# POLITECNICO DI MILANO

FACOLTÀ DI INGEGNERIA DELL'INFORMAZIONE

Corso di Laurea in Ingegneria delle Telecomunicazioni



**Codifica Multivista in Reti di Camere Wireless: Previsione del Guadagno e Analisi Energetica**


Relatore: Prof. Matteo Cesana
Correlatore: Ing. Alessandro Enrico Cesare Redondi

Tesi di laurea di:
Manuel MANDAGLIO
Matr. 797352



Anno Accademico 2015 - 2016

*Dedicata ad ognuno di Voi.*
*Voi, che mi accompagnate ogni giorno.*
*Voi, dagli sguardi dolci e vocine buffe con cui condivido ogni gioia, ansia e conquista della mia esistenza.*
*Voi, che insieme a me contribuite alla ricerca del perfetto stato di disordine.*
*Voi, che condividete le mie passioni anche se svuotano il portafoglio.*
*Voi, che se la mattina vi si parla la risposta è uno sguardo dolcemente severo.*
*Voi, che ricordate al vostro Manù di tosarsi regolarmente la barba.*
*Voi, che scassate le ***** tutte le mattine.*
*Voi, che avete condiviso con me un viaggio in treno e un discorso incoerente.*
*Voi, che ora vestite in completo, io vi ricorderò sempre con la maglietta degli Anthrax.*
*Voi, dalle dodici corde canterine.*
*Voi, con cui ho condiviso vacanze indimenticabili e avventure da ricordare.*
*Voi, delle immense gite in moto alla ricerca di un gelato.*
*Voi, delle levate alla mattina per una pesca infruttuosa.*
*Voi, speciali, che come me controllate tutto prima di ingerire.*
*Voi, dalle colorate calzamaglie rosse e nere.*
*Voi, del "Menù Manuel da 5 euro".*

*Che Voi siate un mio punto di riferimento o che sia io uno dei vostri.*

*M.*

# *Ringraziamenti*

*Ormai dopo un anno di fatiche, il mio lavoro si è concretizzato in tutte le pagine a seguire. Il percorso, che culmina oggi, è stato un viaggio lungo, a volte tormentato, che mi ha consentito di crescere, scoprendo me stesso ed altre bellissime persone.*

*Vorrei, prima di tutti, ringraziare la mia famiglia (cane al seguito) che mi ha sempre sorretto e dedicato il suo tempo, accettandomi per il testardo che sono.*

*Un pensiero speciale va sicuramente alla mia Giorgia, che negli ultimi anni si è subita ogni mia paturnia e sbalzo d'umore, motivandomi con quel bel sorriso che ti rende possibile ogni cosa.*

*Ringrazio tutti i miei più cari amici ed in particolare i miei compagni Marco e Simone con cui ho condiviso tutto della vita accademica e buona parte di quella quotidiana. Un grazie speciale a Simone per avermi sopportato sessione dopo sessione in ogni esame, accompagnando ogni mio appunto con uno "schizzo" di creatività anatomica ed uno anche a Marco, per aver letto la primissima (inumana) versione di questo lavoro.*

*Per ultimo, è dovuto, un ringraziamento al Professor Matteo Cesana, per avermi concesso di lavorare a questa tesi ed avermi affiancato l'Ingegnere Alessandro Redondi, che ha dedicato parte del suo tempo per rispondere alle mie domande (più o meno) intelligenti.*

*GRAZIE!*

# Prefazio

La comodità di avere sistemi altamente scalabili ed energicamente indipendenti ha condotto ad un massivo sviluppo dell'Internet delle Cose. L'ecosistema delle reti di sensori è vastissimo, troviamo molti ambiti applicativi, alcuni dei quali sono: domotica, automazione, indossabili e videosorveglianza.
Proprio la videosorveglianza è argomento di approfondimento in questa tesi. I sensori in grado di acquisire video ed immagini vengono utilizzati per numerosi scopi, tra cui, ricordiamo il riconoscimento di oggetti oppure smart-cities.

Le applicazioni precedenti sono il prodotto di un"“elaborazione video”. Nella fase di codifica le immagini del video vengono elaborate, identificando i punti d'interesse (*Keypoints*) che le costituiscono, fornendo una rappresentazione ugualmente accurata ma più leggera. I sensori camera codificano le immagini riducendo la mole di dati da inviare, diminuendo gli sprechi ed aumentando la propria vita.

Il nostro lavoro cerca di legare tutti i punti precedenti. Grazie alla capacità di codifica, le camere potranno elaborare le immagini prima di inviarle in rete. L'aggregazione sfrutta i possibili campi visivi sovrapposti (più camere inquadrano simultaneamente la stessa area) per risparmiare ulteriormente sui dati da inviare.

In una prima fase abbiamo analizzato le possibili combinazioni per trattare le immagini e codificarle in descrittori, identificando la configurazione ottimale per il lavoro da svolgere.
In seguito, abbiamo cercato di legare il guadagno energetico dell'aggregazione al livello di sovrapposizione tra i campi di vista. Il risultato ottenuto è un modello di previsione applicabile alle reti di sensori.
Infine, sono state simulate delle reti di camere con routing statico, ottimizzando

il numero di hop dalla radice. Il comportamento delle reti è stato studiato, nella configurazione prescelta, con tre diversi approcci, per valutare gli effetti sull'andamento energetico dei nodi costituenti.

# Indice

# Capitolo 1

# Introduzione

Nel panorama moderno delle reti, spicca inequivocabilmente il ruolo delle reti di sensori (*Wireless Sensor Network*, WSN). Nell'ambito dell'Internet delle cose (*Internet of Things*, IoT) esse si basano sul concetto di alta indipendenza energetica e sono costituite da nodi energicamente autosufficienti che mirano ad alleggerire la struttura delle reti classiche. L'alimentazione a batteria permette di essere molto vicino all'elemento da studiare senza dover investire in un sistema di supporto energetico o di posizionamento.

I nodi della rete sono tra loro indipendenti e comunicano utilizzando la famiglia di protocolli IEEE 802.15 a basso consumo energetico. Il numero di elementi che compongono la rete sono molto superiori alle classiche reti di calcolatori, fino a 20 nodi/$m^2$. Le comunicazioni sono solitamente in broadcast a tutti i nodi raggiungibili, ed al centro della rete è posizionato un nodo detto Sink (radice dell'intera rete), dove vengono convogliati i flussi informativi dei vari elementi. Per via della mole di energia spesa nella ricezione/invio di tutti i flussi di dati, che esaurirebbe in poco tempo una normale batteria, il Sink è connesso alla rete elettrica.
L'impiego delle batterie come fonte energetica è uno dei principali vantaggi delle WSN, ma ciò limita il tempo massimo di funzionamento dei nodi. Diventa essenziale un accorto e ben bilanciato utilizzo della risorsa energetica in modo da sfruttare al meglio i vari nodi. Per la maggior parte del tempo i nodi rimangono in una situazione di risparmio energetico fino a che l'evento da studiare li attivi,

scatti un timer di acquisizione o che il Sink interagisca con essi mediante "query".

Per via della loro scarsa longevità, i nodi sono apparecchi elettronici molto semplici con lo scopo di raccogliere dati ambientali (temperatura, umidità, posizione, immagini) elaborarli quanto basta per trasmetterli al proprio vicino, in modo da raggiungere il Sink. Il consumo energetico è, per la maggior parte, speso durante la trasmissione dei dati, in funzione della distanza tra i nodi, e dalla presenza o meno di ostacoli tra la sorgente e la destinazione.

Le WSN che utilizzano videocamere come sensori per l'accumulo e l'analisi di immagini e di video prendono il nome di *Wireless Multimedia Sensor Network* (WMSN), o anche più semplicemente, *Visual Sensor Network* (VSN), che saranno oggetto di studio in questa tesi.

Il metodo tradizionale di analisi video si fonda sul naturale modello umano di

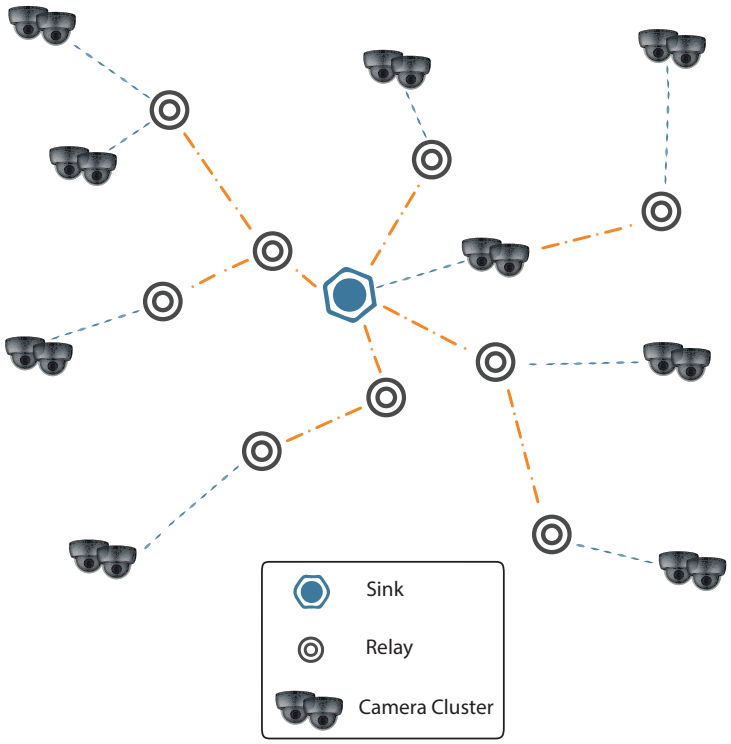



Figura 1.1: Esempio VSN

approccio visivo: gli stimoli visivi vengono acquisiti ed analizzati in modo da

poterli riassumere in concetti su cui costruire una efficiente comunicazione con altri individui. In modo molto simile, le reti di sensori camere gestiscono immagini e video secondo il paradigma *Compressione ed Analisi* (CTA): in una prima fase, i sensori raccolgono le immagini, o le serie di immagini, per poi inviarle al Sink opportunamente codificate. Solo dopo la ricezione, i dati vengono analizzati alla ricerca di pattern specifici per lo scopo richiesto (analisi delle masse, ricerca d'oggetti, riconoscimento del viso,...). Dal punto di vista energetico del sistema, l'approccio risulta poco efficiente.
Abbiamo già riportato poco sopra che la maggior parte della spesa energetica dei nodi è proprio nella fase di trasmissione, quindi, una fase di elaborazione correttamente implementata, potrebbe far risparmiare molta più energia di una trasmissione completa. Su questo semplice concetto si basa il modello di acquisizione *Analisi e Compressione* (ATC) che utilizzeremo in seguito. In questo modello l'analisi viene affrontata dai nodi ottenendo così, una più succinta e altrettanto valida rappresentazione dell'immagine e riducendo la quantità di dati da inviare senza perdere informazioni.

Per ottenere un elemento che rappresenta l'immagine vengono utilizzati i descrittori locali e i descrittori globali che affronteremo in dettaglio nel capitolo Background.
I descrittori locali, detti anche *Features*, vengono estratti dall'immagine in un processo a due fasi: in un primo momento si analizza l'immagine e da essa si estraggono i punti salienti (*Keypoints*); a seguire, nell'intorno di pixel dei *Keypoints*, si codificano tutte le proprietà fotometriche. Esistono diversi algoritmi che permettono di estrarre i descrittori locali: il più famoso è SIFT [1], su cui abbiamo basato il lavoro di tesi, ma si usano anche SURF [2], una versione più compatta, e BRISK [3], concepito espressamente per estrazioni efficienti a basso consumo energetico.
I descrittori globali invece, il cui approccio è molto diverso da quello dei primi locali, risultano ancora più compatti e applicano una quantizzazione sui descrittori locali in *Bag of Words* (BoW). Le varie *Features* vengono raggruppate in parole visuali, rappresentate poi sotto forma di istogramma (*Bag of Visual Words*, BoVW). Il risultato ottenuto è una rappresentazione dell'immagine di partenza con molti meno dettagli ma più snella e concentrata.

Un vantaggio importante dell'utilizzo delle feature locali è la loro praticità in caso di campi visivi sovrapposti.

Nelle reti VSN spesso si posizionano le camere con campi di vista sovrapposti: per aumentare l'accuratezza dell'analisi, o in caso di oggetti che occupano il campo visivo di una delle due camere, oppure per avere una rete robusta ai malfunzionamenti, come nel caso di una camera difettosa (in modo da avere l'altra camera che copra quasi completamente il campo visivo).
Nel caso *Analisi e Compressione*, le camere inviano al Sink una buona quantità di dati ridondanti, processo che risulta inefficiente in una rete che fonda il suo funzionamento sul risparmio energetico. L'utilizzo delle *Local Features* permette di eliminare questa ridondanza aggregando i descrittori simili delle viste. L'aggregazione permette un buon risparmio energetico in fase di trasmissione, quindi un consistente allungamento della vita della rete.

In passato è stata studiata, con buoni risultati, una codifica di tipo temporale, con lo scopo di ridurre le *Feature* duplicate tra i frame successivi temporalmente. Noi, invece, abbiamo voluto sfruttare la ridondanza nelle viste sovrapposte, in modo da contenere lo spreco energetico.
Grazie all'estrazione in *Feature* la quantità di dati da trasmettere è drasticamente diminuita rispetto a CTA, ma la presenza dei duplicati nelle viste comporta un traffico inutile.
Nella codifica multivista, una camera invia il proprio set di *Feature* alla camera che si occupa dell'aggregazione la quale trasmette i flussi aggregati solo a seguito di una valutazione positiva del guadagno di codifica multivista. Al contrario se l'aggregazione fosse al di sotto della soglia accettabile, il processo comporterebbe uno scambio di *Feature* inutile. Abbiamo quindi studiato un metodo che permettesse di evitare l'invio del set completo.
Nelle pagine che seguono approfondiremo il ruolo delle *Global Feature*, le quali permettono di quantizzare l'immagine rappresentandola con un piccolo elemento di qualche kilobyte, detto BoVW, per stimare il guadagno di codifica.

Dopo aver intuito il potenziale delle *Global Feature* abbiamo legato il guadagno di codifica al grado di somiglianza tra istogrammi in diverse configurazioni.

In ogni configurazione abbiamo analizzato il grado di correlazione tra guadagno e distanza: variando il numero di parole visuali di riferimento, convertendo in forma binaria i BoVW, calcolando in diversi modi le distanze tra istogrammi ed infine dividendo l'immagine in aree rettangolari uniformi (settori) più facili da rappresentare.
Una volta in possesso della funzione in grado di stimare il guadagno abbiamo simulato delle VSN in modo da valutare il nostro approccio. Lo scopo della simulazione è comprendere quanto la nostra idea, una volta applicata, consentisse alle rete di risparmiare energia nei punti critici (bottleneck), in modo da allungarne l'operabilità.

La tesi sarà strutturata come segue. Una prima parte di Background (Capitolo 2) approfondiremo nel dettaglio gli strumenti che ci hanno permesso di studiare il problema. A seguire un'introduzione ai diversi approcci alla cooperazione delle VSN (Capitolo 3). Nel Capitolo 4, approfondiremo l'utilità di un codifica multivista invece della classica codifica temporale, e ricercheremo il legame tra distanza di istogrammi visuali e guadagni per trasmissioni aggregate. Nel capitolo 5, applicheremo la funzione che descrive questo andamento alle reti simulate, in modo da esaminare l'andamento dei livelli energetici dei cluster di camere e del bottleneck. Concluderemo con i risultati ottenuti dalle nostre analisi.

# Capitolo 2

# Background

In questo capitolo esamineremo tutti gli strumenti utilizzati nello svolgimento della tesi, in modo da poter facilitare la lettura della stessa e di evitare inutili ripetizioni.
Vedremo la definizione e le fasi di estrazione delle Feature Locali e i principali algoritmi atti alla loro codifica dalle immagini di partenza. Successivamente spiegheremo il ruolo delle Feature Globali e la loro derivazione a partire da quelle Locali. In ultimo studieremo la codifica temporale, da cui abbiamo basato i nostri studi per verificare l'efficacia della codifica multivista per viste sovrapposte.

## 2.1 Feature Locali

Partiamo dal concetto che una corretta e assodata definizione di Feature Locali non esiste. Possiamo definire Feature Locali alcuni punti focali dell'immagine, cioè dove l'immagine cambia bruscamente: un vertice, un contorno che definisce un netto stacco da uno sfondo,... Questi elementi, detti *Keypoints*, talvolta non sono definibili come un singolo pixel ma possono essere piccoli o grandi gruppi che indichereranno la posizione della *Feature* nell'immagine.
Ad ognuno dei *Keypoints* viene poi associato un *vettore di features*, o più semplicemente un *descrittore*, che codifica in forma numerica i pixel nell'intorno del *Keypoint*. La codifica dovrà essere robusta alle classiche deformazioni come: scalatura, cambiamento del punto di vista (traslazione), illuminazione e rotazione.
Gli strumenti che ci consentono di estrarre tutte questi elementi sono definiti

*detectors*. Nel seguito approfondiremo l'algoritmo SIFT e accenneremo a SURF e BRISK.

### 2.1.1 Scale Invariant Feature Transform (SIFT)

Il meccanismo SIFT fu pubblicato per la prima volta da David Lowe nel 1999 [1], da quel giorno viene considerato il punto di riferimento per i vari algoritmi di estrazione di feature ed è quindi importante riuscire a capire al meglio il suo funzionamento. Vedremo nel seguito le fasi principale dell'algoritmo di *Detection* (sono in totale 3) e a seguire la *Description*

#### *SIFT Detection*

In questa prima fase andremo a studiare come l'algoritmo SIFT permette di identificare i *Keypoints*, rendendoli robusti alle deformazioni spaziali (traslazione, ridimensionamento e rotazione). Il processo di definizione del descrittore si compone di innumerevoli passaggi come possiamo vedere in Figura 2.1.

La prima fase di Lowe è la costruzione del *scale-space*, ottenuto come la convolu-

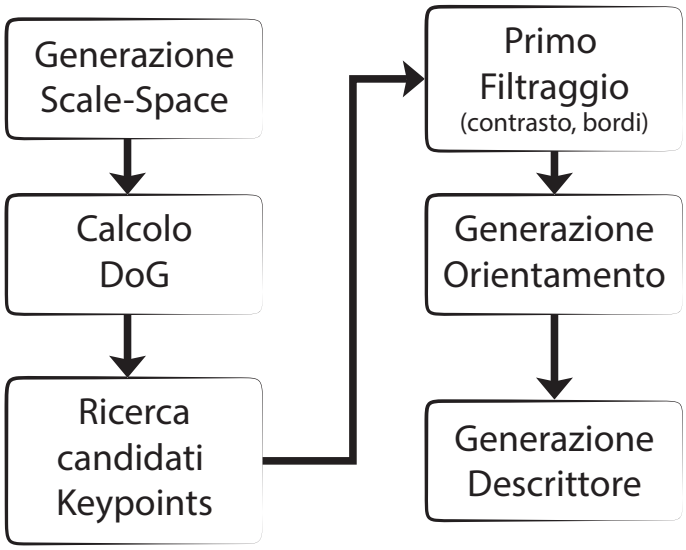



Figura 2.1: Diagramma a blocchi dell'algoritmo

zione tra una funzione Gaussiana $G(x, y, \sigma)$ e l'immagine di partenza $I(x, y)$, da cui otteniamo la funzione convoluta $L(x, y, \sigma) = G(x, y, \sigma) * I(x, y)$. Allo scopo

di aumentare la robustezza dell'algoritmo, Lowe cercò di approssimare il risultato al Laplaciano di Gaussiani [4] andando a convolvere la differenza tra la funzione $G$ e se stessa traslata di un certo fattore $k$, il risultato ottenuto è indicato come Difference-of-Gaussian (DoG). Una volta convoluto con $I$ otteniamo il valore della funzione *D(x, y, σ)*. Per chiarezza si faccia riferimento all'equazione 2.1:

$$\begin{aligned} G(x, y, \sigma) &= 1/(2\pi\sigma^2) \exp^{\frac{x^2+y^2}{\sigma^2}} \\ DoG &= G(x, y, k\sigma) - G(x, y, \sigma) \\ D(x, y, \sigma) &= DoG * I(x, y) \\ &= L(x, y, k\sigma) - L(x, y, \sigma) \end{aligned} \tag{2.1}$$

Vediamo ora il processo seguito dall'algoritmo per produrre i DoG e poi i *Keypoints*. Partendo dall'immagine e dalla funzione Gaussiana, i vari *L(x, y, σ)* vengono calcolati in gruppi di 8, detti ottave (come si vede nella parte sinistre dell'immagine 2.2), convolvendo la stessa immagine per una funzione Gaussiana scalata di un fattore k dalla precedente. Per ottenere i punti d'interesse dallo *scale-space* dobbiamo sottrarre a due a due le varie convoluzioni successive (utilizzando indirettamente l'approccio dei DoG) ottenendo, infine, i valori della funzione *D(x, y, σ)* (parte destra dell'immagine 2.2)). Una volta che il processo è concluso, la *L(x, y, σ)* calcolata con con la varianza pari al doppio dell'originale viene sottocampionata di un fattore 2 e presa come partenza per l'ottava successiva, ripetendo l'intero ciclo.

Una volta in possesso dei valori di *D(x, y, σ)* ogni valore campionato viene confrontato con gli 8 vicini nello stesso livello (vedere Fig. 2.3), con i 9 nel livello superiore e i 9 in quello inferiore. Il punto in analisi sarà definito *Keypoint* se il suo valore sarà il massimo o il minimo tra tutti.

Il processo precedente ha prodotto in uscita una grandissima quantità di *Keypoints*, ma non tutti sono robusti e possono tornare utili nella caratterizzazione dell'immagine. Di conseguenza, si eliminano quelli con basso contrasto e gli outlayer prodotti da disturbi nell'immagine.

Ultimo passaggio per fase di *Detection* è la definizione di orientamento. Per ogni campione di *L(x, y, σ)* vengono calcolati magnitudo e orientamento del

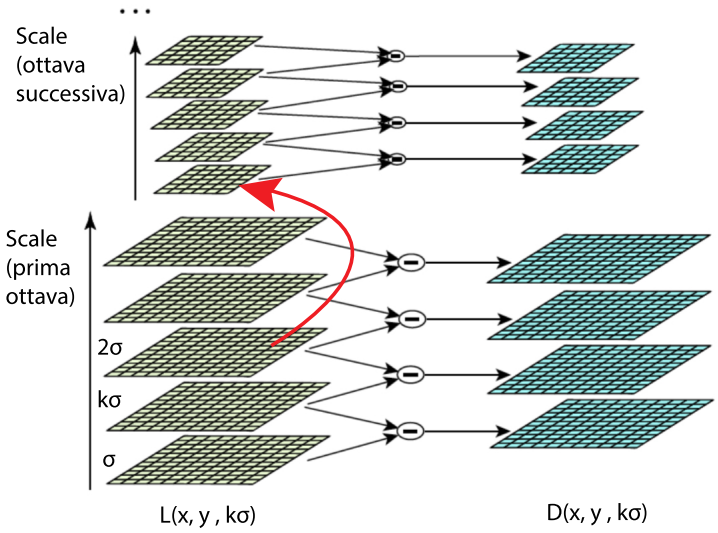



Figura 2.2: Costruzione della piramide Scale-Space

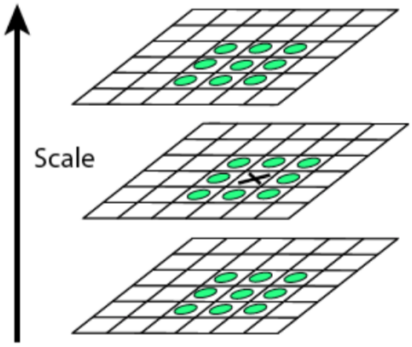



Figura 2.3: Selezione Keypoints

gradiente come in formula 2.2:

$$
\begin{aligned}
m(x,y) &= \sqrt{(L(x+1,y) - L(x-1,y)^2 + (L(x,y+1) - L(x,y-1)^2} \\
\theta(x,y) &= \tan^{-1}\left(\frac{L(x,y+1) - L(x,y-1)^2}{L(x+1,y) - L(x-1,y)^2}\right)
\end{aligned}
\tag{2.2}
$$

A partire dall'orientamento del gradiente dei campioni nell'intorno del *Keypoint* viene creato un istogramma con 36 colonne, detto *istogramma degli orientamenti* (una colonna per ogni 10 gradi, quindi un totale di 360 gradi).
Gli elementi aggiunti all'istogramma, vengono prima pesati per il valore del proprio gradiente.

L'orientamento finale viene tracciato individuando il valore massimo tra i picchi dell'istogramma ed i suoi 2 più vicini. Per avere una maggiore accuratezza i tre valori sono interpolati con una funzione parabolica.

### *SIFT Descriptor*

Abbiamo già detto che la fase precedente ci permette di definire dei *Keypoints* robusti alle deformazione morfologiche, ma una trasformazione importante a cui deve resistere è la variazione di illuminazione, ed è proprio nella fase di *Description* che li renderemo tali. Per il calcolo del descrittore vengono calcolati nuovamente la magnitudo e l'orientamento dei gradienti dei campioni nell'intorno del *Keypoint*. Per ottenere invarianza alla rotazione, le coordinate del descrittore vengono ruotate in funzione del *Keypoint* come in Figura 2.4.

Successivamente, i valori vengono normalizzati per una funzione Gaussiana che

Figura 2.4: Rotazione dei descrittori in funzione dell'orientamento

possiamo vedere in blu nella Figura 2.5, per poi essere riassunti finalmente nel descrittore.

Il descrittore non è altro che un vettore di istogrammi che riassumono il contenuto delle regioni che rappresentano. Ogni colonna contiene la somma delle magnitudo dei gradiente nelle direzioni che rappresentano.

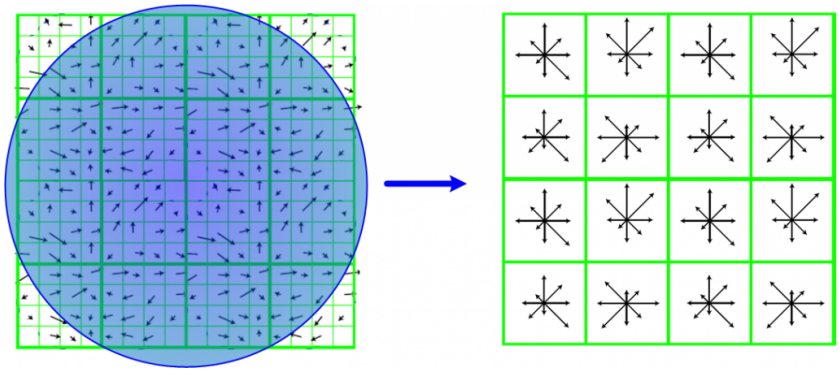


Figura 2.5: Creazione del Descrittore SIFT

Come ultimo passaggio per ridurre gli effetti di luminosità, i valori trovati vengono normalizzati per la norma del vettore.
Il descrittore sarà composto da un *feature element* di 4x4x8 elementi(un totale di 128) per ogni *Keypoints*.

### 2.1.2 Speeded Up Robust Features (SURF)

L'algoritmo SURF, fu proposto nel 2006 da Helbert Bay [5], si basa sugli stessi principi di SIFT ottenendo prestazioni inferiori, ma risulta molto più veloce del precedente.

#### *SURF Detection*

Il Detector SURF al contrario di SIFT, si basa sul calcolo della matrice Hessiana ($\mathcal{H}$) [2] o meglio, su una sua approssimazione. Il calcolo della matrice $\mathcal{H}$ deriva dalla convoluzione delle derivate seconde tra una funzione Gaussiana e l'immagine, come possiamo più semplicemente vedere nella formula 2.3:

$$\mathcal{H}(x,y,\sigma) = \begin{bmatrix} \frac{\partial^2}{\partial x^2}G(\sigma) * I(x,y) & \frac{\partial}{\partial x}\frac{\partial}{\partial y}G(\sigma) * I(x,y) \\ \frac{\partial}{\partial x}\frac{\partial}{\partial y}G(\sigma) * I(x,y) & \frac{\partial^2}{\partial y^2}G(\sigma) * I(x,y) \end{bmatrix} \tag{2.3}$$

Per sopperire al calcolo oneroso della derivata, si utilizzano filtri bidimensionali (*box filters*) applicati a immagini integrali. Il risultato è un'ottima approssimazione della matrice precedente calcolata. Calcoliamo, infine, il determinante della matrice $\mathcal{H}$ come nell'equazione 2.4:

$$det(\tilde{\mathcal{H}}) = D_{xx}D_{yy} - (0.9D_{xy})^2 \tag{2.4}$$

Le fasi di localizzazione dei *Keypoints* comprendono, come per SIFT, la creazione di un *scale-space* che al contrario del primo caso, verrà prodotto applicando all'immagine filtri bidimensionali (*box filters*) di dimensioni variabili. I punti selezionati verranno presi in considerazione se avranno il valore massimo tra i 3x3x3 vicini nello scale-space.
L'orientamento è la fase successiva a quella di detection, che risulta molto simile a quella già descritto precedentemente in SIFT. La differenze tra le due metodologie si basano sull'utilizzo in SURF delle wavelet Haar, ottenendo l'orientamento come il valore massimo tra le somme dei valori di Haar ogni $\pi/3$.

### *SURF Descriptor*

Nella fase di *Description*, viene posizionata un'area quadrata centrata alle coordinate del *Keypoint* ed orientata in funzione dell'orientamento del *Keypoint* stesso. L'area viene divisa in 4x4 zone suddivise nuovamente in 25 sottocampioni di dimensioni proporzionali. Per ogni punto dei sottocampioni vengono calcolati i valori Haar, indicandoli con $d_x$ e $d_y$.
I valori calcolati vengono sommati per sottoregioni arricchendo un *vettore di feature* ($f$), aggiungendo anche la somma dei valori assoluti di $d_x$ e $d_y$. Ogni sottoregione avrà un vettore come quello in 2.5:

$$f = \left[\sum d_x, \sum d_y, \sum |d_x|, \sum |d_y|\right] \tag{2.5}$$

Il descrittore finale sarà una concatenazione di tutti i vettori delle sottoregioni, ottenendo quindi 4x4 vettori di feature ognuno con 4 elementi (un totale di 64 elementi per *Keypoint*). Ultimo passaggio per conferire robustezza al fattore di illuminazione è la normalizzazione del vettore descrittore per la sua norma.

### 2.1.3 Binary Robust Invariant Scalable Keypoints (BRISK)

Nel corso degli ultimi anni la comparsa di dispositivi a basso consumo energetico ha spinto alla ricerca di nuove tipologie di algoritmi per l'estrazione di feature che potessero essere computazionalmente efficienti, ottenendo comunque un'alta qualità. Studiamo l'algoritmo BRISK proposto da Leutenegger nel 2011 [3], esaminando i blocchi dell'algoritmo per evidenziare l'ottima velocità, dovuta essenzialmente a un diverso approccio dello *scale-space*.

#### *BRISK Detector and Descriptor*

Il core del detector di BRISK è ispirato ai detectors di vertici FAST [6] ed alla sua evoluzione AGAST [7], entrambi computazionalmente molto efficienti.
Un punto candidato $p$ ha una propria intensità $I_p$ e viene classificato come vertice (*corner*) se $n$ pixel vicini hanno un valore di intensità pari a $I_p + t$ o $I_p - t$ (dove $t$ indica una ben definita soglia). In base ai confronti con le intensità dei pixel vicini al punto $p$, viene attribuito un punteggio $s$ pari alla soglia massima raggiunta nei confronti.
Il punto analizzato è poi definito "vertice" se è il maggiore tra quelli nell'intorno del proprio livello, in quello superiore ed in quello inferiore, come abbiamo già visto in SIFT e SURF.
L'orientamento in questo algoritmo è molto diverso dai due precedenti, infatti esso si basa sull'analisi delle lunghe e corte distanze di un pattern centrato nel punto candidato.
Il descrittore viene generato molto velocemente poichè si basa sul concetto di *descrittore binario* [8].
Non avendo utilizzato questo algoritmo di estrazione non approfondiamo l'argomento più del dovuto.

## 2.2 Feature Globali

Le Feature Globali forniscono una rappresentazione dell'immagine molto più compatta rispetto alle Locali. Infatti esse non sono altro che un prodotto derivato dal

trattamento delle Feature Locali. La procedura che vedremo a breve può essere considerata come una vera e propria quantizzazione. Essa produce una rappresentazione snella richiedendo una banda di trasmissione inferiore, introducendo una perdita di dettaglio nell'immagine.
Il concetto alla base delle Feature Globali non è altro che il ben noto *"Bag-of-Words" (BoW)* utilizzato nell'elaborazione dei documenti di testo. Avendo a disposizione un vocabolario, il documento viene rappresentato da un istogramma (*Bag of Words*), cioè un vettore con il numero di occorrenze di ogni parola del vocabolario. Avendo a disposizione vocabolari molto vasti, si ottengono solitamente istogrammi molto radi e con poche ripetizioni di parole, quindi non è raro convertire l'istogramma ad una forma binaria. Dopo che il documento è stato ridotto a istogramma può essere confrontato ad un'altro BoW, valutando il grado di somiglianza in funzione della distanza Euclidea tra i due.

Dal punto di vista dei documenti di testo, l'associazione $BoW$ è molto intuitiva, lo è meno per le immagini, ma non per questo complicato da comprendere. Nelle immagini, al posto delle parole, si considerano le Feature Locali per la loro caratteristica robustezza alle principali deformazioni, lavorando quindi con *"Bag-of-Features"* o *"Bag-of-Visual-Words" (BoVW).*
Il vocabolario di parole visuali è un vettore di K elementi, ognuno dei quali è una parola visuale ("Visual Word") rappresentata da un descrittore P-dimensionale. Sivic e Zisserman nel 2003 [9] proposero un approccio par la generazione di vocabolari di parole visuali utilizzando delle immagini di "training". Dalle immagini di partenza si genera un vettore P-dimensionale con tante entry quante sono le *Features* totali successivamente il vettore viene quantizzato in regioni (cluster) dove i vari centri $v_k$, $k = 1, ..., K$, vengono dedotti utilizzando l'algoritmo *k-means.*
Una volta ottenuto il vocabolario si possono definire le BoVW delle varie immagini calcolando a quali parole visuali le *Features* dell'immagine sono più vicine: il calcolo della distanza tra la feature e la parola visuale estratta con SIFT viene effettuato nelle 128 dimensioni del vettore.
Secondo quanto detto, ogni *Features* dell'immagine va ad incrementare gli elementi della specifica parola visuale, costituendo l'istogramma. Ottenuti tutti gli istogrammi richiesti per la formazione del proprio database, si normalizzano tutti secondo lo schema *tf-idf* per migliorare il *matching.*
Per valutare il grado di somiglianza tra due immagini, come per i documenti

di testo, viene calcolata la distanza tra gli istogrammi, valutando il risultato in funzione di una precisa threshold.

Capitano vocabolari dove il numero di parole visuali raggiungono l'ordine del milione e ciò comporta che la generazione dei BoW diventi un'opera computazionalmente onerosa. Per alleggerire le operazioni, nel 2006 Nister e Stewenius [10], introdussero nel modello di Sivic una modifica al vocabolario, rendendolo di fatto una struttura gerarchica, navigabile tramite algoritmi ad-hoc che permetteva una costruzione degli istogrammi meno onerosa.

Nel 2006, dopo aver approfondito la tecnica classica di generazione degli istogrammi, Gemert, si accorse di due importanti debolezze (*assegnamento verosimile* e *assegnamento incerto*) ed introdusse i *Kernel Codebook (KC)*. I due problemi sono un diretto risultato di una distribuzione non omogenea delle *Feature*, introdotta dalla clusterizzazione per generare il vocabolario.
Il primo problema di *verosimiglianza* si manifesta quando la *Features* viene ricondotta a una parola visuale molto distante dal suo valore, introducendo un approssimazione molto grossolana (il triangolo nella Figura 2.6). Il secondo si presenta quando una *Features* è equamente distanziata da due parole visuali (la stella nella Figura 2.6).

Per far fronte ai problemi appena elencati, Gemert propose un assegnamento

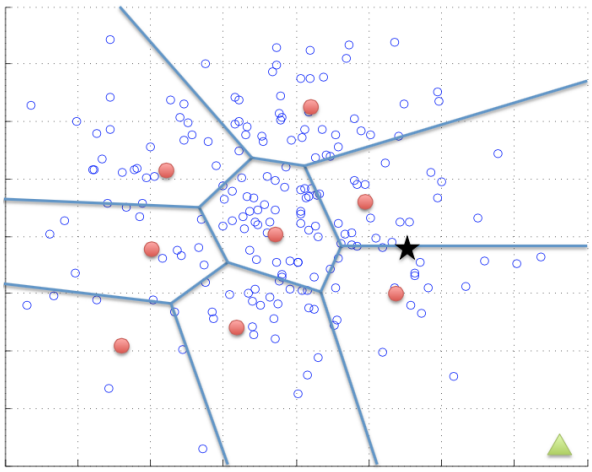


Figura 2.6: *Assegnamento verosimile* e *Assegnamento incerto*

*soft* delle feature locali. Altri dopo di lui continuarono ad approfondire possibili

soluzioni: Philbin con *Soft Coding* e Yang con *Sparse Coding*.

## 2.3 Codifica Temporale

Il lavoro di tesi propone una codifica multivista su frames contemporanei, con la consapevolezza che lo stesso tipo di codifica, ma di tipo temporale, abbia prodotto ottimi risultati.
È importante approfondire il precedente lavoro sulla codifica temporale, per acquisire le conoscenze di base che hanno permesso lo studio della codifica multivista. Per riuscire a comprendere l'argomento, cercheremo di sintetizzare, toccando i punti cardine del lavoro di Baroffio et al.[11], dove si approfondiscono i vantaggi di una comunicazione *Analisi e Compressione* (*Analise-then-Compress*, ATC) contro una *Compressione ed Analisi* (*Compress-then-Analise*, CTA), arricchendo i suoi studi con due fondamentali varianti del ATC: codifica intra-frame (INTRA), cioè senza aggregazione di *Features* e codifica inter-frame (INTER), cioè con aggregazione.

### 2.3.1 Modalità di Codifica

Andando per ordine, studiamo la metodologia INTRA, costituita in una prima codifica delle *Features*, seguita dalla trasmissione delle stesse.
Ogni frame del gruppo di immagini in analisi (*Group of Pictures*, GOP) viene analizzato indipendentemente codificandone le *Features* moltiplicate in seguito per una matrice $\mathbf{T}$. Il valore delle matrici $\mathbf{T}^{INTRA}$ e $\mathbf{T}^{INTER}$ viene stimato utilizzando la Trasformata di Karhunen-Loève (KLT), considerando tutti i descrittori estratti dal set di training. Il prodotto ha molteplici scopi, tra i quali ridurre il numero di *Features* da inviare e soprattutto diminuire il grado di correlazione tra gli elementi.
Le *Features* analizzate, sono quelle ottenute dai descrittori SIFT o SURF, che producono un flusso di bit totale calcolabile come nella formula seguente:

$$R_n = \sum_{i=1}^{M_n} (R_{n,i}^c + R_{n,i}^d) \tag{2.6}$$

Dove $M_n$ sono i *Keypoints* nel frame e $R^c_{n,i}$ sono i bits per trasmettere il vettore $p_{n,i} = [x, y, \sigma, \theta]^T$, che contiene:

- $(x, y)$ indicano la localizzazione dei *Keypoints*
- $\sigma$ indica la scala
- $\theta$ rappresenta l'orientamento

Per ultimo, $R^d_{n,i}$ rappresenta il flusso di bits per trasmettere il vettore P-dimensionale del descrittore SIFT (o SURF).
Più avanti esauriremo l'analisi della modalità INTRA confrontando i costi di coding con quelli della modalità INTER.

È importante notare che tra due frames successivi, le *Features* rimangano per buona parte immutate, modificandosi di poco. Possiamo capire meglio osservando l'immagine in Figura 2.7, dove vediamo alcune *Features* che scompaiono, per il cambiamento di inquadratura (oggetti che coprono o scoprono elementi) o perché non abbastanza definiti, e quindi vengono trascurati; altre invece che compaiono o si ripetono: $\langle d_{n-1,3}, d_{n,1}\rangle$, $\langle d_{n-1,5}, d_{n,4}\rangle$, $\langle d_{n-1,2}, d_{n,8}\rangle$ e $\langle d_{n-1,4}, d_{n,9}\rangle$.

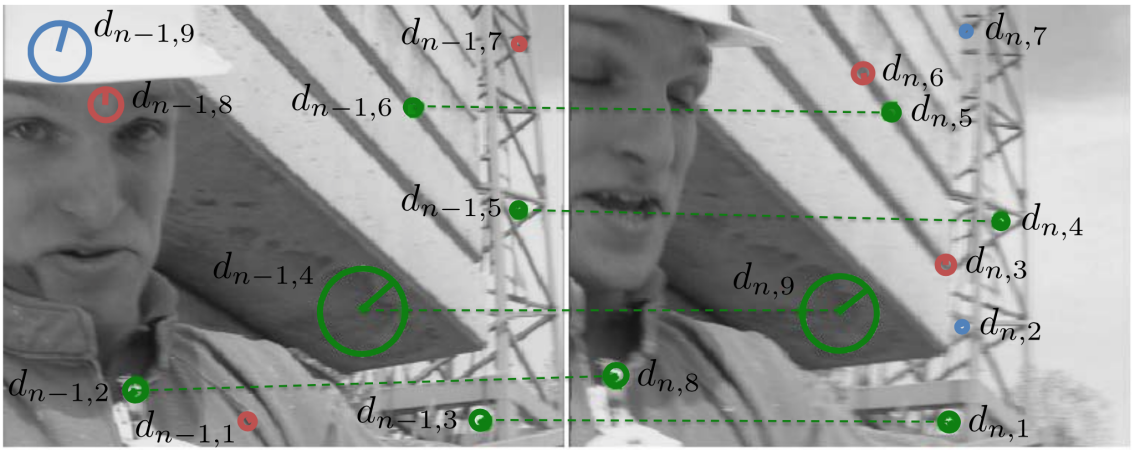



Figura 2.7: Ricerca delle stesse *Features* in frame successivi

Vediamo chiaramente lo sviluppo della codifica temporale nei blocchi della Figura 2.8. Le *Features* $\mathcal{D}_n$ codificate dal frame corrente vengono confrontate con quelle del frame precedente ($\mathcal{D}_{n-1}$). Per velocizzare il processo, le *Features* vengono confrontate solo con una piccola porzione ($\mathcal{C}$) del frame precedente, trovate

nell'intorno di $\pm\Delta\sigma$ del nuovo *Keypoint*. Ottenendo quindi i descrittori aggregati come in Formula:

$$\begin{aligned} &\tilde{d}_{n-1,l^*} = arg \min_{\tilde{d}_{n-1,l} \in \mathcal{C}} J^{INTER}(d_{n,i}, \tilde{d}_{n-1,l}) \\ &dove \\ &J^{INTER}(d_{n,i}, \tilde{d}_{n-1,l}) = \frac{1}{\sqrt{P}} \left\| \mathbf{d_{n,i}} - \tilde{\mathbf{d}}_{\mathbf{n-1,l}} \right\|_2 + \lambda R_{n,i}^{c,INTER}(l) \end{aligned} \tag{2.7}$$

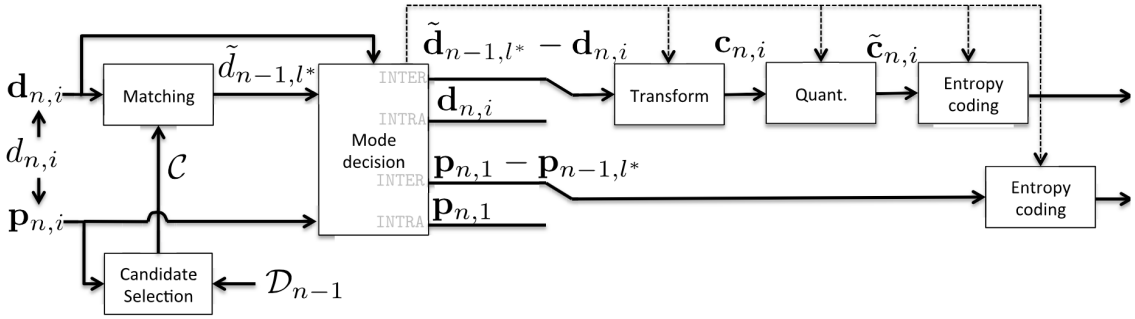



Figura 2.8: Diagramma a blocchi dell'algoritmo di estrazione delle *Features* proposto da Luca Baroffio

P è il Root Mean Square Error tra il descrittore e i nuovi candidati aggregabili, mentre $R_{n,i}^{c,INTER}$ tiene conto del rate di codifica della nuova porzione in funzione dei vecchi elementi.
Come avevamo accennato nel caso intra-frames, i descrittori vengono normalizzati per due diverse matrici di Karhunen-Loève allo scopo di decorrelare i descrittori ed incrementare la velocità di codifica:

$$\begin{aligned} \mathbf{c}_{n,i}^{INTRA} &= T^{INTRA}(d_{n,i}), \\ \mathbf{c}_{n,i}^{INTER} &= T^{INTER}(d_{n,i}, \tilde{d}_{n-1,l^*}), \end{aligned} \tag{2.8}$$

Lo scopo di Baroffio era identificare in quale momento era opportuno o meno aggregare i frames. Tale decisione viene presa osservando il minimo tra i due costi di coding:

$$J^{INTER}(d_{n,i}, \tilde{d}_{n-1,l*}) = \frac{1}{\sqrt{P}} \left\| \mathbf{d_{n,i}} - \tilde{\mathbf{d}}_{\mathbf{n,i}} \right\|_2 + \lambda(R_{n,i}^{c,INTER}(l*) + R_{n,i}^{d,INTER}(l*)), \tag{2.9a}$$

$$J^{INTRA}(d_{n,i}) = \frac{1}{\sqrt{P}} \left\| \mathbf{d_{n,i}} - \tilde{\mathbf{d}}_{\mathbf{n,i}} \right\|_2 + \lambda(R_{n,i}^{c,INTRA} + R_{n,i}^{d,INTRA}), \tag{2.9b}$$

La formula $J^{INTER}$, indicata nell'Equazione 2.9a, è leggermente differente da quella indicata nell' Equazione 2.7. Il dato aggiuntivo $R_{n,i}^{d,INTER}$, ha un impatto minimo poichè vale circa lo 0.3% rispetto a $R_{n,i}^{c,INTER}$.

Come si vede dal diagramma a blocchi 2.8, il passaggio successivo è la quantizzazione per $\Delta_j$, applicata ai descrittori trasformati dalle matrici $\mathbf{T}$:

$$\tilde{c}_{n,i,j} = \Delta_j \cdot round(c_{n,i,j}/\Delta_j) \tag{2.10}$$

Per ultimo, viene affrontato il calcolo dell'entropia, realizzato con una codifica aritmetica. La probabilità dei vari simboli utilizzati nel calcolo entropico è dedotta dal training set.
Viene definita una mappatura ($\mathcal{M}$) della *Features* $\tilde{\mathcal{D}}_n$ del frame corrente in funzione di quelle precedenti $\tilde{\mathcal{D}}_{n-1}$. Le *Features* $\tilde{\mathcal{D}}_n$ vengono poi ordinate ottenendo $\mathcal{M}'$ dove gli elementi sono ordinati dal minimo al massimo. Per ultimo viene eseguita la differenza tra le coppie del vettore $\mathcal{M}'$, ottenendo il vettore $\mathcal{O}$. I valori così ottenuti vengono immagazzinati per essere utilizzati nella codifica aritmetica.

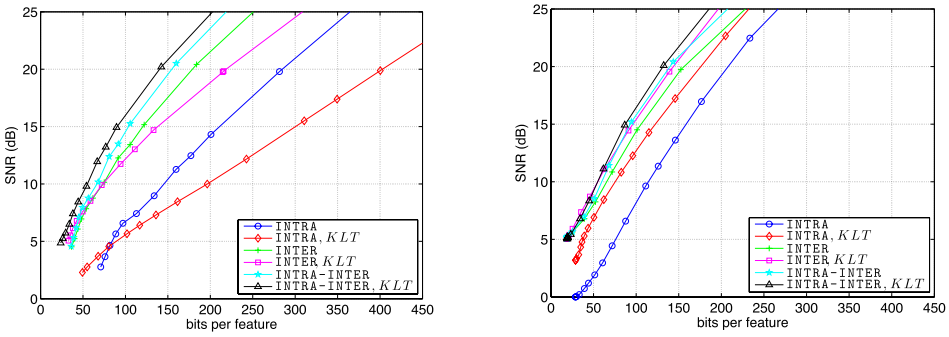
### 2.3.2 Valutazione e Risultati

La valutazione della codifica temporale è svolta utilizzando tre approcci, con metodi e metriche diverse:

- **Rate Distortion Analysis:**
  Da ogni frame $\mathcal{I}$ vengono estratte le features $\mathcal{D}$ e poi quantizzate secondo $\Delta$ ottenendo $\tilde{\mathcal{D}}$ secondo lo schema illustrato poco sopra. Viene studiato il rapporto tra bitrate di codifica richiesto per le *Features* e la distorsione introdotta dalla quantizzazione (calcolata come SNR).

  I risultati ottenuti con SIFT sono ottimi, si riescono a raggiungere livelli più alti di bit per *Features* mantenendo lo stesso grado di SNR ottenuto

con SURF. In ultima analisi, entrambi i descrittori migliorano i risultati con l'ausilio della KLT migliorano i risultati. Per completezza, possiamo vedere l'andamento del metodo nei grafici di Figura 2.9.



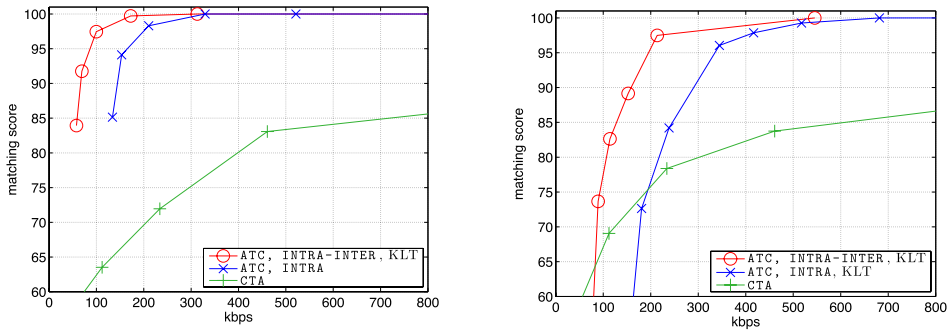

(a) (b)

Figura 2.9: Curve di distorsione del rate con codifica ATC. (a) SIFT, (b) SURF

- **Content Based Video Retrieval (CBVR):**
  Come nell'approccio precedente, i video sono analizzati frame per frame studiando sia il caso ATC che CTA, traendo dei risultati da una valutazione a punteggi.
  L'accuratezza del detector viene stimata secondo la "ripetibilità" tra *Features* originali $\mathcal{D}_n$ e quelle ricostruite $\tilde{\mathcal{D}}_n$, sia nel caso ATC che CTA.
  La ripetibilità viene stimata in funzione delle zone $R_d$. Le aree $R_d$ sono zone proporzionali alle *Features* originali $\mathcal{D}_n$, o a quelle stimate $\tilde{\mathcal{D}}_n$. Le due regioni sono similari quando l'errore di sovrapposizione è $\varepsilon < 0.5$.
  Il punteggio finale si ottiene come media di tutti i confronti su tutti i frames del video.
  Partendo dai risultati ottenuti, si valuta il descrittore, misurando il punteggio di "matching". Vengono conferiti punti ai match che, oltre ad avere $\varepsilon < 0.5$, possiedono anche una piccola distanza Euclidea. Il punteggio finale viene calcolato dal rate tra i match corretti ed il totale dei riscontri possibili, mediando su tutti i frames del video.

  Dall'analisi dei risultati si nota che, nel caso CBVR ATC, la ripetibilità ha valori pari al 100%, al contratio del caso CTA dove la compressione comporta distorsione.

Riportiamo in Figura 2.10 l'andamento del punteggio di matching per

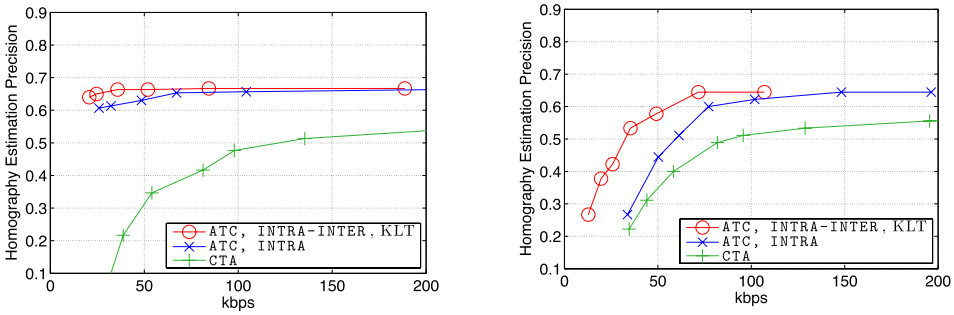



(a) (b)

Figura 2.10: Curve dei punteggi di matching per il caso CBVR. (a) SIFT, (b) SURF

il caso SIFT e SURF, dove in entrambi si osserva che i valori di ATC sono migliori di CTA. Ulteriormente, si nota l'andamento migliore del caso INTER, in particolare per la modalità SIFT.

- **Homography Estimation:**
  Vediamo l'ultima configurazione, nel caso ATC e CTA. Come visto precedentemente, dai frames consecutivi vengono codificate e quantizzate le *Features* $\tilde{\mathcal{D}}_n$ e $\tilde{\mathcal{D}}_m$, calcolando la loro "omografia" $\tilde{H}_{n,m,ATC,\Delta}$ tramite RANSAC. Per il caso CTA, le sequenze sono codificate con un fattore di qualità incrementale (Q = 5, 10,..., 45) calcolando $\tilde{H}_{n,m,CTA,Q}$ dalle *Features* codificate.
  Le performance sono calcolate in funzione delle curve di efficienza del rate, valutando la precisione di Omografia stimata: vengono estratti i quattro corner dal primo frame e tramite $\tilde{H}_{n,m}$ si possono calcolare le stime dei corner del frame successivo. I valori stimati vengono quindi paragonati con i corner reali del frame successivo. Anche in questo ultimo caso vediamo dai grafici in Figura 2.11 che la tecnica ATC è la migliore di CTA, e l'utilizzo di SIFT incrementa il loro gap. SURF, al contrario, è molto più sensibile alla quantizzazione.




(a) (b)

Figura 2.11: Curve dei affidabilità per lo stimatore di Omografia. (a) SIFT, (b) SURF

### 2.3.3 Conclusioni

Si possono quindi notare le ottime performance dell'*Analizzare e poi Comprimere*, nonostante l'estrazione necessaria delle *Features* sia onerosa per i nodi. La codifica temporale riesce a raggiungere una riduzione della ridondanza delle *Features* dell'85% con una codifica di tipo INTER rispetto a una di tipo INTRA.

# Capitolo 3

# Approcci alla Cooperazione in Reti di Camere

Nel capitolo Introduzione abbiamo delineato brevemente i sistemi di sensori con i suoi scopi, vantaggi e svantaggi. In questo capitolo andremo a collocare il nostro lavoro all'interno del panorama scientifico contemporaneo, analizzando ricerche connesse ai nostri studi.

Quello riportato in Figura 1.1 nell'Introduzione è un esempio approssimativo di *Communication Graph*, dove è indicato il collegamento mediante protocolli 802.15 tra sensori generici. Nelle VSN è comodo aggiungere un livello di astrazione superiore, indipendente dal precedente, detto *Vision Graph*. Come possiamo vedere in Figura 3.1(c), i collegamenti tra gli elementi sono tracciati in base all'area visiva condivisa dalle camere.
L'argomento è ampiamente discusso nei lavori di Dhanya Devarajan nel 2006 e 2008 [12] [13]. In maniera molto simile alla nostra procedura, le camere codi-

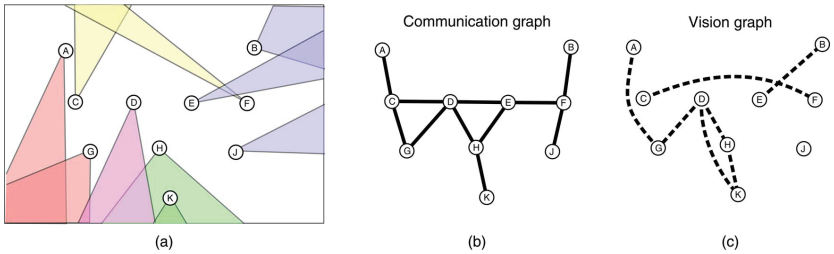



Figura 3.1: a) Campi di vista delle camere. b) Rispettivo Communication Graph. c) Rispettivo Vision Graph

ficano le *Features*, del loro campo visivo, tramite il descrittore SIFT. Dalle piú significative, e spazialmente ben distribuite, si produce un digest da inviare ai propri vicini, da cui poter valutare il livello di coincidenza tra i campi visivi.
Il matching tra le *Features* è effettuato nelle 128 dimensioni del descrittore SIFT. Quando il numero di *Features* riconosciute, tra le due viste in analisi, supera una soglia prestabilita, le camere sono unite nel *Vision Graph*.
La connessione nel *Vision Graph* non prescinde dallo stato dei due sensori nel *Communication Graph*. Nel lavoro di Devarajan il *Vision Graph* è generato senza considerare movimenti delle camere dopo la creazione dei link.

Passiamo ad esaminare il metodo di Hengel [14], dove si deduce la zona di sovrapposizione visuale tramite esclusione. Il metodo è stato studiato per la forte crescita di elementi nella rete dove le immagini da analizzare simultaneamente sono troppe per un singolo individuo.
Il metodo è l'opposto del modello proposto da Devarajan, infatti esso parte dal presupposto iniziale in cui tutte le camere osservano uno stesso oggetto. I vari campi visivi vengono suddivisi in zone piú ristrette, dette finestre, a cui viene associato uno stato di "occupato", se un oggetto copre lo sfondo o "inoccupato", altrimenti.
Sulla base di tale concetto, due finestre possono essere contemporaneamente solo occupate o inoccupate. Quando assumono valori opposti, significa che non osservano lo stesso spazio, e quindi vengono disassociate.
Reiterando il processo nel tempo, si accumulano gli stati delle finestre in vettori occupazionali. Infine, tramite semplici operazioni *xor* tra questi vettori, si è in grado di delineare le zone comuni.
La creazione dei vettori prescinde da una precisa sincronizzazione temporale ed un'accurata ed imparziale divisione in finestre, vincoli anche troppo costrittivi che nel lavoro di Hengel vengono rilassati.
Il processo di eliminazione dei falsi positivi in reti di 100 elementi di Hengel si è rilevato molto piú veloce di quelli descritti precedentemente per la generazione secondo Devarajan.

Altro argomento di interesse è riportato nella review di Kiran Maraiya [15]. Il concetto di clustering e aggregazione dei dati è un punto fondamentale del lavoro di tesi che segue.

Lo scopo dell'aggregazione è il risparmio energetico prodotto dalla riduzione dei dati ridondanti da trasmettere.
L'elemento di clustering verrá utilizzato nel nostro lavoro, poichè le camere delle prossime simulazioni sono raggruppate in cluster di due elementi.
La review analizza nel dettaglio anche altri numerosi aspetti dell'aggregazione accennando ad approcci aggregativi (tree based, multipath, hybrid), alle modalitá lossy e lossyless e ai valori di autenticitá e confidenzialitá dei dati.

Rimanendo nell'ambito dell'aggregazione, nel lavoro di Mario Christoudias [16], si cerca di risolvere un limite dell'aggregazione. Come vedremo piú avanti, l'aggregazione ha svariati vantaggi, ma paradossalmente per raggiungere un buon livello d'efficienza occorrerebbe una completa conoscenza dei frames di tutte le camere.
Viene proposto un algoritmo per correggere questo problema utilizzando un Processo Gaussiano (GP nella Figura 3.2) e un apprendimento preliminare di un modello statistico, con una sequenza di *training*.
Ogni camera è provvista di un encoder, come possiamo vedere nella Figura 3.2,

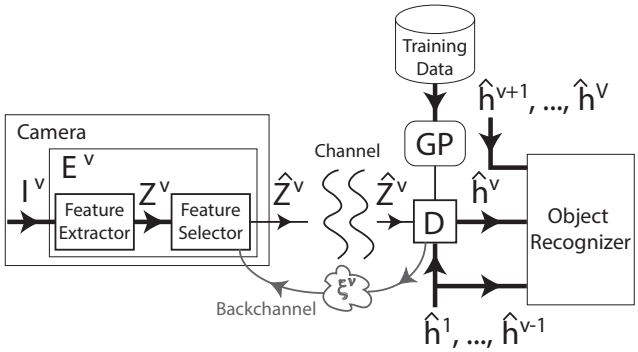



Figura 3.2: Diagramma comunicazione camera-decoder per lo scambio-richiesta delle *Features*

da cui vengono codificate tutte le *Features* ($Z^v$) ed inviate al blocco decisionale. Il blocco *Feature Selector* è comandato direttamente dai feedback del decoder ($\xi^v$), in modo da evitare l'invio di *Features* ripetute. In uscita, le camere avranno solo alcune delle $Z^v$ *Features* originali indicate come $\hat{Z}^v$.
Il ruolo del decoder (D) è di fondamentale importanza in quanto in base al proprio bagaglio informativo, comunica alle camere le *Features* mancanti tramite il backchannel. Il bagaglio informativo si genera accumulando tutti i flussi delle

camere, organizzati in istogrammi ($\hat{h}_v$).
Tramite richieste periodiche il Decoder interroga le camere ed indica le *Features* uniche che si possono inviare.

La modalitá con cui il decoder analizza le *Features* delle camere è in linea con il nostro modello di confronto basato sugli istogrammi, o Bag of Visual Words. Essi rappresentano un modo per riassumere in maniera efficiente le *Features*.
Un altro metodo è il Fischer Kernel [17], che riassume campioni di dimensioni variabili in un vettore definito. Il metodo sfrutta la relazione dei campioni di partenza con un modello parametrico stimato dal training set.
Nel lavoro di Hervè Jègou [18] se ne introduce anche un altro: *Vector of locally aggregated descriptors* (VLAD). I VLAD rappresentano una semplificazione del Fisher Kernel unendolo alla tecnica dei BoVW. Il risultato consiste in codebook molto efficaci anche con dizionari dalle dimensione ridotte (dalle 16 alle 256 parole).
Al contrario dei BoVW, nei VLAD si salva la distanza rispetto alla parola visuale di riferimento, arricchendo il carico informativo di un normale istogramma.

Ultimo lavoro fondamentale per la tesi è quello di Stefania Colonnese e altri [19], che abbiamo utilizzato come punto di partenza delle nostre analisi.
Il lavoro di Colonnese affronta passo passo l'aggregazione spaziale, utilizzando video campioni senza sfruttare i Descrittori Locali. Il primo passo del suo lavoro è la valutazione delle zone di sovrapposizione tra le camere indicata come *Common Sensed Area* (CSA). L'identificazione della CSA viene affrontata suddividendo i punti dell'immagine ($x_i$, $y_i$) (i punti sono ottenuti come proiezione di punti reali 3D su un piano i-esimo di riferimento) in elementi mobili ($M_i$) ed elementi di Background ($B_i$). Come seconda analisi, date due immagini (i, j), si valutano i punti mobili e di background comuni, ottenendo $CB_{ij}$ e $CM_{ij}$.
La CSA viene valutata come rapporto tra i punti comuni e tutti i punti dell'immagine di riferimento:

$$\alpha(i,j) = \frac{|CB_{ij} \cup CM_{ij}|}{|B_i \cup M_i|} \tag{3.1}$$

Nel corso dell'articolo, $\alpha(i,j)$ viene sostituito con uno stimatore ($\tilde{\alpha}(i,j)$) poichè è ricavabile in tempo reale e non occorre differenziare gli oggetti mobili dal background. Noi utilizzeremo, invece, le distanze tra gli istogrammi che rappresentano

le *Features* dei frames.
Lo scopo di Colonnese è quello di legare il grado di sovrapposizione ($\alpha(i,j)$) al guadagno d'aggregazione ($\eta(i,j)$).
Per calcolare $\eta$, si considerano due camere, i e j, da cui si valutano il bitrate totale generato dalle due in maniera indipendente ($r^{AVC}(i)$ e $r^{AVC}(j)$) e quello ottenuto con l'aggregazione ($r^{MVC}(i,j)$):

$$\eta(i,j) = 1 - \frac{r^{MVC}(i,j)}{r^{AVC}(i) + r^{AVC}(j)} \tag{3.2}$$

Per valutare il grado di correlazione tra $\eta(i,j)$ e $\alpha(i,j)$ sono stati sfruttati gli stessi database usati da noi nel seguito (*Akko & Kayo, Kendo e Balloons*).
Il legame tra $\eta$ e $\alpha$ è realizzato in modo empirico, tramite un modello iperbolico che riportiamo di seguito:

$$\eta(\alpha) = \eta_{max} \cdot \frac{\epsilon}{1 - \alpha + \epsilon} \tag{3.3}$$

$\eta_{max}$ è stimato dai flussi AVC, mentre $\epsilon$ varia tra $0.015 \div 0.08$.
Dopo aver trovato il modello empirico, Colonnese, ha simulato delle reti di sensori, con lo scopo di valutare i benefici dell'aggregazione stimato dal valore di CSA. I modelli utilizzati sono due: caso Single Hop dal Sink e caso Multi Hop.

# Capitolo 4

# Modello di Previsione del Guadagno di Codifica

L'obiettivo del Capitolo 4 è motivare la ricerca di un modello in grado di legare il guadagno al grado di somiglianza raggiunto tra due frames.
Nel Capitolo 5, applicheremo il modello di previsione ad una rete simulata, in modo da testare l'andamento energetico del bottleneck e dei singoli cluster.

## 4.1 Composizione Vocabolari

Il primo passo è stata la creazione del dizionario come descritto nel sottocapitolo 2.2 della sezione Background, partendo dal dataset *Akko & Kayo* dei *Fuji Laboratory Nagoya University*.

Nella Figura 4.1 possiamo osservare la disposizione delle camere per il dataset in analisi. Ogni camera del dataset è un sensore camera in grado di acquisire le immagini in bassa risoluzione, ad una velocità di 30 frame al secondo. In orizzontale, le camere sono disposte in fila ogni 5 cm e, in verticale, ogni 20 cm per un totale di 5 righe con 20 camere l'una, ottenendo un sistema di 100 elementi. Abbiamo preso un frame ogni 15 dalle camere 0, 5, 8, 9, 10, 30, 50, 90 (sapendo che i video in analisi sono da 30 frames al secondo, abbiamo accumulato 2 fotogrammi al secondo, per un totale di 21 fotogrammi a video).

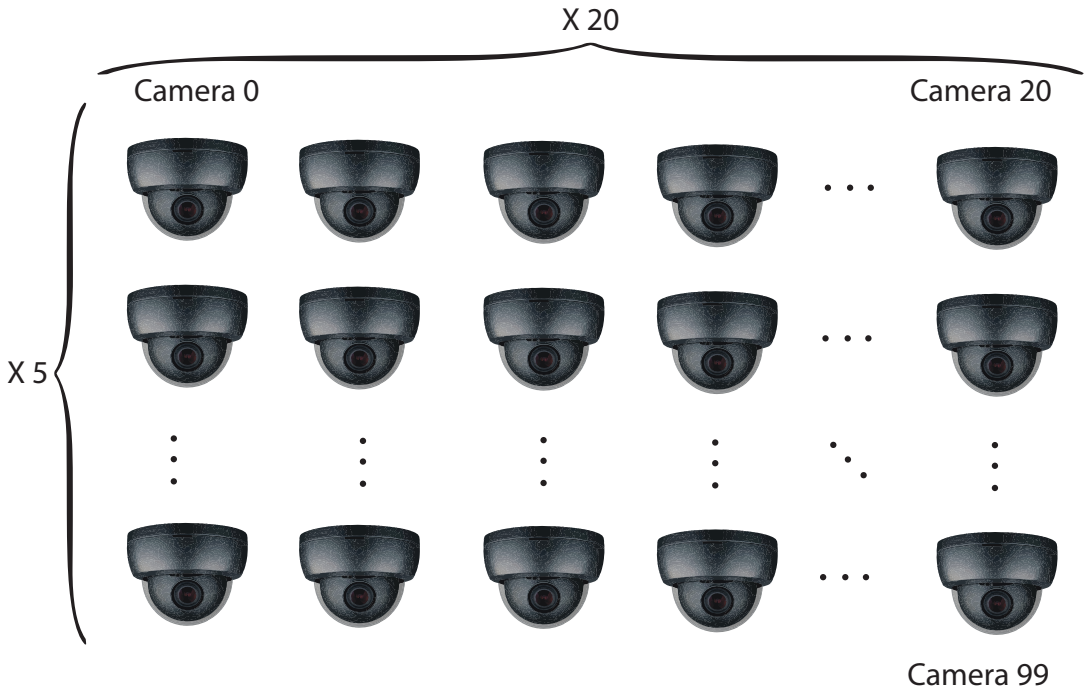



Figura 4.1: Disposizione Camere Dataset Akko&Kayo

Da ogni vista, vengono analizzati i frames contemporanei, codificando le rispettive features $\mathcal{D}_n$. Tutti gli elementi accumulati dalle analisi dei fotogrammi ci hanno permesso di creare tre dizionari dalle dimensioni incrementali: 1024, 4096 e 16384 parole visuali. L'utilizzo di dizionari sempre più vasti è facilmente intuibile: avendo a disposizione un numero maggiore di parole avremmo potuto ottenere istogrammi (BoVW) sempre più accurati, e quindi risultati migliori.
L'istogramma non è altro che la rappresentazione dell'immagini in termini di parole del dizionario, indicandone la quantità di parole visuali trovate nel fotogramma in analisi.

## 4.2 Analisi Immagini ed Estrazione Dati

Successivamente alla creazione dei dizionari, abbiamo confrontato ogni fotogramma con tutti i restanti delle altre camere nel medesimo attimo temporale. Per ogni confronto abbiamo seguito le seguenti fasi:

- **Calcolo Bitrate**:

Tramite lo script di Luca Baroffio (che abbiamo approfondito nel capitolo di Background), ricaviamo il bitrate prodotto dall'invio di due set di features *128x*$N_2$ non aggregate (codifica INTRA) e da un flusso di descrittori *128x*$N_1$ risultante dall'aggregazione di due set INTRA (codifica INTER). Dai due bitrate calcoleremo il guadagno raggiunto.

- **Produzione BoVW**:
  Produzione degli istogrammi (*In*) che rappresentano i frame in termini di *Bag of Visual Words* (BoVW). I frames che hanno istogrammi simili, e cioè che hanno una distanza piccola, sono immagini non molto diverse, quindi producono un bitstream aggregato migliore come output dello script di Baroffio (cioè minore della somma di due codifiche INTRA).
  Avendo dizionari molto vasti, ci si è accorti che i valori delle varie parole (le altezze dei vari bin degli istogrammi) erano tutti di valore molto modesto, approssimabile a 1 o 0. Sapendo che, successivamente, avremmo dovuto scambiare gli istogrammi per aggregarli, abbiamo intuito che più l'istogramma fosse stato piccolo più avremmo risparmiato energia.

- **Calcolo delle distanze tra i BoVW:**

  a) **Euclidea**: calcolata come la somma del quadrato delle differenze tra i valori delle varie parole del dizionario per i due frames, il tutto sotto radice quadrata:

  $$dE_n = \sqrt{\sum_{i=1}^{n}(I1_{\text{Wordi}} - I2^2_{\text{Wordi}}} \qquad \text{con n = 1024, 4096, 16384} \quad (4.1)$$

  detta anche di Hamming nel caso binario

  b) **Pesata**: al momento della creazione dei vocabolari viene anche generata una matrice *Adiff* dai frames di partenza, dove vengono calcolate le distanze euclidee tra ogni parola del dizionario creato. Questo permette di far pesare maggiormente nel calcolo della distanza alcune parole rispetto ad altre. Il valore della *Adiff* viene poi moltiplicato

per il valore della differenza tra gli istogrammi ($In$).

$$\begin{aligned} \Delta I &= I1 - I2 \\ dP &= \sqrt{\Delta I * Adiff * \Delta I^T} \end{aligned} \tag{4.2}$$

c) **L1**: calcolata come somma dei moduli delle differenze tra i valori delle varie parole del dizionario per i due frames

$$dL1_n = \sum_{i=1}^{n} |I1_{\text{Word}i} - I2_{\text{Word}i}| \qquad \text{con n = 1024, 4096, 16384} \tag{4.3}$$

d) **Kolmogorov Smirnov**: calcolato come il valore massimo tra tutte le distanze degli istogrammi

$$dKS_n = \max_{i=1}^{n} |I1_{\text{Word}i} - I2_{\text{Word}i}| \qquad \text{con n = 1024, 4096, 16384} \tag{4.4}$$

Questo processo viene ripetuto per tutti e tre i dizionari e suddividendo i frame in più porzioni. Ogni porzione prende il nome di settore, ottenuto suddividendo il frame in zone rettangolari di uguale dimensione (come possiamo vedere nell'esempio con 4 settori in Figura 4.2). Abbiamo analizzato i frame con 1 settore (nessuna suddivisione), 4 settori e 16 settori.

Studiare porzioni di immagine più piccole permette di avere istogrammi sempre più precisi, ottenendo una funzione di andamento il più fedele possibile alla realtà.

Nelle fasi preliminari non ci preoccupavamo ancora dell'impatto a livello di bitrate che avrebbe avuto una codifica con molte parole e molti settori. In seguito, vedremo come abbiamo definito la configurazione migliore per i nostri scopi.

## 4.3 Scelta Miglior Configurazione

Come accennato poco fa, avere più dizionari, più settori e più modalità di differenza produce numerosi dati, discriminati da elementi di partenza. Questi elementi di partenza verranno identificati da qui in avanti come configurazioni. Per chiarezza elenchiamo gli elementi che compongono le varie configurazioni:

- Dimensione del dizionario (1024, 4096 e 16384)

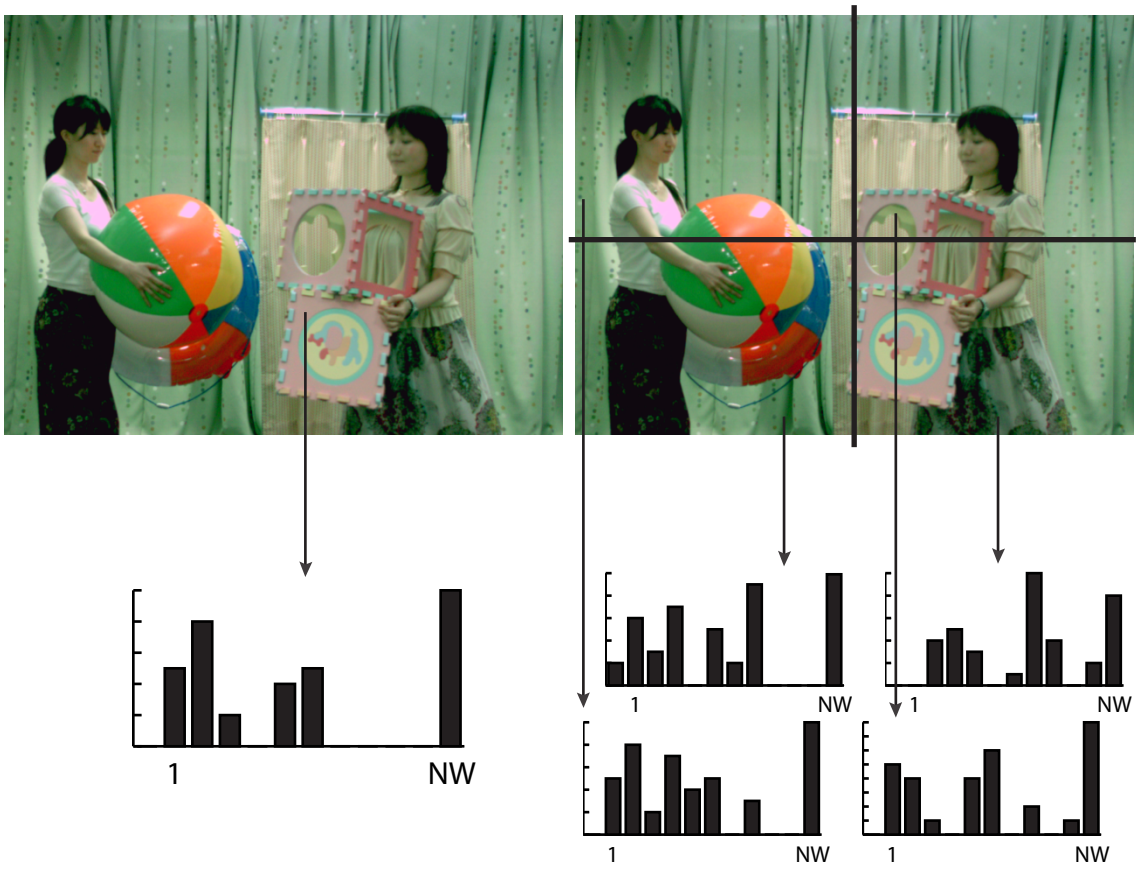



Figura 4.2: Suddivisione frame in settori (NW = Numero parole del dizionario)

- Numero di settori (1, 4 e 16)
- Tipologia di distanza (Euclidea, Pesata, L1 e Kolmogorov Smirnov)
- 8bit o binario

In totale avremo 36 configurazioni con istogrammi da 8bit e altrettante con istogrammi binari. Quindi è stato necessario ricercare la migliore delle 72 calcolando il valore di correlazione tra le distanze degli istogrammi della configurazione in analisi ed il loro $\eta(i,j)$.

$$\eta(i,j) = 1 - \frac{R_c^{INTRA} + R_{c,b}^{INTER}}{R_c^{INTRA} + R_b^{INTRA}} \tag{4.5}$$

Il valore di $\eta(i,j)$ è ottenuto come somma del valore di $R_c^{INTRA}$, cioè il valore in bitrate per comunicare la matrice della camera di confronto senza aggregare

nulla, sommato con $R_{c,b}^{INTER}$, il valore della matrice SIFT aggregata della camera che si occuperà della comunicazione (Camera Riferimento) al nodo successivo. A denominatore, invece, avremo i valori di una comunicazione senza aggregazioni, quindi entrambe le camere comunicano interamente la propria matrice SIFT senza pensare all'aggregazione.
Il valore di correlazione ($\rho$) è calcolato tramite la funzione *corrcoef*, che valuta il livello di correlazione tra due variabili casuali $\eta$ e distanza tra BoVW ($d$):

$$\rho(\eta, d) = \frac{cov(\eta, d)}{\sigma_\eta \sigma_d} \tag{4.6}$$

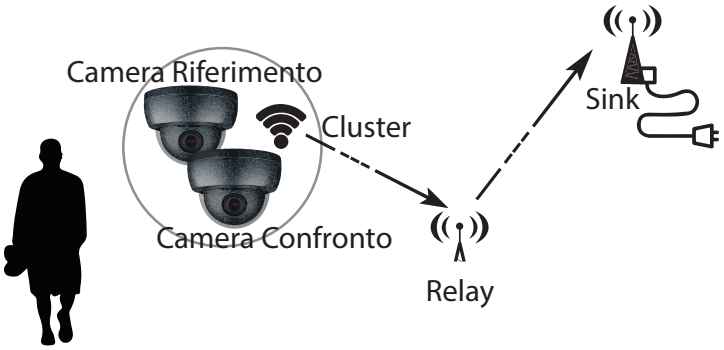



Figura 4.3: Esempio comunicazione Cluster camere - Relay - Sink

La configurazione che volevamo utilizzare doveva essere un giusto compromesso tra un'alta correlazione ed un basso bitrate totale. Vediamo di seguito gli istogrammi (Figura 4.4) delle migliori configurazioni, per ogni dataset. Per concedere una migliore lettura si è deciso di visualizzare solo i primi 10 miglior risultati. In rosso vediamo le configurazioni con istogrammi binari ed in blu quelli a 8bit. Si nota che non esiste una grande differenza in termini di correlazione ma, tra le due, le configurazioni binarie sono molto più leggere (come si potrà notare col grafico di Figura 4.5).
Nei primi posti troviamo configurazioni con bitrate molto alti, e, per le nostre analisi abbiamo deciso di scegliere la configurazione N. $15_b$ (configurazione binaria con vocabolario da 4096 e un settore con differenza L1, vedere Tabella 4.1) motivati dal fatto che possiede in tutti e tre i dataset un buon valore di correlazione ed al tempo stesso, non ha un bitrate troppo oneroso a livello energetico.

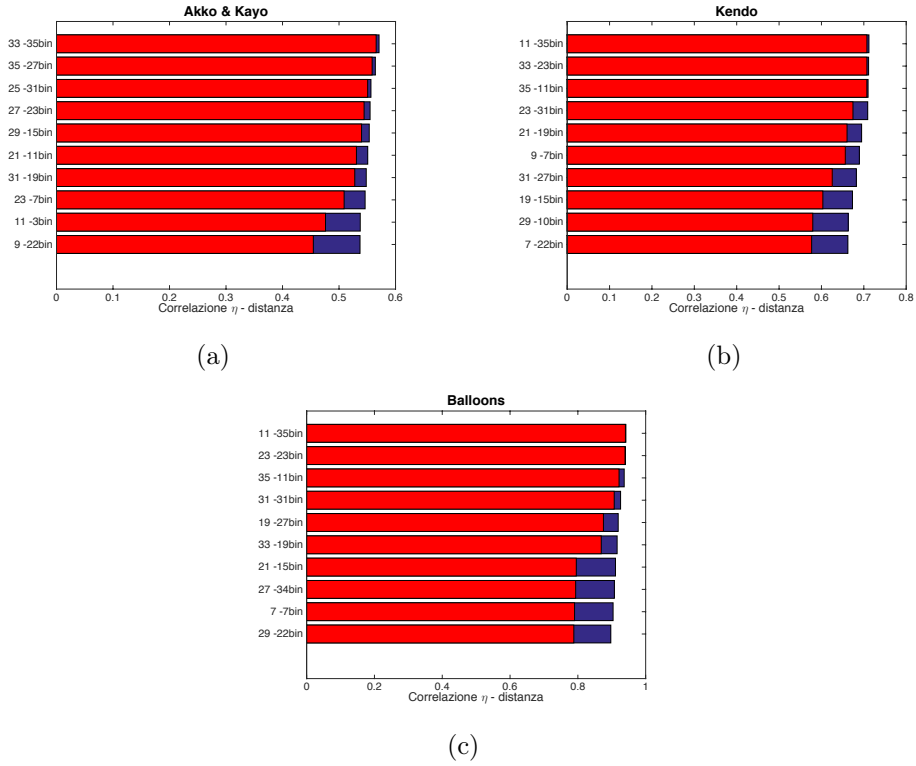



Figura 4.4: Confronto correlazione $\eta$ - distanze tra le configurazione a 8bit (N) e quelle binarie (Nbin)

| N | Parole Vocabolario | Numero Settori | Tipo di Differenza |
|---|---|---|---|
| 1, $1_b$ | 1024 | 1 | Euclidea |
| 2, $2_b$ | 1024 | 1 | Pesata |
| 3, $3_b$ | 1024 | 1 | L1 |
| 4, $4_b$ | 1024 | 1 | Kolmogorov Smirnov |
| 5, $5_b$ | 1024 | 4 | Euclidea |
| 6, $6_b$ | 1024 | 4 | Pesata |
| 7, $7_b$ | 1024 | 4 | L1 |
| 8, $8_b$ | 1024 | 4 | Kolmogorov Smirnov |
| 9, $9_b$ | 1024 | 16 | Euclidea |
| 10, $10_b$ | 1024 | 16 | Pesata |
| 11, $11_b$ | 1024 | 16 | L1 |
| 12, $12_b$ | 1024 | 16 | Kolmogorov Smirnov |
| 13, $13_b$ | 4096 | 1 | Euclidea |
| 14, $14_b$ | 4096 | 1 | Pesata |
| 15, $15_b$ | 4096 | 1 | L1 |
| 16, $16_b$ | 4096 | 1 | Kolmogorov Smirnov |
| 17, $17_b$ | 4096 | 4 | Euclidea |
| 18, $18_b$ | 4096 | 4 | Pesata |
| 19, $19_b$ | 4096 | 4 | L1 |
| 20, $20_b$ | 4096 | 4 | Kolmogorov Smirnov |
| 21, $21_b$ | 4096 | 16 | Euclidea |
| 22, $22_b$ | 4096 | 16 | Pesata |
| 23, $23_b$ | 4096 | 16 | L1 |
| 24, $24_b$ | 4096 | 16 | Kolmogorov Smirnov |
| 25, $25_b$ | 16384 | 1 | Euclidea |
| 26, $26_b$ | 16384 | 1 | Pesata |
| 27, $27_b$ | 16384 | 1 | L1 |
| 28, $28_b$ | 16384 | 1 | Kolmogorov Smirnov |
| 29, $29_b$ | 16384 | 4 | Euclidea |
| 30, $30_b$ | 16384 | 4 | Pesata |
| 31, $31_b$ | 16384 | 4 | L1 |
| 32, $32_b$ | 16384 | 4 | Kolmogorov Smirnov |
| 33, $33_b$ | 16384 | 16 | Euclidea |
| 34, $34_b$ | 16384 | 16 | Pesata |
| 35, $35_b$ | 16384 | 16 | L1 |
| 36, $36_b$ | 16384 | 16 | Kolmogorov Smirnov |

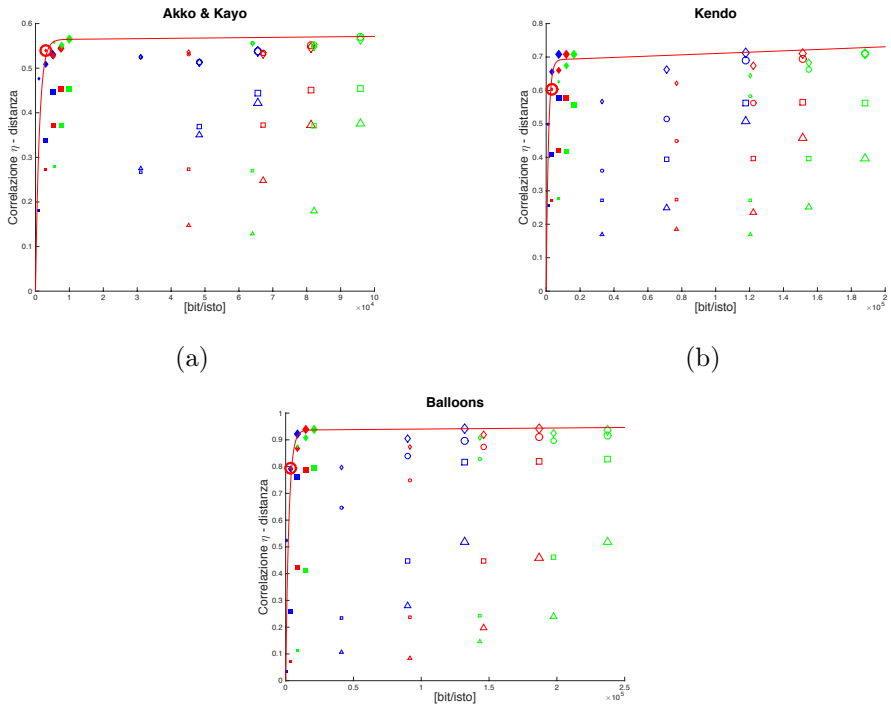
Tabella 4.1: Associazione Numero - Configurazione. Le configurazione elencate si riferiscono sia agli istogrammi a 8bit sia a quelli binari ($b$)

Per comprendere meglio la nostra scelta, aggiungiamo un gruppo di altri tre grafici (sempre divisi per dataset) (Figura 4.5) dove indichiamo con un cerchietto rosso la configurazione N. $15_b$.
I grafici tracciano l'andamento delle 72 configurazioni (36 binarie e 36 a 8bit) in relazione al proprio bitrate totale ed il loro valore di correlazione. Riportiamo, in seguito, le Formule 4.7 per il calcolo del bitrate ($R$), sfruttando il valore di entropia del dizionario ($H$):

$$\begin{aligned} R_{8bit} &= H(voc, sett) * NW * NS * 8 \qquad \text{[bit/isto]} \\ R_{bin} &= H_{bin}(voc, sett) * NW * NS \qquad \text{[bit/isto]} \\ &\quad NW = \textit{1024, 4096 e 16384} \\ &\quad NS = \textit{1, 4 e 16} \end{aligned} \tag{4.7}$$

I valori dei dizionari a 8bit e di quelli binari sono funzione dell'entropia, del numero di parole visuali del vocabolario ($NW$) e del numero di settori ($NS$) della configurazione in uso. Osservando i grafici si nota che il valore della correlazione ($\eta$) è simile per gli istogrammi binari e quelli a 8bit. Tuttavia, il bitrate di quelli

binari è notevolmente più basso, ciò provoca un consumo energetico molto inferiore.
All'interno dei grafici avremo gruppi ben distinti di simboli che andiamo a riassumere nella tabella 4.2.
Nella scelta, ci siamo concentrati sulla parte di costellazione più aderente possibile all'inviluppo sinistro segnato in rosso, che, come abbiamo più volte ripetuto, sono quelle con un buon livello di correlazione ed al tempo stesso un bitrate contenuto.




(a) (b)

(c)

Figura 4.5: Dispersione delle configurazioni in funzione di bitrate e correlazione

## 4.4 Finalizzazione Modello di Previsione

Ora che abbiamo approfondito la scelta della configurazione numero $15_b$, possiamo passare alla realizzazione del nostro modello di previsione. Lo scopo è ricercare una funzione in grado di legare in maniera univoca e più precisa possibile l'andamento della distanza tra gli istogrammi dei frames ed il guadagno atteso. Le camere, tramite il modello, sono in grado di verificare la convenienza dell'aggregazione sfruttando solo i BoVW e non i completi flussi di descrittori. L'utilizzo

| | |
|---|---|
| ○ | Differenza Euclidea |
| □ | Differenza Pesata |
| ◇ | Differenza L1 |
| △ | Differenza Kolmogorov Smirnov |
| ♦ | Istogrammi binari |
| ◇ | Istogrammi 8bit |
| ♦ | Dimensione dizionario = 1024 |
| ♦ | Dimensione dizionario = 4096 |
| ♦ | Dimensione dizionario = 16384 |
| ♦ ♦ ♦ | Proporzionali al numero di settori per frame (16, 4, 1) |

Tabella 4.2: Legenda

degli istogrammi, grazie alle loro ridotte dimensioni, permette di prolungare la vita del sistema.

Per finalizzare i nostri scopi, calcoliamo i valori di $\eta$ e distanza confrontando un frame con tutti gli altri, nello stesso attimo temporale. Ripetendo l'operazione per i tre dataset, in ogni istante successivo, e rappresentando i valori calcolati, otteniamo il grafico di Figura 4.6. Sull'asse delle $y$ abbiamo $\eta(i, j)$ mentre sull'asse delle $x$ le distanze normalizzate secondo la massima distanza riscontrata in tutti i dataset. Il risultato ottenuto ha un andamento molto dispersivo, se osservato in un contesto di distanza tra la minima (circa 0.7) e la massima ottenuta (circa 0.9), ma risulta molto più semplice identificarlo se osserviamo la Figura 4.6.
I valori tracciati indicano che le immagini sono tanto più simili quanto più le distanza tendono al limite 1, e sempre più differenti avvicinandosi a 0. Vediamo di seguito il calcolo delle distanze ($d$) normalizzate ($d_{Norm}$) sull'asse dell'ascissa del grafico in Figura 4.6:

$$d_{Norm} = 1 - \frac{d}{NW} \tag{4.8}$$

Anche se è facilmente identificabile l'andamento nel grafico scalato, prima di definirlo lineare abbiamo messo a confronto diverse funzioni di fitting andando a confrontare i valori rSquared restituiti da Matlab.
Il valore della funzione di fitting è linearmente collegato al valore di correlazione. Per il caso lineare (cioè il nostro prescelto) il valore è stato tra i più alti delle

realizzazioni, sia nel caso cumulato di Figura 4.6 che in ogni singolo dataset. L'andamento lineare viene riportato nell'equazione 4.9:

$$\begin{aligned} &\text{p} = 1.477 \\ &\text{q} = \text{-}1.089 \\ &\eta(d_{Norm}) = \text{p} * d_{Norm} + \text{q} \end{aligned} \tag{4.9}$$

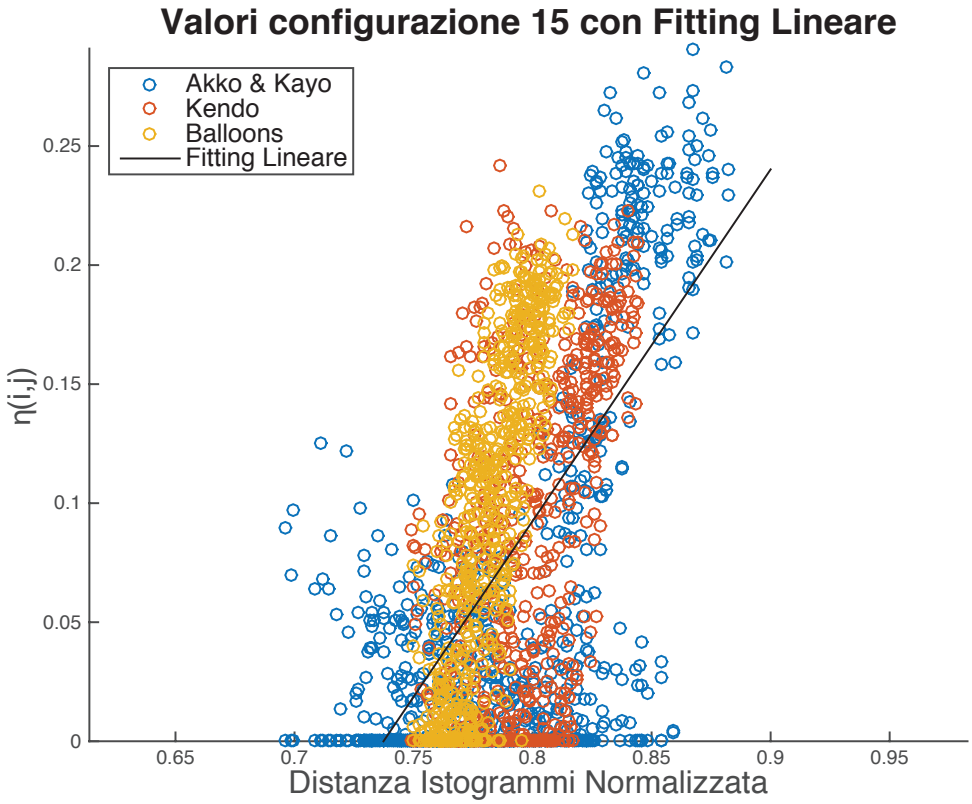



Figura 4.6: Grafico dispersione $\eta(i, j)$ - distanze tra istogrammi

Altro dettaglio che traspare dalla Figura 4.6 è il netto raggruppamento dei risultati suddivisi per dataset, la motivazione dell'andamento è dovuto all'origine dei dizionari che sono stati creati a partire solo dal dataset *Akko & Kayo* per una questione di praticità.

# Capitolo 5

# Valutazione Energetica su Reti Simulate

Una volta in possesso della funzione d'andamento, che lega la somiglianza dei frames al guadagno in codifica INTRA-INTER ($\eta(i, j)$) abbiamo simulato una VSN dove potessimo studiare l'andamento della modalità aggregata contro quella standard (senza aggregazione).
Andiamo quindi ad analizzare l'andamento energetico della rete, studiando i flussi di bit che, partendo dai clusters (coppie di camere), andranno a collidere al Sink. Studieremo il consumo energetico sul bottleneck, dove un risparmio energetico corrisponde ad un prolungamento della vita della rete.
Nella sezione a seguire, vedremo l'organizzazione dei dati raccolti durante l'analisi dei dataset, la creazione delle reti simulate, il calcolo energetico e le conclusioni raggiunte.

## 5.1 Organizzazione dei Dati Postanalisi

Nella sezione precedente abbiamo già descritto approfonditamente i calcoli ed i risultati raccolti per ogni frame. Dopo essere stati analizzati, i tre dataset vengono organizzati per consentire una più agevole elaborazione nelle sezioni successive.
Le strutture che riassumono i nostri dataset sono tre matrice *NxN*, dove $N$ sono le camere analizzate (N=8 per *Akko&Kayo* e N=7 sia per *Kendo* che per *Balloons*). Ogni camera viene confrontata nello stesso istante temporale, con tutte le altre, per un totale di 15 istanti di tempo per ogni coppia.

La strutta era stata realizzata prima dell'individuazione della configurazioni migliore (la numero $15_b$ come visto nel Capitolo 4), quindi la matrice conteneva 9 configurazione (3 vocabolari x 3 dim settori).

Per ogni istante temporale salviamo:

- **Gli istogrammi:**
  L'istogramma della Camera di Riferimento, cioè quella dell'indice riga, e quella della Camera di Confronto, cioè dell'indice colonna. Ulteriormente anche le loro conversioni in forma binaria.

- **Distanze:**
  Un vettore contente le quattro distanze tra gli istogrammi a 8bit.

- **INTRA bitrate:**
  Il bitrate che occorre per inviare per intero le *Features* codificate dal frame della Camera di Confronto

- **INTER bitrate:**
  Il bitrate per inviare in forma aggregata le *Features* del frame acquisito.

Per chiarezza, riportiamo nella Figura 5.1 uno schema della struttura della matrice.

## 5.2 Creazione Rete Simulata

Ora che siamo in possesso di tutti i dati in una forma più comoda, possiamo utilizzarli per analizzare il comportamento su reti simulate.
La generazione delle reti, che vediamo in Figura 5.2, sono prodotte automaticamente. Viene fornita la massima e la minima copertura, in metri, della rete desiderata, l'effettiva dimensione dell'istanza generata casualmente all'interno del range specificato, e la threshold (sempre in metri) della distanza massima raggiunta dai sensori.

Avendo i dati in input, le reti vengono create secondo due metodologie diverse. Nel caso di Figura 5.2(b), la rete viene ottimizzata con lo scopo di minimizzare le distanze di trasmissione e quindi raggiungere un'altissima qualità del segnale.

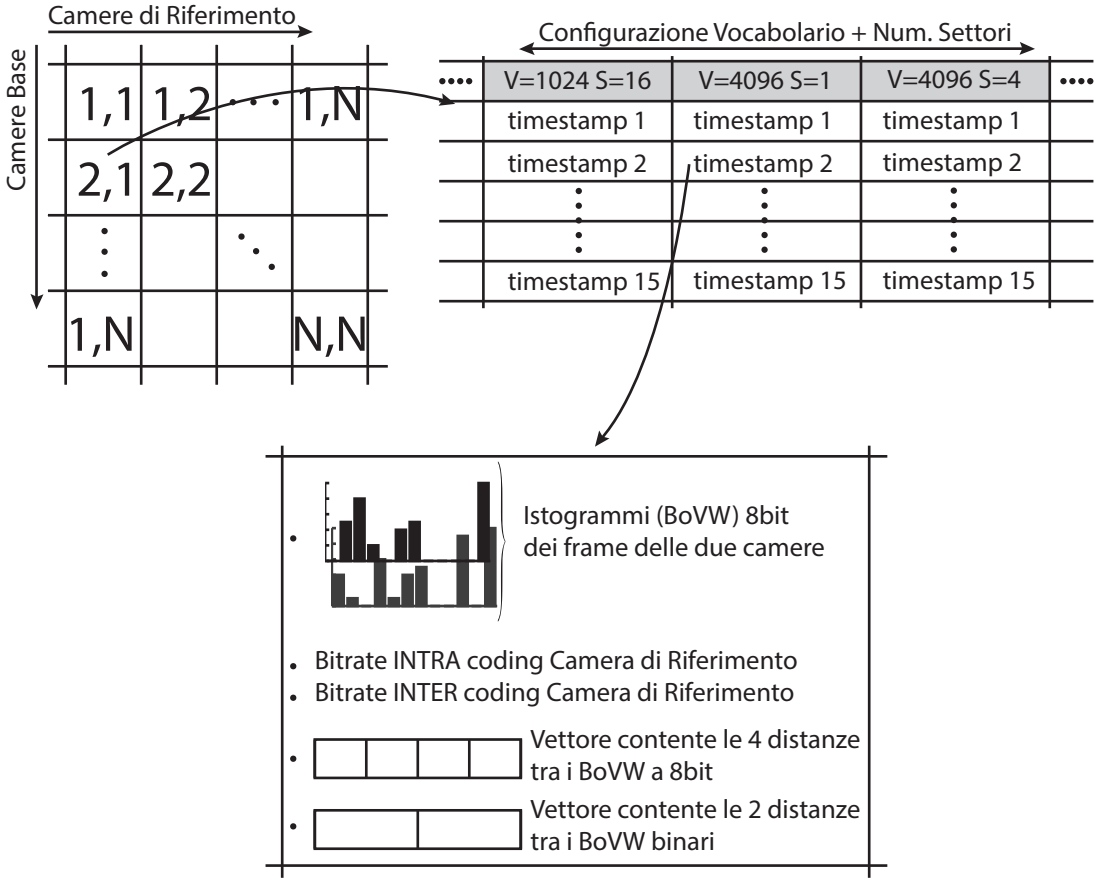


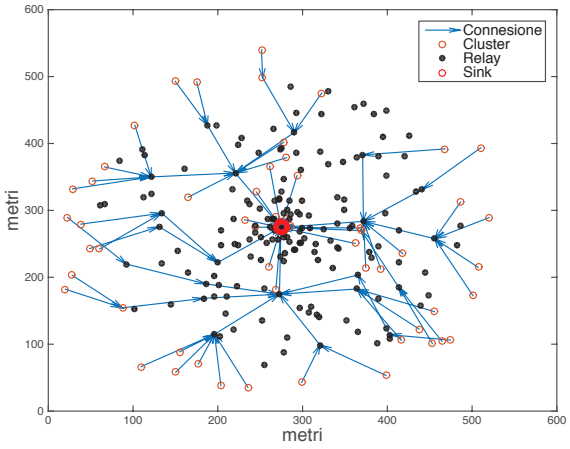

Figura 5.1: Struttura della matrice riassuntiva *NxN*

Con la Shortest Path si ottiene un alto numero di hop per raggiungere il Sink. Nel caso 5.2(a) si è preferito minimizzare il numero di hop, rendendo la rete molto simile alla realtà.
Nelle fasi di analisi successive, useremo come caso studio le reti di tipo *a*.

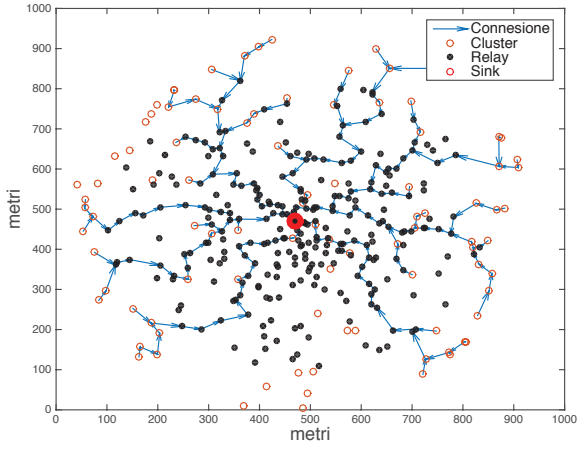
Le reti che vengono create sono composte:

- **Nodi Relay:**
  Sono sensori adibiti alla sola replicazione del segnale in ingresso, sono praticamente dei ripetitori.

- **Nodi Cluster:**
  Indicati in giallo, questi cluster sono composti da coppie di videocamere




(a)




(b)

Figura 5.2: Esempio reti simulate: (a)Hop Minimi (b)Shortest Path

wireless con un certo valore di distanza tra i loro BoVW. A seguire vedremo come avviene l'assegnazione delle distanze.

- **Nodo Sink:**
  Radice della rete dove collimano tutti i collegamenti come già indicato, è l'unico nodo ad essere connesso alla rete elettrica.

Durante le successive fasi analizzeremo il comportamento energetico dei vari cluster ed individueremo il bottleneck della rete. Il bottleneck non è altro che uno dei nodi più vicini al Sink, che ha una spesa di trasmissione più alta di tutti gli altri. Rimanendo comunque alimentato a batteria, una volta che questa si esaurisce il nodo si spegne e la rete (o una parte consistente di essa) muore. Alleggerire il carico di lavoro del bottleneck ha l'effetto di aumentarne la sua vita e di conseguenza quella della rete.

Prima di iniziare l'analisi energetica, la rete simulata prodotta viene arricchita dai valori delle matrici riassuntive. Viene assegnato il bitstream medio in codifica INTRA a tutte le camere, calcolato come media di tutti quelli contenuti nelle tre matrici *NxN*.
Vengono assegnate anche le distanze in modo casuale, non secondo una classica distribuzione omogenea, ma tenendo presente la distribuzione di tutte le distanze contenute nelle matrici precedenti. Il risultato che possiamo osservare in Figura 5.3 permette di replicare un andamento molto realistico.

Alla fine dell'assegazione, la rete per la simulazione è pronta, ogni cluster è in possesso di:

- Distanza tra BoVW
- Funzione di fitting che lega distanza al valore di codifica INTER
- Bitstream in codifica INTRA

## 5.3 Calcolo Energetico

Per capire meglio l'impatto energetico delle trasmissioni, dobbiamo fare una digressione sul lavoro pubblicato da Heinzelman nel 2000 [20], dove si spiega come

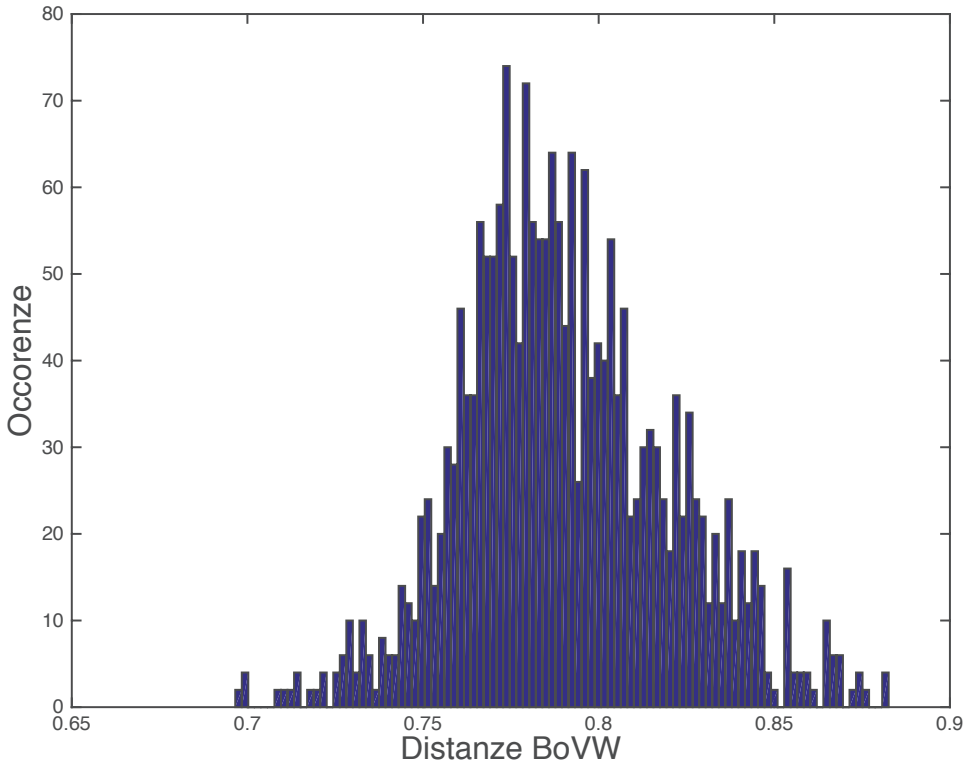



Figura 5.3: Distribuzione delle distanze tra istogrammi

poter legare il valore energetico speso da un sensore ai dati ($k$ nelle formule a seguire) inviati e ricevuti da esso ad una certa distanza ($d$). Vediamo subito le formule utilizzate:

$$\begin{aligned} E_{Tx}(k,d) &= E_{elec} * k + \epsilon_{amp} * k * d^2 \qquad \text{[nJ]} \\ E_{Rx}(k) &= E_{elec} * k \qquad \text{[nJ]} \end{aligned} \tag{5.1}$$

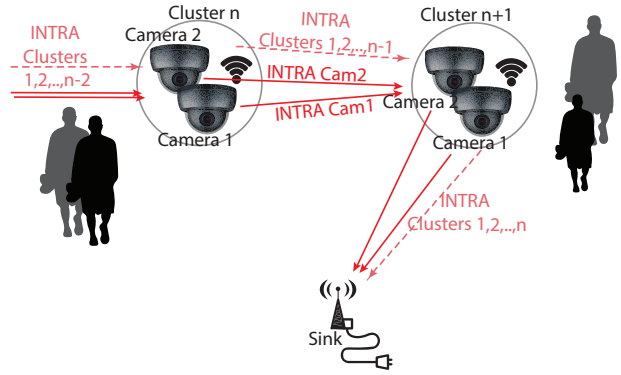
Il valore $E_{elec} = 50\ nJ/bit$ indica il valore energetico dissipato per processare un bit informativo mentre $\epsilon_{amp} = 100\ pJ/bit/m^2$ indica il valore amplificativo per raggiungere il valore minimo richiesto di SNR per tecnologie 802.15.
La simulazione sulla rete viene affrontata in tre diversi modi:

- **Modalità INTRA:**
  Tutte le camere comunicano indipendentemente il proprio bitstream non aggregato al proprio cluster di riferimento. La procedura seguita dai vari

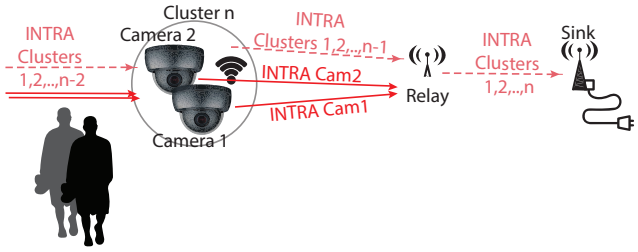
clusters è:

- ricezione dei *2n* flussi di traffico degli *n* cluster precedenti
- invio dei 2 flussi generati dal cluster corrente (tranne nel caso sia un relay)
- inoltro dei *2n* flussi ricevuti

La procedura genera un bitstream medio di 5,9 Mbit/s pari a 394 kbit/frame (3078 *Feature* per frame) calcolato in media su tutte le trasmissioni.
Per maggiore chiarezza vediamo la Figura 5.4, sia nel caso (a) che (b).




(a)




(b)

Figura 5.4: Esempio scambio bitrate in modalità INTRA: (a)caso cluster - cluster (b)caso cluster - relay

- **Modalità Cooperativo (INTER):**
  I clusters inviano un solo flusso codificato INTRA appartenente alla Camera di Riferimento ed un flusso codificato INTER dove viene aggregato

il bitstream della Camera di Confronto. Vediamo, nell'Immagine 5.5, lo scambio delle informazioni che rende possibile la modalità INTER.

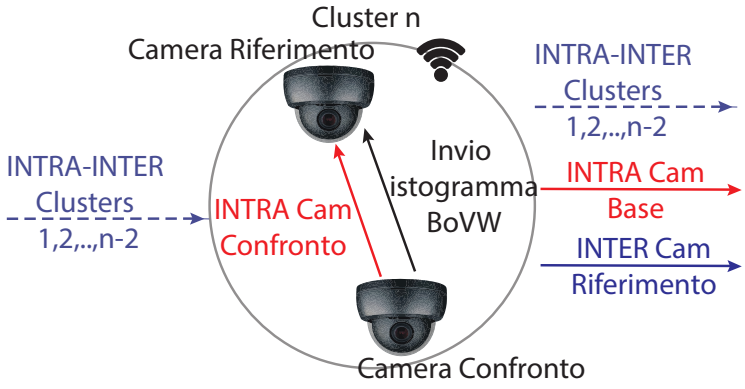



Figura 5.5: Esempio scambio bitrate interno al Cluster

In primo luogo, la Camera di Confronto invia il proprio istogramma, da cui la Camera di Riferimento ricava la distanza (distanza che nella nostra simulazione abbiamo precedentemente distribuito). La Camera di Riferimento valuta il livello di aggregazione calcolando il valore $\eta$, ed in caso di valori postitivi la Camera di Confronto invia il proprio bitstream INTRA.
Dalla distanza ed il bitstream della Camera di Confronto ($R_c^{INTRA}$), la Camera di Riferimento è in grado di ricavare il bitstream in codifica INTER ($R_{c,b}^{INTER}$) sfruttando il suo bitstream INTRA ($R_b^{INTRA}$) e la funzione di fitting vista nella sezione 4.9:

$$\begin{gathered} \eta(i,j) = 1 - \frac{R_c^{INTRA} + R_{c,b}^{INTER}}{R_c^{INTRA} + R_b^{INTRA}} \\ quindi \\ R_{c,b}^{INTER} = R_b^{INTRA} - \eta(i,j) * (R_c^{INTRA} + R_b^{INTRA}) \end{gathered} \tag{5.2}$$

Una volta ricavato $R_{c,b}^{INTER}$, la Camera di Riferimento invia i due flussi al nodo superiore aggiungendolo eventualmente con i flussi ricevuti dai suoi nodi inferiori.
Ricapitolando quanto detto, oltre all'invio finale(costituito da un flusso INTRA ed uno INTER), all'interno del cluster avremo i seguenti flussi:

– Invio BoVW per il calcolo di $\eta(i, j)$

– Invio del bitstream INTRA della Camera di Confronto (solo nel caso di $\eta(i, j)$ positivo)

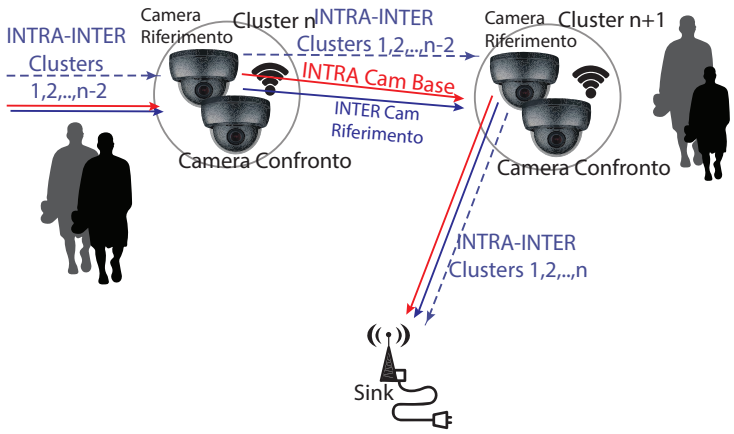



Figura 5.6: Esempio scambio bitrate in modalità INTER

Rispetto al caso INTRA, dove avevamo solo i due flussi INTRA delle due camere, il caso Cooperativo produce una spesa energetica maggiore.
Se dovessimo valutare la modalità migliore per la rete, il caso INTER produce sempre un aumento della vita del bottleneck, ma comporta anche una spesa maggiore per i singoli clusters. L'unico modo per avere sia il miglioramento della condizione del bottleneck che dei cluster, è attivare la modalità INTER solo quando la distanza dal prossimo hop permette di ammortizzare le spese interne.

- **Modalità Adattativa:**
  La modalità Cooperativa invia, a prescindere, un flusso INTRA (della Camera di Riferimento) ed un flusso della Camera di Confronto. Il secondo bitstream della Camera di Confronto è, se la distanza tra gli istogrammi lo consente, un flusso aggregato (INTER) altrimenti sarà di tipo INTRA. Nel peggiore dei casi la modalità Cooperativa si comporta come una normale INTRA con in aggiunta l'invio del BoVW.
  La modalità Adattativa rimane pressoché immutata, con la differenza che

l'aggregazione avviene solo se la distanza dal nodo superiore permette di ammortizzare gli scambi informativi interni al cluster.

$$\begin{aligned} En_{Coop} &= En_b^{INTRA} + En_c^{INTRA} + En_{b,c}^{INTER} + En_c^{isto} \\ En_{NonCoop} &= En_b^{INTRA} + En_c^{INTRA} + En_c^{isto} \\ &quindi \\ En_{Coop} &\leq En_{NonCoop} \end{aligned} \tag{5.3}$$

In quest'ultimo caso, l'invio dell'istogramma ($En_c^{isto}$ pari in media allo 0,869% del set di descrittori del frame $En_c^{INTRA}$) permette alla Camera di Riferimento di richiedere il set di descrittori solo se si ottiene un certo guadagno d'aggregazione.
Se la distanza tra i frame e tra i cluster lo permettono, la comunicazione avviene in modalità INTER ($En_{Coop}$, ammortizzando i flussi), altrimenti, avremo una comunicazione in modalità INTRA ($En_{NonCoop}$), maggiorata dall'invio del solo istogramma iniziale.
Le nostre analisi hanno constatato un risparmio energetico sui cluster a scapito di una riduzione del risparmio sul bottleneck.

I risultati ottenuti vengono analizzati ponendo il caso INTRA come riferimento, studiando quanto migliorano nelle modalità INTER ed Adattativo.
Abbiamo cercato di analizzare le istanze di rete, aumentando il numero di cluster in modo da avere più flussi di traffico, e di studiare possibili variazione nel comportamento della rete. I risultati ottenuti, però essendo valori relativi, non subiscono alterazioni. Dalla Figura 5.7 osserviamo l'andamento del $\Delta$ energetico sul nodo di bottleneck. Relativo al numero medio di hop per raggiungere il Sink in tre diversi casi. Nei casi "*x 2*" e "*x 3*" abbiamo raddoppiato e poi triplicato il numero di camere nell'istanza di partenza.
Mediando tutti i dati si nota che nel caso Cooperativo avremo un risparmio di circa 8%, mentre del 2% per l'Adattativo.

Nel secondo grafico in Figura 5.8 vediamo, invece, un riferimento al livello energetico medio, sui clusters nella singola istanza di rete, nei tre casi precedenti. I valori graficati sono ottenuti in riferimento al relativo valore del caso INTRA calcolando il valore di *Guadagno Cooperativo* ($\Delta En_{Coop}$) e *Guadagno Adattativo*

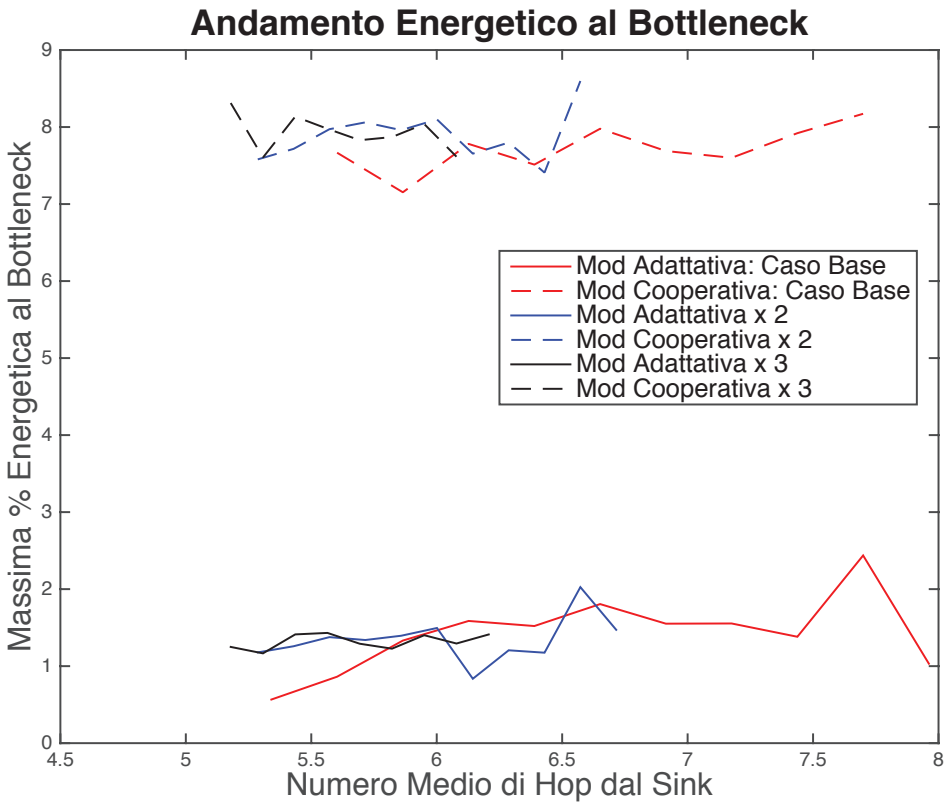



Figura 5.7: Andamento energetico al Bottleneck con numero crescente di sensori camere

$(\Delta En_{Adatt})$:

$$\begin{aligned} \Delta En_{Coop} &= \frac{En_{Coop} - En_{INTRA}}{En_{INTRA}} \\ \Delta En_{Adatt} &= \frac{En_{Adatt} - En_{INTRA}}{En_{INTRA}} \end{aligned} \tag{5.4}$$

Vediamo che, nel caso cooperativo, in media i cluster consumano il 3% in più rispetto al caso INTRA, mentre riescono a risparmiare il 3% nel caso Adattativo.

Dai grafici del capito precedente avevamo osservato un $\eta$ anche del 20%, che nelle simulazioni si riduceva ad un risparmio energetico dell'8%. La causa della riduzione è la distribuzione di probabilità usata per assegnare le distanze tra BoVW, che produce in media valori con un $\eta(i, j)$ nullo.

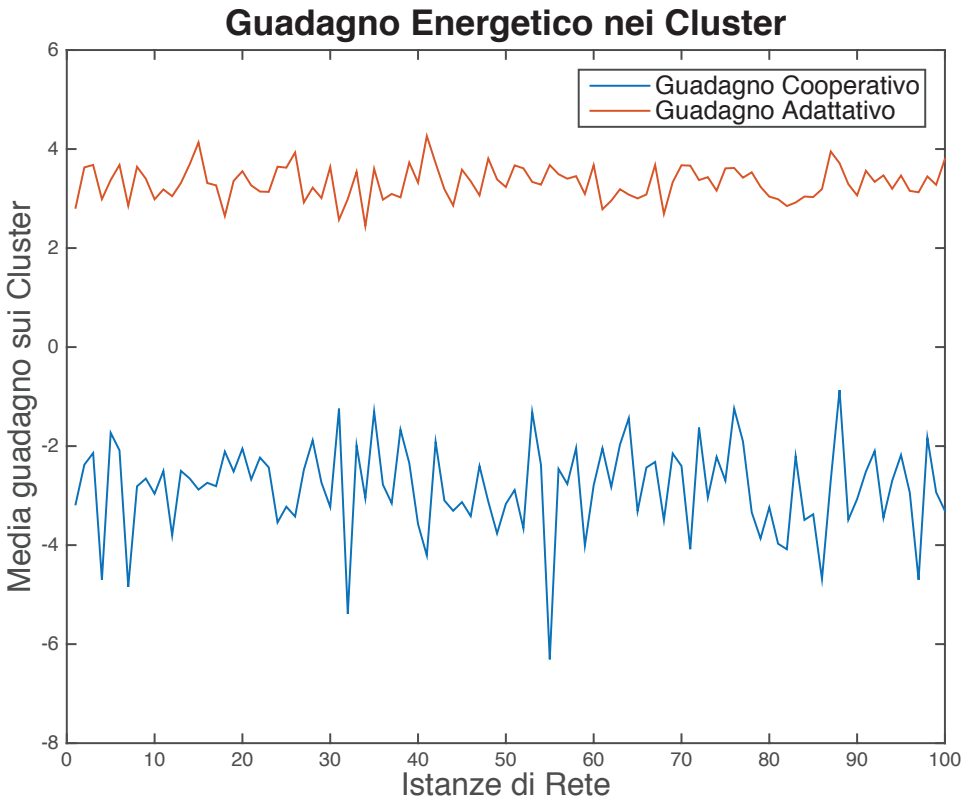



Figura 5.8: Guadagno energetici sui Clusters

Nonostante i valori dedotti non producano alti valori percentuali il riscontro finale è comunque positivo e realistico.

# Capitolo 6

# Conclusioni

Il lavoro proposto ha consentito di trovare una funzione che premette ai cluster di valutare il vantaggio o meno di una collaborazione tra le camere che li costituiscono. La valutazione può avvenire tra le camere senza spedire totalmente il proprio set di descrittori, ma solo con un istogramma da 4096bit, che rappresenti il frame in analisi.
Grazie a questa possibilità abbiamo testato, tramite reti simulate, il livello di risparmio energetico sul nodo più sfruttato (bottleneck), constatando un possibile allungamento medio dell'operabilità della rete fino all'8%, toccando valori di $\eta$ del 20%

Il gudagno energetico medio dell'8% è riferito ad un sistema dove l'invio è sempre aggregato (Modalità INTER nel Capitolo 4). Abbiamo anche valutato un comportamento aggregativo dove le camere potessero valutare (in funzione della distanza) l'utilizzo o meno della cooperazione, raggiungendo risultati inferiori ma positivi (circa 2%). Il risultato nel caso "Adattativo", appena riportato, è riferito al guadagno energetico sul bottleneck. Tale abbassamento è controbilanciato da un aumento del risparmio energetico su tutti i cluster pari, in media, al 3%.

Il modello Adattativo è stato analizzato utilizzando parametri energetici standard per i protocolli della famiglia 802.15, ma non è da escludere l'utilizzo di protocolli 802.11 (ad esempio con i sensori BeagleBone [21]) che consentirebbero una copertura maggiore ed una più concreta possibilità di ammortizzare la spesa energetica nella modalità Adattativa.
Possibili migliorie a livello di spesa energetica sono realizzabili andando a toccare

i descrittori e non solo le tecnologie di trasmissione. L'utilizzo di BRISK potrebbe essere una valida alternativa al descrittore SIFT, che nonostante sia molto efficiente conduce ad una forte spesa energetica.

Nel lavoro appena concluso, la valutazione energetica delle reti simulate è sviluppato con routing statico, basandosi sulla curva di guadagno definita nel modello di Capitolo 4. Uno sviluppo sicuramente degno di approfondimento è l'inserimento di un'ottimizzazione robusta, che prendendo in considerazione la variabile casuale del guadagno permette di realizzare un routing dinamico.
In un caso specifico si potrebbe studiare, nella modalità INTER, la possibilità di stimare lo stato energetico del bottleneck. Nel momento di collasso energetico, generare una nuova topologia di rete dove si eviti il passaggio dal nodo scarico. La probabilità di raggiungere il Sink esiste, ed evitare il vecchio bottleneck permetterebbe alla rete di guadagnare più tempo, sfruttando l'energia rimasta nei nodi restanti.
Come ultima analisi a livello strutturale, si potrebbe valutare l'ampliamento dei cluster. Nel lavoro di tesi prendiamo in considerazione clusters dalle dimensioni fisse di due camere, ma nulla vieta di ampliarne il numero di elementi e studiarne gli effetti.

# Bibliografia


[1] D. G. Lowe, "Object recognition from local scale-invariant features," *The Proceedings of the Seventh IEEE International Conference on Computer Vision (ICCV)*, vol. 2, pp. 1150-1157, 1999.

[2] T. Lindeberg, "Feature detection with automatic scale selection," *International Journal of Computer Vision*, vol. 30, no. 3, pp. 76-116, 1998.

[3] S. Leutenegger, M. Chli, and R. Siegwart, "Brisk: Binary robust invariant scalable keypoints," *IEEE International Conference on Computer Vision (ICCV)*, pp. 2548-2555, 2011.

[4] T. Lindeberg, "Scale-space theory: A basic tool for analysing structures at different scales," *Journal of Applied Statistics*, vol. 21, no. 12, pp. 225-270, 1994.

[5] H. Bay, T. Tuytelaars, and L. J. Van Gool, "Surf: Speeded up robust features," *ECCV*, pp. 404-417, 2006.

[6] E. Rosten, R. Porter, and T. Drummond, "Faster and better: A machine learning approach to corner detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 1, pp. 105-119, 2010.

[7] E. Mair, G. D. Hager, D. Burschka, M. Suppa, and G. Hirzinger, "Adaptive and generic corner detection based on the accelerated segment test," *ECCV*, pp. 183-196, 2010.

[8] M. Calonder, V. Lepetit, M. Ozuysal, T. Trzcinski, C. Strecha, and F. Pascal, "Brief: Computing a local binary de- scriptor very fast," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 7, pp. 1281-1298, 2012.

[9] J. Sivic and A. Zisserman, "Video google: A text retrieval approach to object matching in videos," *International Conference on Computer Vision*, pp. 1470-1477, 2003.

[10] Nister and H. Stewenius, "Scalable recognition with a vocabulary tree. in computer vision and pattern recognition," *IEEE Computer Society Conference*, pp. 2161-2168, 2006.

[11] L. Baroffio, M. Cesana, A. Redondi, M. Tagliasacchi, and S. Tubaro, "Coding visual features extracted from video sequences," *IEEE TRANSACTIONS ON IMAGE PROCESSING*, vol. 23, no. 5, pp. 2262-2276, 2014.

[12] D. Devarajan, Z. Cheng, and R. J. Radke, "Calibrating distributed camera networks," *Proceedings of the IEEE*, vol. 96, no. 10, pp. 1625-1639, 2008.

[13] Z. Cheng, D. Devarajan, and R. J. Radke, "Determining vision graphs for distributed camera networks using feature digests," *EURASIP Journal of Applied Signal Processing, Special Issue on Visual Sensor Networks*, vol. 2007, no. 1, pp. 220-220, 2007.

[14] A. van den Hengel, A. Dick, H. Detmold, A. Cichowski, and R. Hill, *Computer Vision ACCV 2007*. Springer Berlin Heidelberg, 2007.

[15] N. G. Kiran Maraiya, Kamal Kant, "Wireless sensor network: A review on data aggregation," *International Journal of Scientific & Engineering Research*, vol. 2, no. 4, pp. 1-6, 2011.

[16] C. M. Christoudias, R. Urtasun, and T. Darrell, "Unsupervised feature selection via distributed coding for multi-view object recognition," *Computer Vision and Pattern Recognition, 2008. CVPR 2008*, pp. 1-8, 2008.

[17] T. Jaakkola and D. Haussler, "Exploiting generative models in discriminative classifiers," *Advances in neural information processing systems (NIPS)*, pp. 487-493, 1999.

[18] H. Jegou, M. Douze, C. Schmid, and P. Perez, "Aggregating local descriptors into a compact image representation," *Computer Vision and Pattern Recognition (CVPR)*, pp. 3304-3311, 2010.

[19] S. Colonnese, F. Cuomo, and T. Melodia, "An empirical model of multi-view video coding efficiency for wireless multimedia sensor networks," *IEEE TRANSACTIONS ON MULTIMEDIA*, vol. 15, no. 8, pp. 1800-1814.

[20] W. Rabiner Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," 2000.

[21] "Beaglebone, http://beagleboard.org/."

# Elenco delle figure

# Elenco delle tabelle