. La regione totale  $R$  (quadrato  $20 \times 20$ ) è stata, come in precedenza, discretizzata da una griglia con celle quadrate di lato 1. Il  $dq$  del campo visivo è stato preso pari a 0.5.

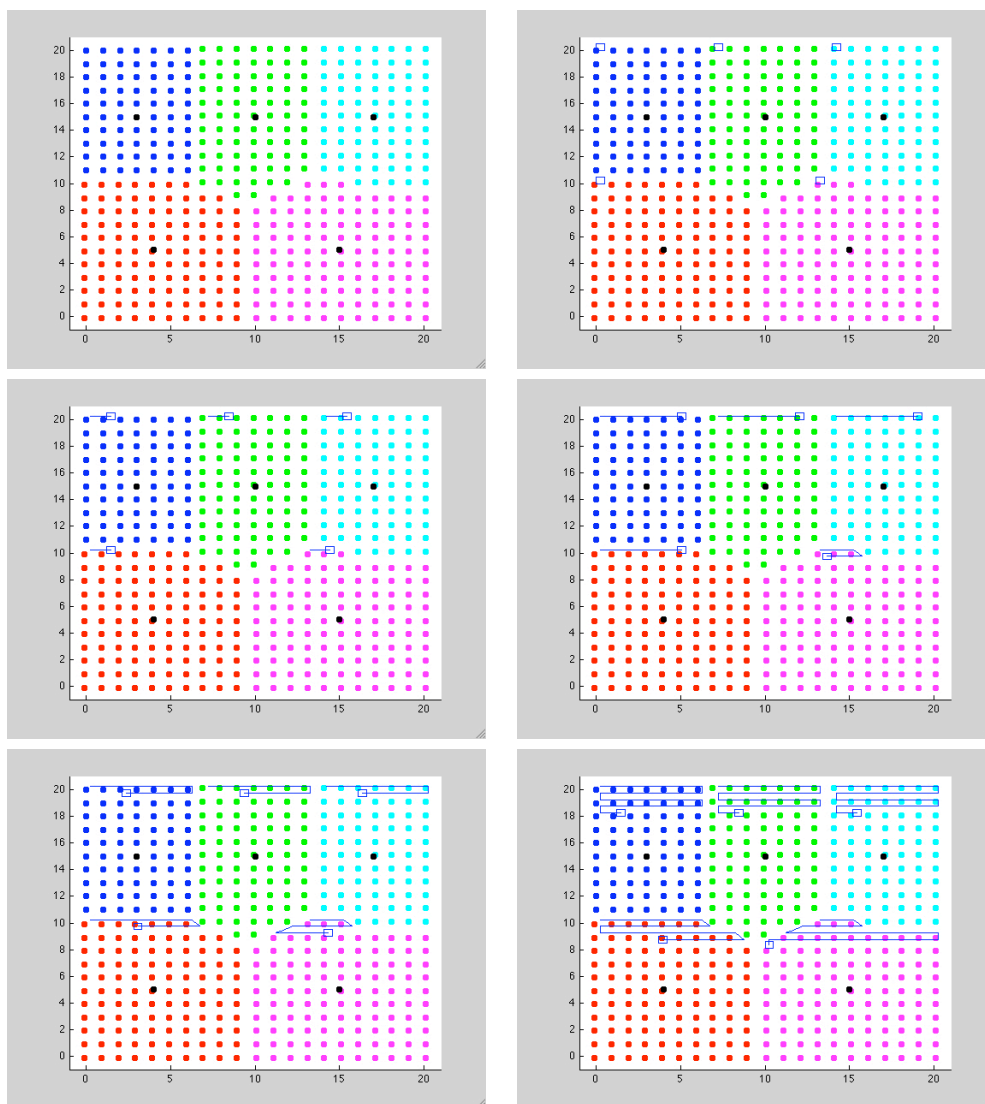


Figura 4.13: Patrolling (prima parte)

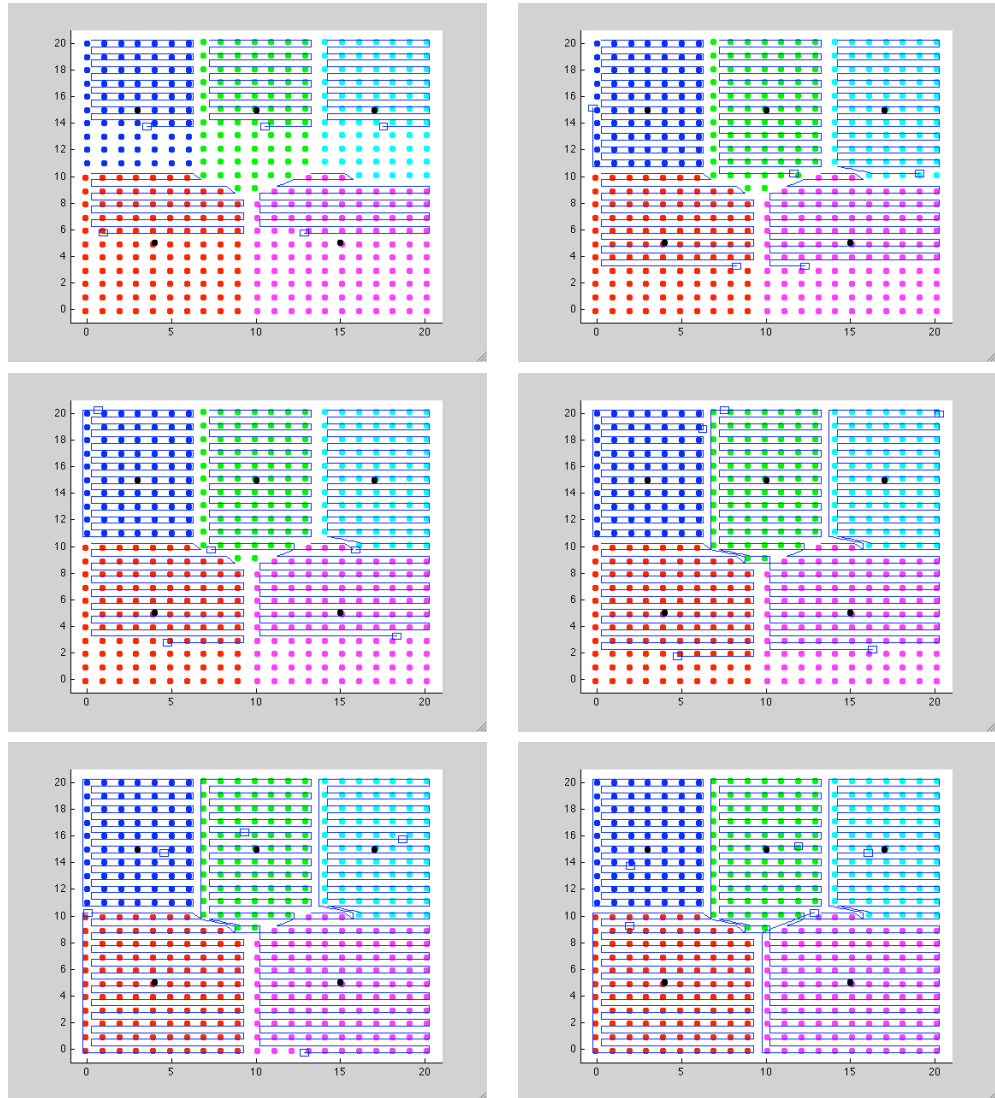


Figura 4.14: Patrolling (seconda parte)

Per verificare la bontà o meno del partizionamento effettuato è necessario valutare il costo che richiederebbe il patrolling dell'intera regione. Ad ogni iterazione viene valutato il costo del patrolling di ogni sottoregione. Si osservano quindi in ogni regione le lunghezze delle traiettorie di patrolling  $l_t$ . Queste infatti sono uguali ai tempi che si impiegherebbero nel compierle, viste le ipotesi assunte. Il patrolling dell'intera regione si considera completo quando tutte le sottoregioni sono state perlustrate completamente. Il costo del patrolling totale all' $i$ -esima iterazione, con un determinato ricoprimento della regione  $R$ , è quindi il tempo di patrolling maggiore tra i tempi di per-

lustrazione di tutte le sottoregioni.

Si possono considerare, data la condizione iniziale di partenza  $R_i(0) = C_i, i = 1, \dots, N$ , l'*upper bound* ed il *lower bound* del costo di patrolling.

L'*upper bound* coincide col costo totale assunto nell'istante iniziale, prima di avviare il partizionamento di  $R$ . Questo perchè inizialmente le sottoregioni sono nella situazione di coincidere con i loro constraint e quindi sono le maggiori che si possano mai ottenere.

Il *lower bound* invece si determina considerando la miglior area che possa essere creata: in una regione totale  $\Omega$  di area  $A$  da pattugliare con  $N$  telecamere, questa può essere associata ad una regione  $\Omega_i$  quadrata (o rettangolare) di area  $\frac{A}{N}$ . Il costo che deriva dal patrolling di  $\Omega_i$  coincide col *lower bound*.

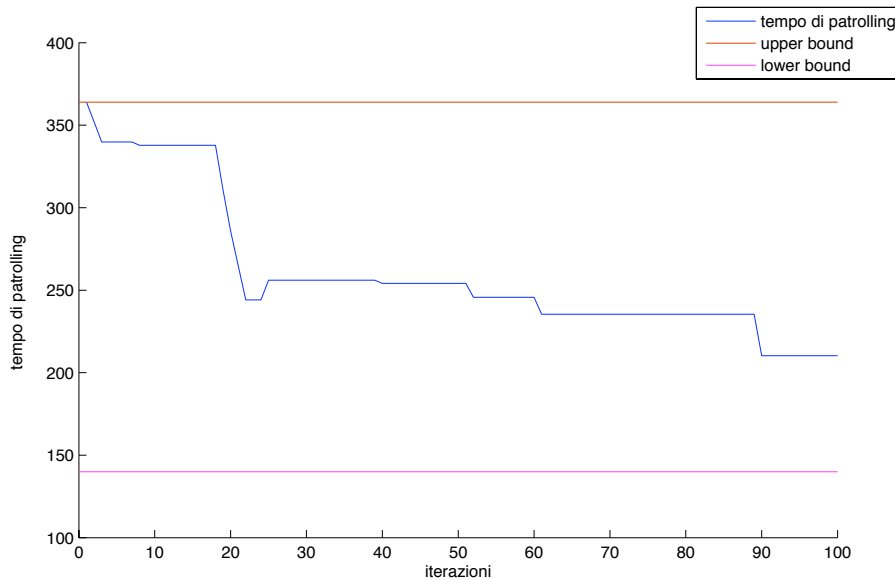


Figura 4.15: Costo del patrolling

Per non appesantire troppo la trattazione viene preso in considerazione solo il caso più ostico ossia quando viene utilizzato l'*APASimm*.

Nel caso di questa specifica simulazione, ma con valenza generale, il costo di patrolling (come si può notare in Figura 4.15) risulta essere tendenzialmente decrescente anche se localmente, in alcuni tratti, risulta essere non decrescente. Il valore del costo si stabilizza intorno alla novantesima iterazione convergendo ad un valore che è circa del 40% minore di quello iniziale e ovviamente rimane interno ai due bounds. L'aleatorietà della comunicazione tuttavia non permette di formulare leggi valide in assoluto ma su più simulazioni compiute questo valore non subisce importanti modifiche. Si

può comunque affermare che il costo minore appare quando si presenta il partizionamento ottimo della regione.

**Osservazione 1** *Problemi che si potrebbero riscontrare nel calcolo del costo di patrolling, utilizzando le metodologie adottate in questo lavoro, sono legati all'evoluzione delle sottoregioni nello svolgersi dell'algoritmo. Infatti talvolta potrebbe capitare che le regioni diventino momentaneamente non connesse per poi convergere ad una soluzione finale connessa. Come è già stato spiegato nel capitolo 3 la strategia di patrolling, in caso di regioni non connesse, dev'essere ripensata.*



## Capitolo 5

# Tipologie di partizionamento alternative

Esistono molte tecniche di partizionamento, alcune basate su regioni continue ed altre su regioni discretizzate.

In questo capitolo si accenna al problema dell'equitable partition, affrontato completamente in [3].

### 5.1 Equitable partition

Sia  $N$ , il numero di telecamere che devono perlustrare una regione compatta e convessa  $A \subseteq \mathbb{R}^d$ , sia inoltre  $\lambda$  una misura definita su  $A$  tale che:

$$\lambda \neq 0 \quad \forall x \in A$$

$$\lambda = 0 \quad \forall x \notin A$$

Per ogni insieme  $S$  noi definiamo il *carico di lavoro* per la regione  $S$  come

$$\lambda_S = \int_S \lambda(x) dx$$

Data una misura  $\lambda$  e una partizione  $\{A_i\}_{i=1}^N$  della regione  $A$ , si dice che la partizione è equa se

$$\lambda_i = \lambda_j \quad \forall i, j \in 1, \dots, N$$

L'algoritmo che determina l'equitable partition fa uso del Power Diagram, un'evoluzione del partizionamento di Voronoi.

Il Power Diagram è definito da una regione  $A \subseteq \mathbb{R}^d$  e un insieme di  $N$  punti

$$G_W = \{(g_i, w_i) : g_i \in A \quad e \quad w_i \in \mathbb{R} \quad \forall i \in 1, \dots, N\}$$

detti anche punti generatori.

Si definisce quindi il Power Diagram della regione  $A$  generato da  $G_W$  l'insieme

$V(G_W) = (V_1(G_W), \dots, V_N(G_W))$ , dove

$$V_i(G_W) = \{x \in A : \|x - g_i\|^2 - w_i \leq \|x - g_j\|^2 - w_j, \forall j \neq i, j \in 1, \dots, N\}$$

Ad ogni telecamera può essere associato quindi un punto generatore  $(g_i(t), w_i(t))$  e si può definire  $V(G_W(t)) = (V_1(G_W(t)), \dots, V_m(G_W(t)))$  dove ogni  $V_i(G_W(t))$  è la corrispondente sotto-regione dell' $i$ -esima telecamera.

Si introduce il funzionale costo  $H_V : S \rightarrow \mathbb{R}$

$$H_V(W) = \sum_{i=1}^N \left( \int_{V_i(W)} \lambda(x) dx \right)^{-1}$$

dove

$$S = \{(w_1, \dots, w_N) \in \mathbb{R}^N : \lambda_{V_i(G_W)} > 0, \forall i \quad e \quad \sum_i w_i = 0\}$$

e  $W = (w_1, \dots, w_N)$

**Lemma 1** *Il vettore  $\bar{w} = (\bar{w}_1, \dots, \bar{w}_N)$  che individua l'equitable partition, corrisponde al minimo globale di  $H_V$ .*

Data per ogni telecamera la condizione iniziale  $(g_i(0), w_i(0))$  che individua  $V(G_W(0))$ , l'equazione d'aggiornamento differenziale di primo grado

$$\dot{w}_i = -\frac{\partial H_V}{\partial w_i}(W)$$

a regime individua proprio gli  $(\bar{w}_1, \dots, \bar{w}_N)$  che generano l'equitable partition.

Si è tentato, in una prima fase del progetto, di risolvere il problema utilizzando l'equitable partition. Purtroppo problemi d'implementazione e la restrizione a regioni connesse ha reso questa strada impraticabile.

## Capitolo 6

# Conclusioni e possibili sviluppi futuri

In questo progetto si sono valutate metodologie diverse per il partizionamento attraverso due approcci totalmente differenti: il primo considera regioni continue sul piano (poligono convesso  $\Omega \subset R^2$  insieme infinito non numerabile di punti); il secondo, che si rivelato il più efficace, basato sulla discretizzazione delle regioni (come presentato nel paragrafo 1.1.3).

Vista l'ampia teoria solida e dimostrata presente in letteratura, gran parte del lavoro si è concentrato inizialmente sulle regioni continue. Giunti alla fase simulativa, ci si è scontrati però con il grosso problema di implementare algoritmi basati su computazioni di natura continua su sistemi che lavorano a tempo discreto. Le simulazioni che si sono ottenute sono abbastanza buone, ma sono basate su un approccio approssimativo e non dimostrato, difficilmente adattabile a sistemi scalabili; ci si è quindi concentrati su regioni discretizzate.

I tre algoritmi presentati per la soluzione nel caso di regioni discretizzate sono validi ed offrono una buona soluzione al problema del patrolling di ambienti bidimensionali. Ovviamente si utilizzerà uno piuttosto che l'altro a seconda delle necessità del caso. Infatti, talora, la scelta del tipo di comunicazione da effettuare tra le telecamere è imposta perché vincolata da scelte obbligate dovute alla situazione pratica particolare (restrizioni dovute ad eventuali fattori esterni). Avendo la possibilità di scelta, il caso in cui si presentano le migliori prestazioni è chiaramente quando viene utilizzata la comunicazione sincrona. Talvolta però quando il numero di telecamere è elevato un tipo di comunicazione sincrona risulta essere difficilmente applicabile. In questa circostanza comunicazioni di tipo asincrono sono l'ideale. In special modo la comunicazione asincrona asimmetrica risolve il problema sotto ogni punto di vista in quanto è adattabile ad ogni tipo di situazione risolvendo molte problematiche che nella pratica sono frequenti. Inoltre, come visto nelle simulazioni, è efficiente e robusta anche nel caso della perdita di una teleca-

mera per malfunzionamento o altro task (come un tracking).

Come considerazione personale noi riteniamo che l'approccio che dovrà essere perseguito in un futuro, e che probabilmente sarà anche il più indicato nella soluzione di questo problema, è quello che fa capo alle equitable partition. Orientandosi sulle equitable partition infatti sarebbe possibile partizionare la regione in sottoregioni sia molto più somiglianti tra loro che di aree molto simili. Questo renderebbe i costi di patrolling molto migliori in quanto probabilmente li avvicinerebbe maggiormente al lower bound.

Un'altra miglioria applicabile potrebbe essere quella riguardante le metodologie di calcolo della distanza tra i punti della regione (paragrafo 1.1.3). Questa potrebbe essere calcolata attraverso l'uso dei grafi. In questo modo si potrebbero considerare regioni non convesse e con buchi.

Infine si potrebbero modificare gli algoritmi presentati al fine di ottenere, nel transitorio dell'evoluzione del partizionamento, sempre regioni connesse.

# Bibliografia

- [1] R. Carli, A. Cenedese, and L. Schenato. Distributed partitioning strategies for perimeter patrolling, 20XX.
- [2] S. P. Lloyd. Least squares quantization in pcm. *IEEE Transactions on Information Theory*, 28(2):129–137, 1982.
- [3] M. Pavone, E. Frazzoli, and F. Bullo. Equitable partitioning policies for mobile robotic networks. *47th IEEE Conference on Decision and Control*, 2008.
- [4] J. Cortès F. Bullo and S. Martinez. *Distributed Control of Robotic Networks*. Princeton University Press, 2009.
- [5] J. Cortes, S. Martinez, T. Karatas, and F. Bullo. Coverage control for mobile sensing networks. *IEEE Transactions on Robotics and Automation*, 20(2):243–255, 2004.
- [6] C. Gao J. Cortès and F. Bullo. Notes on averaging over acyclic nodes on averaging over acyclic digraphs and discrete coverage control. *Automatica*, 44(8):2120–2127, 2008.
- [7] J. W. Durham, R. Carli, P. Frasca, and F. Bullo. Discrete partitioning and coverage control for gossiping robots. *IEEE Transactions on Robotics*, November 2010. Submitted.