ALMA MATER STUDIORUM · UNIVERSITÀ DEGLI STUDI DI BOLOGNA

FACOLTA' DI SCIENZE MATEMATICHE, FISICHE E NATURALI
Corso di Laurea in Scienze e Tecnologie Informatiche

REALIZZAZIONE DI UNA APPLICAZIONE PER L'ANALISI DI IMMAGINI IPERSPETTRALI

Tesi di Laurea in Basi di Dati

Presentata da:
FABRIZIO MASINI

Relatore:
Prof. **DARIO MAIO**

Correlatore:
Dott. **MATTEO FERRARA**

Sessione II

Anno accademico 2012/2013

Indice

Introduzione

Negli ultimi anni si è visto nascere un crescente interesse per le procedure di telerilevamento, e per le tecniche di analisi e interpretazione delle immagini acquisite.

Il *Telerilevamento* (in inglese *Remote Sensing*) è una disciplina che permette di monitorare la superficie terrestre per mezzo di sensori aereospaziali, e di ricavare informazioni utili sugli ambienti e sugli oggetti posti su di essa. Nella fase di rilevamento vengono inviati segnali elettromagnetici sulla superficie terrestre, e vengono utilizzati appositi sensori per l'analisi e la misurazione dei segnali in risposta. I sensori impiegati sono molteplici e si distinguono sia per la loro tipologia (trasportati da aerei, da palloni sonda o da satelliti), sia per le loro caratteristiche tecniche.

Le prime tecniche di telerilevamento sono nate a metà degli anni '80 e sono state ideate per analizzare rocce e minerali in campo geologico. Successivamente sono state utilizzate in ambito archeologico, per lo studio di reperti e documenti storici, e in ambito aero-spaziale, per studiare fenomeni climatici ed agenti atmosferici. Fino alla prima metà degli anni '90, le immagini iperspettrali e le tecniche di telerilevamento sono rimaste oggetto di studio di enti di ricerca e aziende di nicchia, di specifici settori aziendali, soprattutto per questioni economiche visti gli alti costi dei dispostivi di acquisizione e dei calcolatori necessari per processare moli di dati così elevate. Nell'ultimo ventennio, complici il progresso tecnologico e l'avvento dei nuovi calcolatori, le immagini iperspettrali hanno preso piede in numerosi campi di applicazione. Ad oggi contano decine e decine di settori applicativi diversi, da quelli di ricerca scientifica a quelli militari, dalle applicazioni in ambito civile a quelle di analisi dello sviluppo urbano-industriale. Uno dei principali settori di applicazione è quello

mirato alla redazione di carte tematiche o statistiche, relative a cambiamenti di fenomeni nel tempo, come ad esempio lo sviluppo urbano o demografico in una determinata zona territoriale, oppure lo studio di cambiamenti dello scenario ambientale (geologia, oceanografia, innalzamento del livello delle acque, fratture del terreno e spostamento delle placche tettoniche).

Inoltre l'evoluzione tecnologica degli ultimi anni, ha portato ad un miglioramento sensibile nella qualità e nella risoluzione delle immagini del dominio del telerilevamento ottico. I laboratori di ricerca e gli enti universitari hanno affinato le tecniche di analisi e di elaborazione già esistenti, e hanno sviluppato nuovi algoritmi di estrazione delle informazioni statistiche.

Negli ultimi anni sono sorte nuove necessità legate all'analisi e alla memorizzazione di grandi quantità di immagini di grandi dimensioni (ad esempio le immagini iperspettrali possono raggiungere le centinaia di bande, e occupare decine e decine di gigabyte di spazio dati). L'elaborazione di immagini di tali dimensioni, e la richiesta di estrarre particolari informazioni da esse, rappresenta una sfida scientifica e tecnologica di grande rilievo.

Al pari delle tecniche di telerilevamento, nell'ultimo decennio, si è visto crescere un particolare interesse per le tecniche di classificazione automatica delle immagini satellitari.

La *classificazione non supervisionata* (anche detta *Clustering*) nell'ambito delle immagini, consiste nell'analizzare i pixel di un'immagine e "etichettarli" secondo precise regole di raggruppamento, in modo da ottenere gruppi di pixel omogenei, appartenenti a *classi* d'interesse. Il clustering viene applicato alle immagini satellitari per cercare di riconoscere le particolari categorie di oggetti che compongono l'immagine, e classificarli secondo regole di *similarità tra cluster*. Tipicamente si cerca di riconoscere le tipologie di coperture del suolo presenti in un'immagine telerilevata e si cerca di istruire un calcolatore, in modo che svolga la classificazione senza dover richiedere informazioni preliminari al programmatore (attività non supervisionata).

Lo *scopo* di questo progetto di tesi è quello di sviluppare un applicativo software che permetta la gestione e l'analisi di immagini satellitari. L'applicativo dovrà mettere a disposizione dell'utente strumenti di analisi delle immagini e di estrazione delle informazioni statistiche, e dovrà dare la possibilità all'utente di categorizzare le varie zone delle immagini secondo classi predefinite dall'utente stesso. Inoltre verranno effettuati dei test per valutare l'accuratezza di alcuni algoritmi di clustering presenti in letteratura, applicati alle immagini telerilevate. L'implementazione e lo studio di tali tecniche servirà a far luce su pregi e difetti caratteristici di ognuna delle tecniche studiate.

Il lavoro di tesi verrà strutturato come segue.

Il *primo capitolo* descriverà nel dettaglio le tecniche e le caratteristiche del telerilevamento, includendo brevi cenni teorici riguardanti lo spettro elettromagnetico e la teoria dei colori. Inoltre verranno elencate le principali differenze tra le varie tipologie di immagini telerilevate, presentando numerose immagini d'esempio.

Nel *secondo capitolo* verranno introdotte le scelte implementative che hanno portato dalla progettazione concettuale alla progettazione logico-strutturale dell'applicazione oggetto della tesi. Verrà data particolare enfasi alla descrizione delle classi utilizzate per la memorizzazione e la gestione delle immagini iperspettrali. Inoltre verranno illustrate le principali funzionalità fornite dal front-end grafico dell'applicazione.

Nel *terzo capitolo* saranno citati brevi cenni teorici legati alla *Visione Artificiale* e al *Pattern Recognition*, per poi introdurre gli algoritmi di *Clustering* implementati.

Il *quarto capitolo* riporterà i risultati ottenuti su alcune immagini satellitari dagli algoritmi di clustering implementati. I vari algoritmi di clustering verranno confrontati fra loro e i risultati riportati verranno commentati mettendone in luce pregi e difetti.

Infine verranno esposte le conclusioni scaturite dal lavoro di tesi e verranno delineati alcuni possibili sviluppi futuri.

Capitolo 1
Il Telerilevamento

1.1. Generalità

Il termine "telerilevamento" deriva dal greco ed è composto dalle parole "tele" e "rilevamento", che letteralmente significano "osservazione da lontano".

Il Telerilevamento (o *Remote Sensing*), è una disciplina tecnico-scientifica che permette di ricavare informazioni quantitative e qualitative, misurando l'emissione, la trasmissione e la riflessione di radiazioni elettromagnetiche di superfici e corpi posti ad enorme distanza dall'osservatore.

Solitamente il rilievo di una superficie con tecniche di telerilevamento prevede tre fasi distinte:

- La raccolta dei dati (tramite aerei, satelliti o sonde spaziali)
- L'elaborazione (tramite tecniche e applicativi software)
- L'analisi (interferometria, analisi spettrale, analisi statistica, …)

Nei successivi paragrafi verranno presentate le caratteristiche che contraddistinguono i sistemi di telerilevamento, e che permettono di estendere e migliorare le capacità percettive dell'occhio umano.

1.2. Spettro elettromagnetico

Lo spettro elettromagnetico (o spettro EM) è l'insieme di tutte le possibili onde elettromagnetiche caratterizzate da una lunghezza d'onda e da una frequenza. Queste due componenti sono inversamente proporzionali, per questo, al crescere della frequenza calerà la lunghezza d'onda del segnale misurato. Lo spettro rappresenta un insieme di frequenze continue da zero ad infinito.

E' quindi possibile definire una classificazione delle onde elettromagnetiche (si veda Tabella 1.1), partendo da quelle con frequenze minori e proseguendo in ordine crescente:

Tipologia di radiazione	Frequenza	Lunghezza d'onda
Onde Radio	$\leq 300\ MHz$	$\geq 1\ m$
Microonde	$300\ MHz - 300\ GHz$	$1\ m - 1\ mm$
Infrarossi	$300\ GHz - 428\ THz$	$1\ mm - 700\ nm$
Luce Visibile	$428\ THz - 749\ THz$	$700\ nm - 400\ nm$
Ultravioletti	$749\ THz - 30\ PHz$	$400\ nm - 10\ nm$
Raggi X	$30\ PHz - 300\ EHz$	$10\ nm - 1\ pm$
Raggi Gamma	$\geq 300\ EHz$	$\leq 1\ pm$

Tabella 1.1 - Classificazione delle radiazioni elettromagnetiche.

1.2.1. Spettro del visibile

Lo "spettro del visibile" si trova nella parte centrale dello spettro ottico, il quale comprende anche i raggi infrarossi e quelli ultravioletti. E' composto da quella parte di spettro elettromagnetico che include tutte le colorazioni percepite dall'occhio umano, partendo dal rosso fino ad arrivare al viola (si veda Figura 1.1).

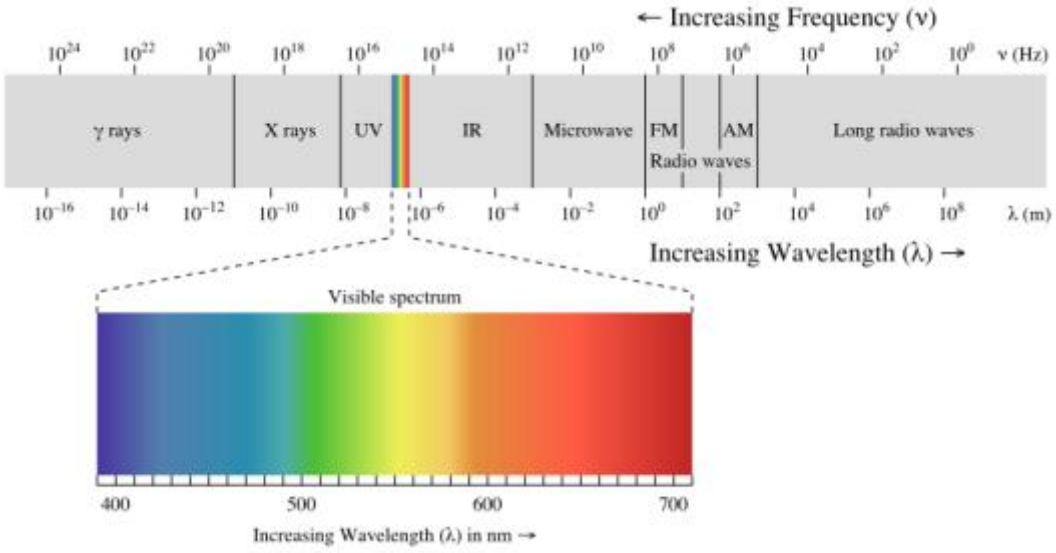


Figura 1.1 - Rappresentazione dello spettro elettromagnetico, con dettaglio dello spettro visibile.

Le lunghezze d'onda contenute nell'intervallo del visibile variano dai 400_{nm} ai 700_{nm}, mentre le frequenze cadono nell'intorno tra i $428\,THz$ e i $749\,THz$. In media la massima sensibilità dell'occhio umano si percepisce intorno ai $540\,THz$.

1.3. Teoria delle immagini

I sensori per il telerilevamento forniscono un insieme di misure di radianza che vanno a comporre le immagini digitali che saranno oggetto dei processi di analisi ed elaborazione dei dati.

Ogni sensore di telerilevamento è contraddistinto da una serie di parametri che influenzeranno il processo di elaborazione.

- *Risoluzione spaziale*: indica la dimensione dell'area geografica rappresentata da un singolo pixel. Ad oggi i sensori più precisi riescono a rappresentare con un pixel un'area quadrata di alcune decine di centimetri.
- *Risoluzione spettrale*: rappresenta la larghezza delle bande spettrali che il sensore è in grado di acquisire.
- *Risoluzione radiometrica*: rappresenta il numero di bit con cui ogni pixel viene memorizzato, o meglio, la cardinalità delle intensità di radiazione che vengono

identificate dal sensore. Quindi con una risoluzione di 1 bit (si potranno rappresentare solamente 2 livelli (bianco e nero), con risoluzioni di 8, 16 o 32 bit si rappresenteranno immagini con 2^b livelli, dove b è il numero di bit scelto (si veda Figura 1.2).

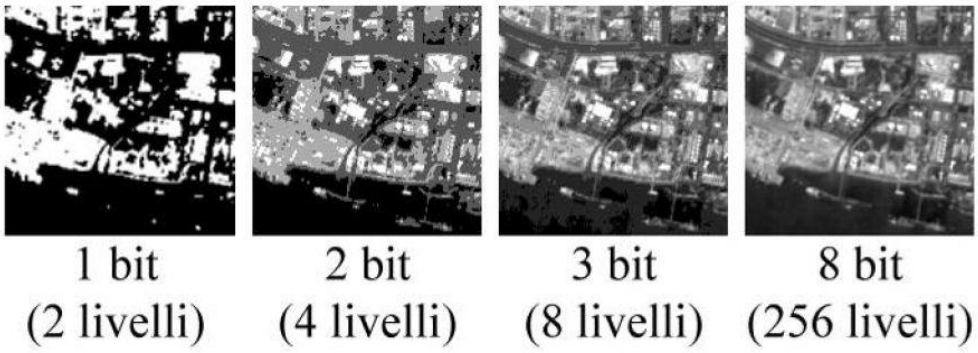


Figura 1.2 - Esempio di immagine a diversi livelli di risoluzione radiometrica

- *Risoluzione temporale:* indica il periodo trascorso tra un rilevamento e il successivo. Questo parametro è molto utile nei campi di applicazione legati allo studio dei fenomeni e dei cambiamenti nel tempo di una specifica area territoriale.

In generale, le immagini ottenute durante il processo di raccolta possono appartenere a tre differenti famiglie di immagini digitali, con diverse caratteristiche e differenti livelli di dettaglio:

- *Immagini Pancromatiche*, utilizzano tutta la banda dello spettro ottico, e rappresentano una sola banda con un'elevata risoluzione spaziale (si veda Figura 1.3).
- *Immagini Multispettrali*, normalmente possiedono un numero di bande compreso tra tre e quindici, e sono caratterizzate da una bassa risoluzione spaziale.
- *Immagini Iperspettrali*, possiedono un numero elevato di bande, solitamente superiore al centinaio, ed una bassa risoluzione spaziale.

Figura 1.3 - Immagine pancromatica (in scala di grigio) ad alta definizione spaziale.
(San Francisco, California, USA - Satellite SPOT-5).

Nei seguenti due paragrafi verranno trattate in dettaglio le *Immagini Multispettrali* e *Iperspettrali*.

1.3.1. Immagini Multispettrali

Le immagini multispettrali rappresentano la principale categoria di immagini acquisite dai sensori di telerilevamento. A differenza delle immagini pancromatiche, che per ogni pixel memorizzano un solo valore, le immagini multispettrali suddividono lo spettro elettromagnetico in tante bande (o canali), e assegnano ad ogni pixel un valore appartenente a tali bande. Questi valori vanno a comporre lo *Spazio di Feature* dell'immagine multispettrale.

Le immagini multispettrali sono nate con il compito di memorizzare un maggior numero di informazioni rispetto alle immagini pancromatiche, estendendo le informazioni archiviate nei singoli canali RGB. I primi impieghi di queste immagini hanno avuto luogo in campo spaziale e archeologico, per lo studio di reperti e documenti, difficilmente analizzabili ad occhio nudo (si veda Figura 1.4).

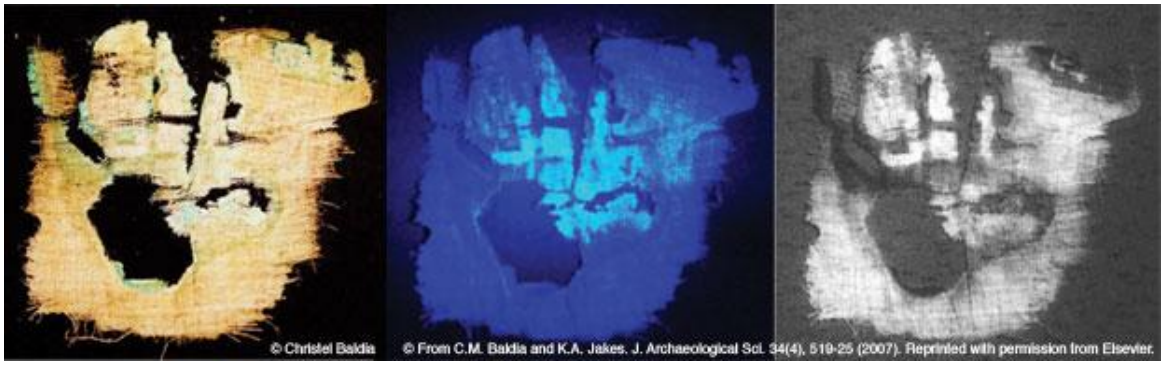


Figura 1.4 - Reperto archeologico scansionato con la tecnica multispettrale.

E' possibile introdurre una classificazione delle bande spettrali suddivise secondo precisi range di lunghezza d'onda (questa suddivisione risulta comunque approssimativa poiché i valori esatti dipendono dal sensore preso in considerazione, e dalla sua precisione):

- *Blue* $(450\ nm - 520\ nm)$
  Utilizzata per le riprese atmosferiche e per le riprese oceanografiche profonde.
- *Green* $(520\ nm - 600\ nm)$
  Utilizzata per le riprese di vegetazione e ambienti acquatici di media profondità.
- *Red* $(600\ nm - 690\ nm)$
  Utilizzata per ritrarre oggetti costruiti dall'uomo, acque di bassa profondità e alcuni tipi di vegetazione.
- *NIR "Near Infrared"* $(750\ nm - 900\ nm)$
  Dedicata al telerilevamento di ambienti naturali.
- *MIR "Middle Infrared"* $(1550\ nm - 2350\ nm)$
  Utilizzata per ritrarre particolari ambienti geologici, umidità ed eventi atmosferici.
- *TIR "Thermal Infrared"* $(10400\ nm - 12500\ nm)$

Analizza l'emissione di radiazioni e non la riflessione. Consente di rilevare contesti atipici quali fiamme, differenze termiche nelle correnti marine, ambienti notturni.

1.3.2. Immagini Iperspettrali

L'*Hyperspectral Imaging* e il *Multispectral Imaging* appartengono entrambi ad una più grande famiglia di tecniche di rappresentazione delle immagini detta *"Spectral Imaging"* o *"Spectral Anlysis"*.

Nelle immagini iperspettrali, come in quelle multispettrali, ogni elemento dell'immagine (pixel) non è costituito da un semplice valore monocromatico (immagini pancromatiche o in scala di grigio), o da una terna di valori (immagini a colori RGB), ma da un insieme di valori appartenenti allo spettro elettromagnetico.

La distinzione tra immagini multispettrali e immagini iperspettrali (Figura 1.5) è basata sul numero di bande che vengono memorizzate, e sullo spettro di frequenze rappresentato.

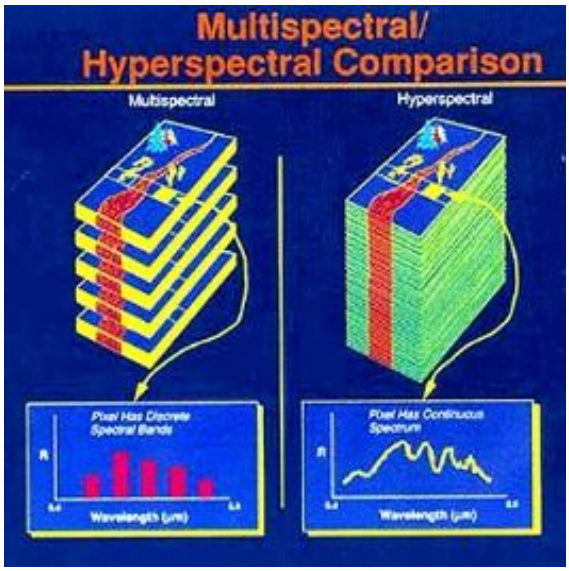


Figura 1.5 - Confronto tra Multispectral e Hyperspectral Imaging.

In particolare, le prime possiedono un numero limitato di bande e non sono tenute a memorizzare bande di frequenze contigue. Ovvero è possibile che all'interno di una stessa immagine vi siano alcune bande relative allo spettro del visibile, ed altre bande

relative a quello degli infrarossi, e che lo spazio di frequenze tra di esse non sia totalmente coperto dalle restanti bande. Le immagini iperspettrali invece possiedono un numero elevato di bande, che rappresentano intervalli discreti dello spettro elettromagnetico, e producono uno spettro continuo per ogni pixel ritratto nella scena.

Un'immagine iperspettrale può dunque essere considerata come un cubo di dati, costituito da tanti piani quante sono le bande che compongono lo spettro telerilevato, e di larghezza e altezza pari alle dimensioni dell'area catturata. Idealmente è possibile considerare le due dimensioni spaziali come se fossero appoggiate sulla superficie ritratta e la terza dimensione (spettrale) perpendicolare ad esse (Figura 1.6). Se ci si sposta lungo la direzione spettrale, si ottengono valori e colorazioni diverse, e se si preleva una delle bande telerilevate, si ottiene un'immagine monocromatica. Prelevando le informazioni relative ad un singolo punto dell'immagine, si ottiene lo spettro continuo di quel determinato pixel.

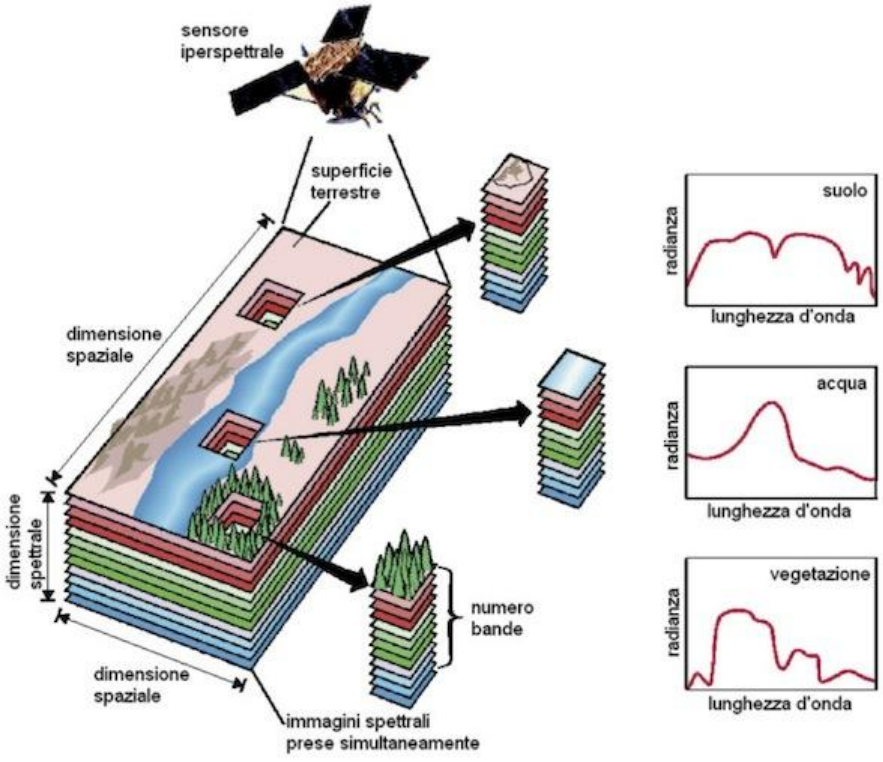


Figura 1.6 - Rappresentazione grafica di un cubo di dati iperspettrale.

Spesso lo spettro associato a ciascun pixel è uno *"spettro di riflettanza"*, che può contenere fino a qualche centinaio di valori monocromatici.

La ***riflettanza*** è un numero puro, compreso tra 0 e 1, che rappresenta l'efficienza con la quale una superficie riesce a riflettere la luce di una data lunghezza d'onda (Figura 1.7).

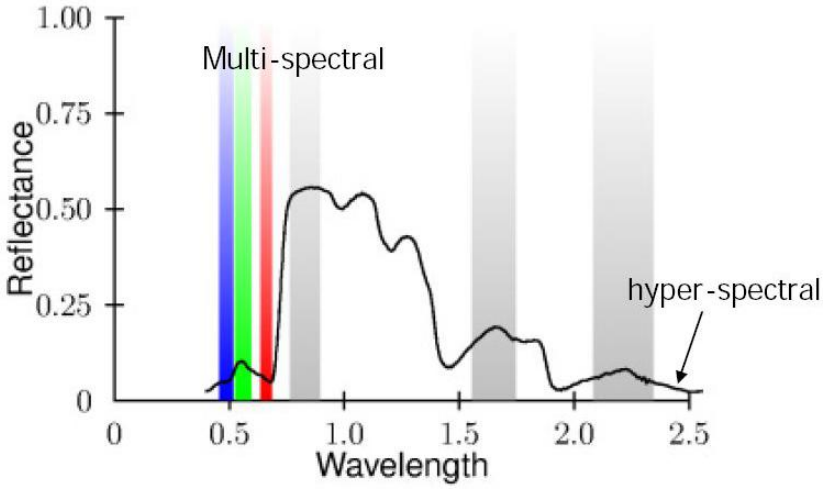


Figura 1.7 - Grafico rappresentante la risoluzione spettrale delle bande, al variare di *riflettanza* e *lunghezza d'onda*.

1.3.3. Modelli di rappresentazione

Le immagini multispettrali e iperspettrali possono essere visualizzate con tre diversi modelli di rappresentazione.

- ***Gray-scale (scala di grigio)***

  È la rappresentazione utilizzata per la visualizzazione di un solo canale dell'immagine. Ogni punto dell'immagine telerilevata viene rappresentato da un singolo pixel in tonalità di grigio.

- ***True-Color***

  Vengono utilizzati solamente i canali RGB (se presenti), oppure vengono scelte le bande spettrali che meglio li rappresentano. E' utile per l'analisi di oggetti prodotti dall'uomo, ed è facile da comprendere e interpretare, in quanto rappresenta la scena così come l'avrebbe ritratta l'occhio umano (si veda Figura 1.8(a)).

- ***False-Color***

  Vengono scelte particolari combinazioni di bande spettrali, in modo da far emergere specifici elementi dell'immagine. Questa rappresentazione risulta

molto differente dalla rappresentazione True-Color, a cui l'occhio umano è abituato, ma consente di enfatizzare alcuni elementi che vengono maggiormente irradiati da segnali a frequenze lontane da quelle dei canali RGB (si veda Figura 1.8(b)).

(a) (b)

Figura 1.8 - (a) Rappresentazione *True-Color*. (b) Rappresentazione *False-Color* (NIR-Red-Green). (Immagine rilevata dal satellite Landsat sulla Baia di Chesapeake, Baltimora, USA).

Esistono alcune combinazioni di bande spettrali che vengono comunemente utilizzate in campo geo-satellitare:

- *NIR-Red-Green*

  È la combinazione più utilizzata e consente di enfatizzare gli elementi boschivi. La vegetazione viene irradiata maggiormente dai segnali con frequenze NIR, quindi andando a sostituire questi canali con il canale "Red" della visualizzazione RGB, si otterrà una innaturale colorazione rossa (si veda Figura 1.9).

Figura 1.9 - Rappresentazione *NIR-Green-Red* della città di Baltimora, USA.
(Immagine ottenuta dal satellite ASTER della NASA).

- *MIR-NIR-Blue*

  Questa combinazione permette di visualizzare contesti diversi, che vanno dagli incendi (si veda Figura 1.10), alle profondità marine.

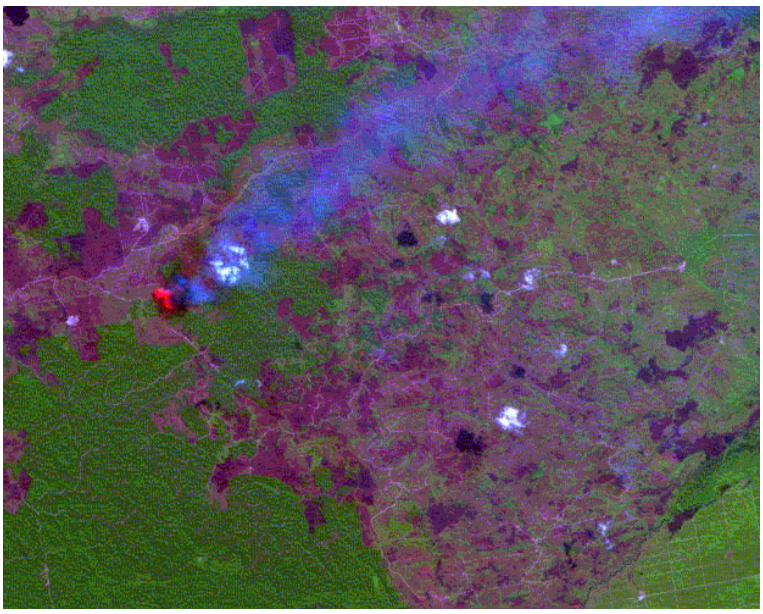


Figura 1.10 - Rappresentazione *MIR-NIR-Blue* di un incendio boschivo.
(Immagine rilevata dal satellite SPOT 4).

1.4. Archivi pubblici di immagini telerilevate

Per popolare l'archivio di immagini, sono state ricercate sul web le principali piattaforme di condivisione gratuite di immagini satellitari. La procedura di popolamento del database ha richiesto parecchio tempo, ed una particolare attenzione nella selezione delle immagini adatte allo scopo e al contesto applicativo.

Qui di seguito verranno elencati alcuni dei siti web dai quali sono state prelevate le immagini:

- Earth Explorer - USGS.Gov (United States Geological Survey) [1]
- Landsat Mission - USGS.NASA (National Aeronautics and Space Administration) [2] - [3]
- Glovis – USGS.Viewer (Global Visualization Viewer – USGS Government) [4]
- Satellite Imaging Corporation [5]
- EyeFind – RapidEye [6]
- GeoEye [7]

Al termine della procedura di popolamento, l'archivio di immagini satellitari è risultato essere di circa 200 immagini multispettrali e iperspettrali, di diversi formati dati, e con differenti misure e dimensioni.

1.5. Campi di applicazione

Il telerilevamento è nato con lo scopo di analizzare ed ispezionare materiale geologico (minerali, rocce, stratificazioni nel terreno, ecc.), per poi spostarsi verso un'altra gamma di problemi quali l'archeologia e l'astronomia. Con il passare degli anni si sono affinate le tecniche di analisi ed elaborazione dei dati, e si sono studiati nuovi strumenti e sensori utili per il telerilevamento. La ricerca si è concentrata maggiormente sullo studio di satelliti artificiali più affidabili e accurati, in grado di resistere anche nei contesti climatici più difficili, e su sensori di telerilevamento in grado di raccogliere informazioni sempre più precise e dettagliate.

Anche se le applicazioni di telerilevamento rimangono uno strumento di nicchia, e vengono utilizzate da pochi enti privati, i laboratori di ricerca e gli enti universitari continuano lo studio su queste tecniche di analisi ed elaborazione di dati telerilevati, sperando che un giorno possano prendere piede anche in campo pubblico.

Ad oggi il telerilevamento vanta diversi campi di applicazione:

- **Settore climatico-ambientale**
  - *Geologia, Idrologia e Glaciologia*
  - *Espansione e disboscamento di aree verdi (flora, fauna, ecc)*

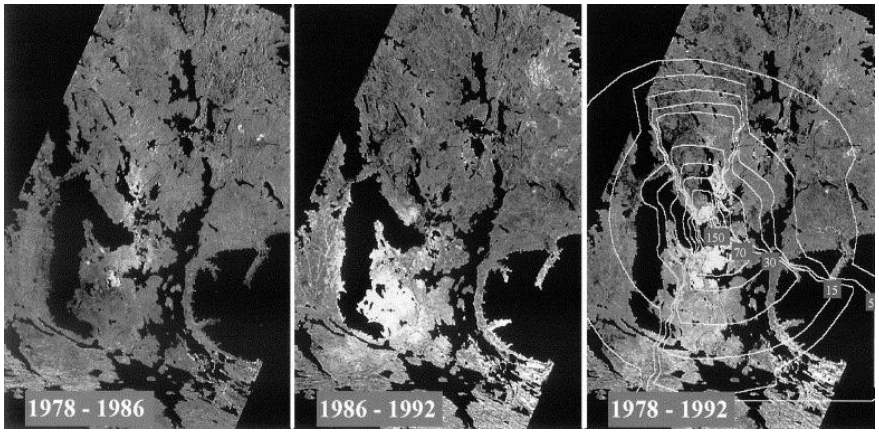


Figura 1.11 - Analisi multispettrale temporale della stessa zona territoriale.
Nella terza immagine da sinistra, le linee isocline indicano la concentrazione media di SO_2nell'aria, calcolata in base alle emissioni a lungo termine.
[Immagine prelevata dal satellite Landsat-MSS su Monchegorsk (Russia)].

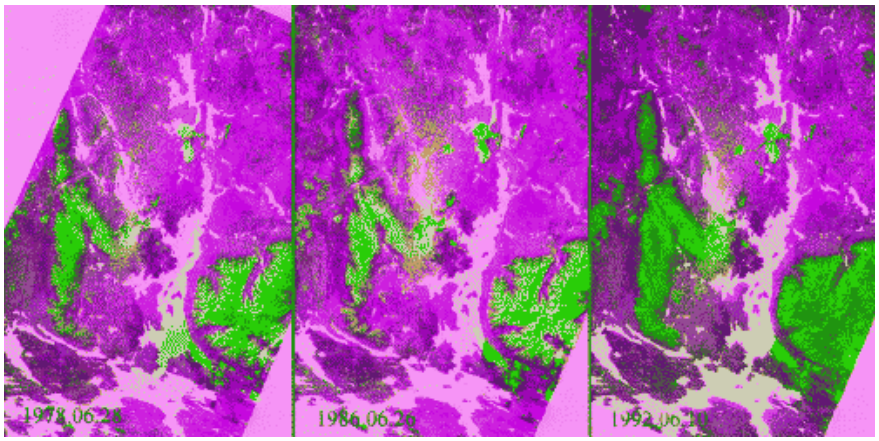


Figura 1.12 - Classificazione delle aree verdi, per verificare il disboscamento nel tempo.
(*False*-Color, in verde le aree non boschive, nelle tonalità di rosa le aree naturali).
[Immagine prelevata dal satellite Landsat-MSS su Monchegorsk (Russia)].

- **Settore militare**
  - Indagini militari
  - Videosorveglianza

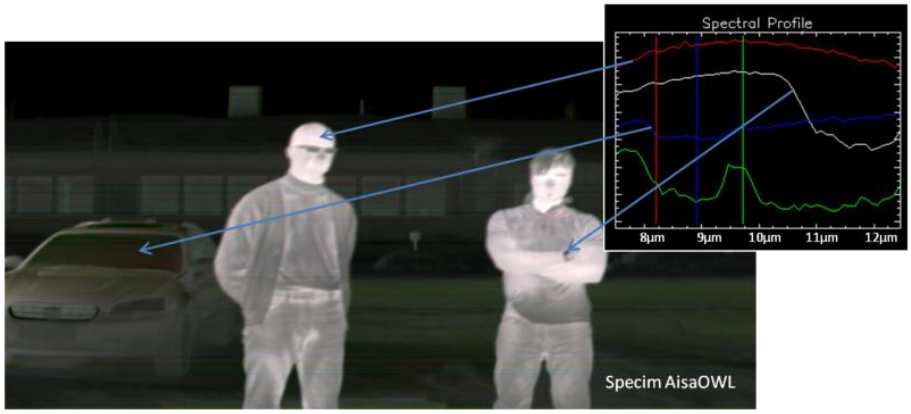


Figura 1.13 - Analisi iperspettrale di segnali infrarossi termici per l'individuazione di profili diversi.

- **Settore aerospaziale**
  - *Monitoraggio dell'atmosfera*
  - *Dinamica degli inquinanti*
  - *Climatologia*

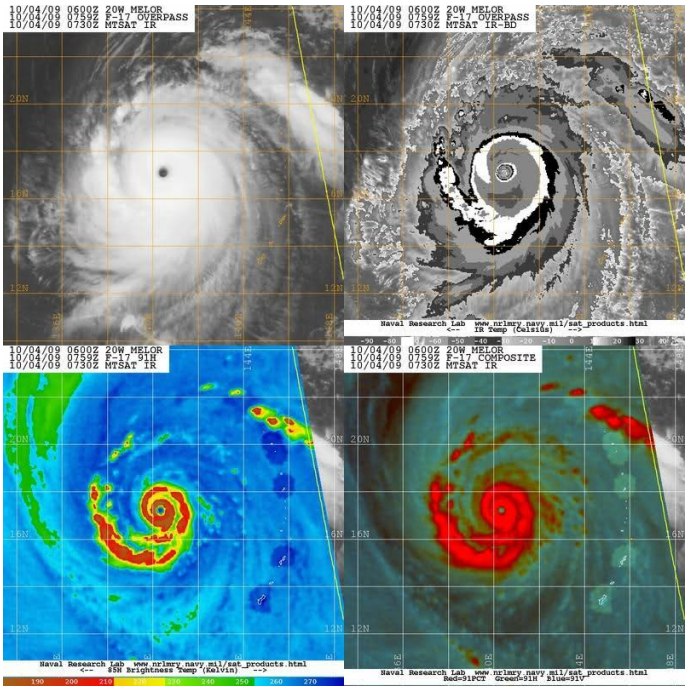


Figura 1.14 - Immagini multispettrali del "Tifone Melor" - Analisi dello spettro infrarossi. (Giappone - Ottobre 2009).

- **Settore di gestione e controllo del territorio**
  - *Topografia e Cartografia*
  - *Sviluppo sostenibile*
  - *Studio dell'impatto ambientale*
  - *Lotta all'abusivismo*
  - *Previsione delle catastrofi*
  - *Evoluzione urbanistica*

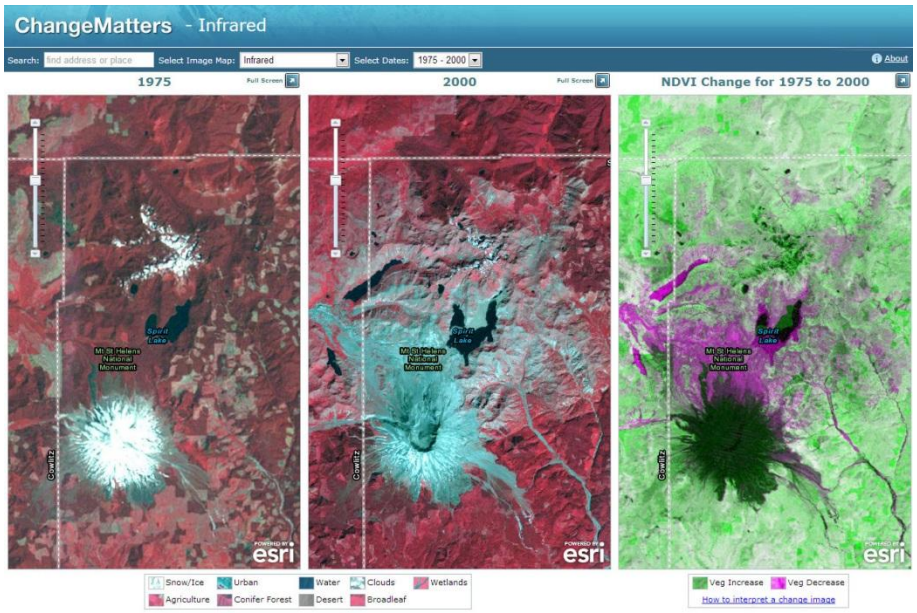


Figura 1.15 - Evoluzione urbanistica nel tempo (dal 1975 al 2000) nella zona di Mount St. Helens National Volcanic Monument (Washington USA). Immagini prelevate dal satellite ESRI-Landsat e visualizzate in *False-Color*. Sito web: [8].

- **Altri settori**
  - *Archeologia (stato di decomposizione di reperti, documenti, ecc)*
  - *Analisi chimica di oggetti, elementi e materiali*
  - *Medicina (risonanze, radiografie, ecc)*

Capitolo 2
Progettazione dell'applicazione

2.1. Linguaggio e ambiente di sviluppo

L'ambiente di sviluppo utilizzato è **Visual Studio 2010**, prodotto d'eccellenza di casa *Microsoft*, è uno degli IDE *(Integrated Development Environment)* più utilizzati al mondo. Viene definito un ambiente RAD *(Rapid Application Development)* in quanto permette ai programmatori di velocizzare la stesura del codice grazie a strumenti quali l'*IntelliSense* o il *Designer* visuale per le form. Supporta un numero elevato di linguaggi di programmazione, tra i quali: *C, C++, C#, ASP .Net* e *Visual Basic .Net.*

Il linguaggio scelto per lo sviluppo software è il linguaggio **C#**, un linguaggio di programmazione orientato agli oggetti, ideato dall'azienda Microsoft.

E' nato dalle esperienze precedenti di linguaggi come C++, Delphi e Java, e possiede innumerevoli funzioni e migliorie rispetto a questi ultimi. E' un linguaggio facile da interpretare, altamente documentato e notoriamente conosciuto sul web, possiede enormi vantaggi relativi all'interfaccia grafica e ai costrutti per la programmazione avanzata.

Oltre a tutti questi vantaggi, la scelta del linguaggio di programmazione è stata fortemente vincolata dalle librerie necessarie al raggiungimento dello scopo. Nel prossimo paragrafo verranno introdotte le librerie *BioLab* e *GDAL*, entrambe prodotte per il linguaggio C#.

2.2. Librerie esterne

Per sviluppare l'applicazione software richiesta si è deciso di utilizzare due librerie C# esterne. Entrambe le librerie forniscono collezioni di classi e metodi utili all'apertura, all'elaborazione e all'analisi di immagini e di contenuti iperspettrali.

2.2.1. BioLab

La libreria ***BioLab*** è un prodotto del laboratorio di ricerca denominato ***"Biometric System Laboratory"*** facente parte del *DISI (Dipartimento di Informatica - Scienza e Ingegneria)* dell'Università di Bologna. [9]

Questa libreria fornisce una collezione di classi, metodi e strutture dati che permettono il caricamento, il salvataggio e la manipolazione di immagini memorizzate nei formati più comuni (es. BMP, PNG, JPG, ecc.). Al suo interno si trovano diverse classi che permettono operazioni di *Image Processing,* e numerosi componenti che ne permettono una facile visualizzazione.

2.2.2. GDAL

La libreria utilizzata per il caricamento e l'elaborazione delle immagini satellitari è la libreria **GDAL (Geospatial Data Abstraction Library)** [10].

GDAL è una libreria *Open Source* facente parte di un progetto di più ampio respiro, coordinato dall'ente non-profit *OSGeo (Open Source Geospatial Foundation)* che promuove ed investe sulle tecnologie aperte e sui dati geospaziali.

La libreria GDAL è in grado di rappresentare svariati formati di dati geografici (sia raster, che vettoriali) utilizzando un modello di dati astratto e permettendo alle applicazioni che ne fanno utilizzo, di accedere, modificare ed elaborare tali informazioni indipendentemente dal particolare formato con cui sono memorizzate.

2.2.2.1. Formati riconosciuti

Grazie alla libreria GDAL l'applicazione sviluppata supporta tutti i formati più comunemente utilizzati per la memorizzazione di immagini telerilevate (Tabella 2.1):

Extension	Long Format Name
*.lan	Erdas 7.x (Earth Resources Data Analysis)
*.gis	Erdas 7.x (Earth Resources Data Analysis)
*.ecw	Erdas Compressed Wavelets
.tiff/.tif	Tagged Image Format File (GeoTiff o BigTiff)
*.envi	Envi Labelled Raster

Tabella 2.1 - Estensioni e formati supportati dall'applicazione sviluppata.

2.3. Progettazione architetturale

Il progetto architetturale dell'applicazione è nato da un'attenta fase di "*Analisi dei Requisiti"*. Studiando il contesto applicativo e definendo le principali funzionalità richieste, sono stati sviluppati tre ***moduli software*** principali (si veda Figura 2.1):

- *APPLICATION CORE*

  In questo modulo sono state implementate le classi che gestiscono le interazioni tra l'interfaccia utente e la parte algoritmica dell'applicazione. L'applicazione contiene un *Core* centrale che gestisce le chiamate alla visualizzazione delle Form contenute nella GUI, e permette l'interazione con le immagini iperspettrali.
- *GUI (GRAPHICAL USER INTERFACE)*

  Questo modulo fornisce un'interfaccia grafica per il modulo applicativo, e gestisce tutte le interazioni tra l'utente e l'applicazione. Per svolgere questo compito la GUI sfrutta due importanti componenti forniti dall'ambiente .Net : *Windows Form* e *User Control*. (Nel Paragrafo 2.4 verranno descritte dettagliatamente le parti relative all'interfaccia utente).
- *HYPERSPECTRAL IMAGES*

  Questo modulo permette il caricamento, l'analisi e l'elaborazione di immagini iperspettrali, comprendendo tutte le classi adibite alla gestione di dati iperspettrali e all'estrazione di informazioni statistiche. (Nel Paragrafo 2.5 verranno descritte le principali classi di questo modulo).

Al centro dello schema (Figura 2.1) è presente una classe inter-modulo denominata *"HyperspectralProject"*, che viene utilizzata per salvare o caricare dati di precedenti sessioni di lavoro, e funge da contenitore per tutti i dati rilevati dalle immagini iperspettrali. (Questa particolare classe verrà descritta dettagliatamente nel Paragrafo 2.6).

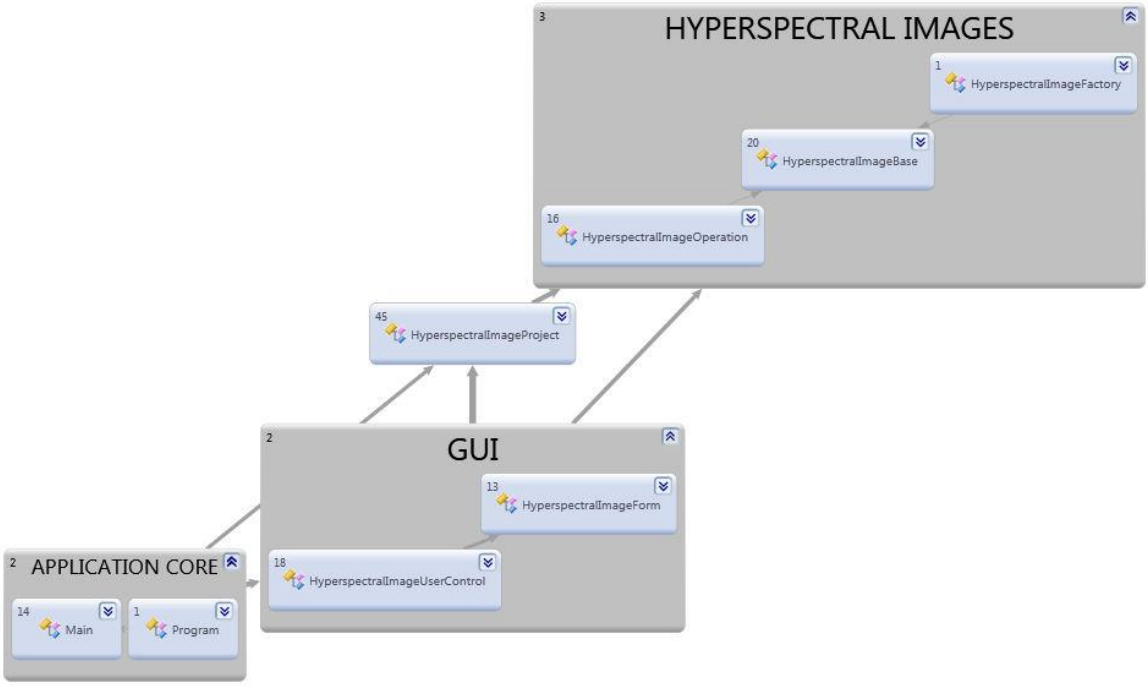


Figura 2.1 - Moduli architetturali dell'applicazione (ottenuto da Visual Studio 2010 - Class Dependencies Designer). Vengono mostrate le dipendenze che intercorrono tra i moduli software dell'applicazione.

2.4. Implementazione del front-end grafico

All'avvio dell'applicazione viene visualizzata la *Main Form* (o form principale) nella quale sono stati predisposti menù e un elenco di opzioni che permettono la creazione dinamica di componenti grafici e la visualizzazione di form secondarie. (si veda Figura 2.2)

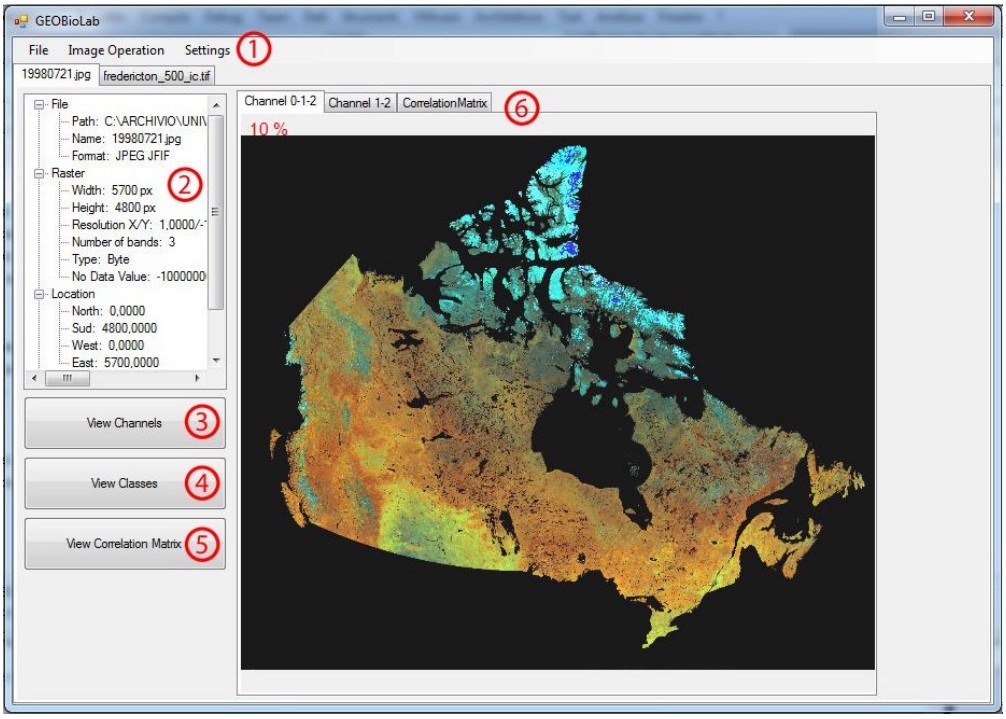


Figura 2.2 - Visualizzazione della *MainForm* (form principale).
(1) Menù principale con le funzionalità di File (Open, Load, Save di immagini e project), ImageOperation (ViewChannel, ViewClasses e ViewCorrelationMatrix), Settings (con le impostazioni del software).
(2) Visualizzazione delle informazioni estratte dall'immagine con la libreria GDAL.
(3) Tasto per la visualizzazione della *ViewChannelForm*.
(4) Tasto per la visualizzazione della *ViewClassesForm*.
(5) Tasto per la visualizzazione della *ViewCorrelationMatrix*.
(6) Schede dedicate alla visualizzazione dei canali dell'immagine scelti.

In particolare sono state implementate tre *Form Secondarie* (si veda Figura 2.3):

- *"ViewChannelsForm"*, dedicata alla visualizzazione dei canali dell'immagine.
- *"ViewClassesForm"*, adibita alla visualizzazione dei file delle classi. Nell'applicazione a Windows Form è stata gestita solamente la visualizzazione di file delle classi da abbinare a progetti di lavoro. Questi file delle classi saranno descritti in dettaglio nel Capitolo 3.
- *"ViewCorrelationMatrix"*, dedicata alla visualizzazione di informazioni statistiche, e alla loro rappresentazione (si vedano i Paragrafi 2.5.2.1, 2.5.2.2).

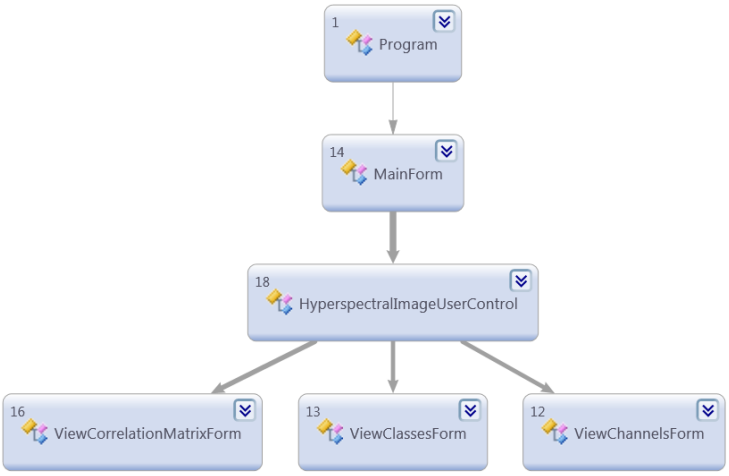


Figura 2.3 - Diagramma delle dipendenze tra *Main Form* e *Form Secondarie*.

Sfruttando gli *User Control* forniti dall'ambiente .Net, è stata predisposta un'interfaccia dinamica in grado di adattarsi alle richieste dell'utente. Infatti l'utente può decidere di aprire o chiudere le pagine dedicate alle varie sessioni di lavoro, può affiancare diversi progetti ad altre immagini iperspettrali, e può ridimensionare le finestre dell'applicazione in qualsiasi momento lo desideri. Questo meccanismo migliora il rendimento grafico, consentendo all'utente una maggior libertà di utilizzo e una buona flessibilità.

Gli User Control hanno anche un'importante funzione di estendibilità del codice, in quanto permettono al programmatore di riutilizzare gli stessi componenti grafici in form e interfacce differenti. Per questo è stata predisposta una gerarchia di Hyperspectral User Control (si veda Figura 2.4) che ereditano caratteristiche comuni:

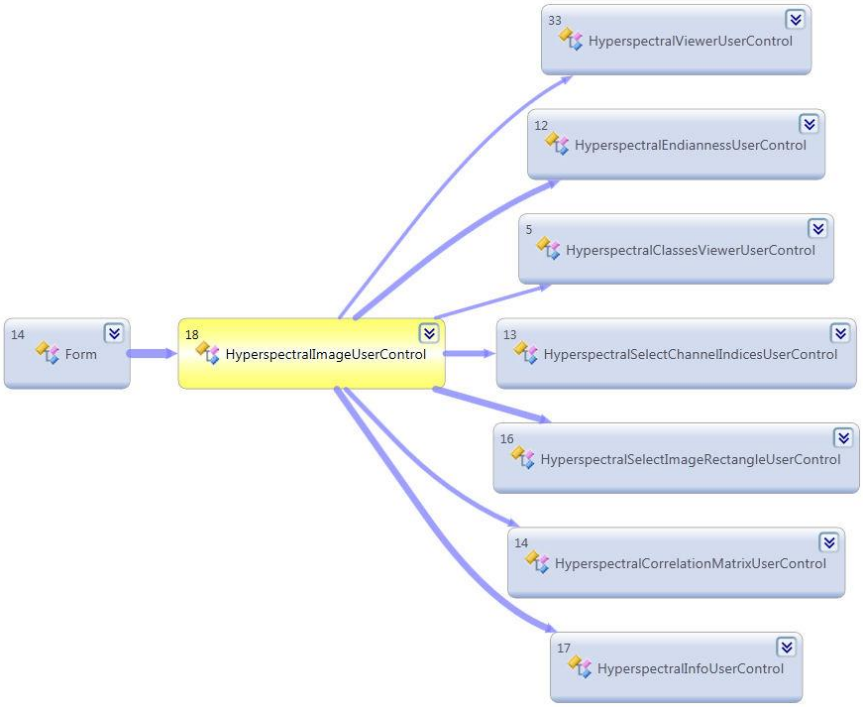


Figura 2.4 - Classificazione dei principali *User Control* implementati nell'applicazione.

2.4.1. Funzionalità

Per la visualizzazione delle immagini iperspettrali è stato sviluppato uno "*HyperspectralViewerUserControl"* ereditandolo dalla classe *ImageViewer* (della libreria esterna *BioLab*), la quale presenta metodi e proprietà utili per la gestione di immagini "normali". Queste funzionalità sono state estese per poter gestire e visualizzare immagini multispettrali e iperspettrali.

Date le dimensioni e il numero di bande delle immagini iperspettrali, si sono dovute affrontare problematiche legate alla gestione e al caricamento in memoria di grandi quantità di dati (normalmente non prese in considerazione quando si manipolano immagini "normali"). Per permettere una corretta visualizzazione, è stata implementata una finestra che consente la selezione di determinati parametri di visualizzazione (si veda Figura 2.5), quali il numero di canali da visualizzare o la selezione di una particolare area d'interesse.

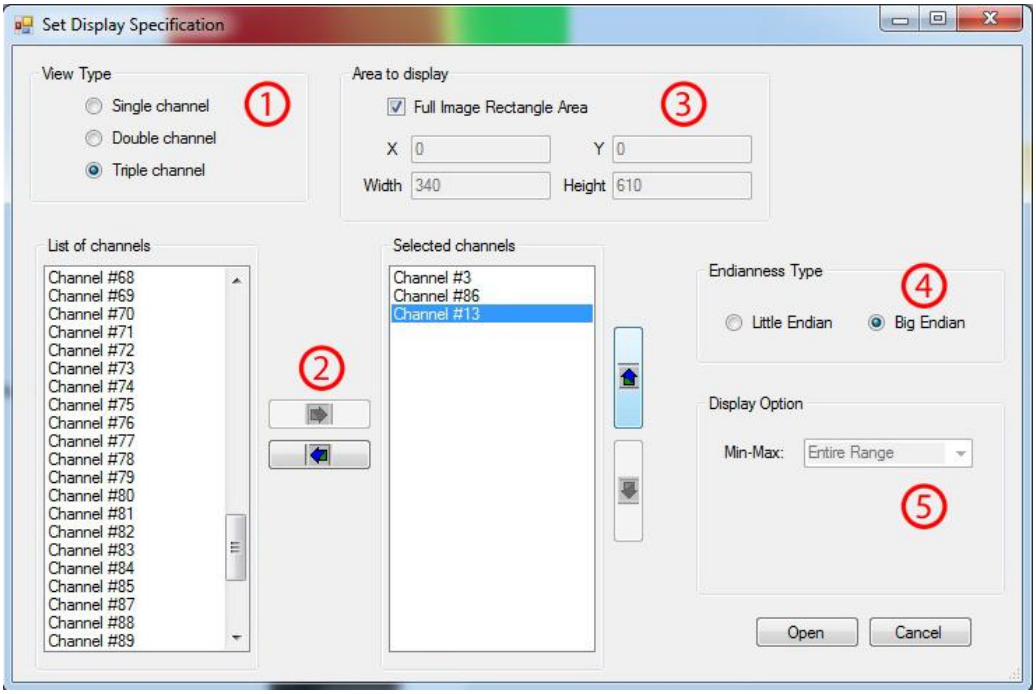


Figura 2.5 - Form di selezione dei parametri di caricamento delle immagini iperspettrali.
(1) Selezione della tipologia di visualizzazione da effettuare (1, 2 o 3 canali).
(2) Selezione dei canali da visualizzare.
(3) Selezione dell'area da visualizzare.
(4) Selezione dell'endianness (ordinamento di byte) utilizzata per la memorizzazione dell'immagine (se nota a priori).
(5) Selezione dell'intervallo di valori Min-Max per effettuare il mapping con le 255 tonalità di colore della visualizzazione in byte (per default, viene preso tutto il range dell'immagine *(Entire Range)*).

2.4.1.1. Sistema di Preview

Per ridurre il numero di risorse dedicate al caricamento delle immagini, e per evitare di incorrere in interruzioni forzate del programma, è stato predisposto un sistema di *"preview"* (o anteprime). Queste preview rappresentano l'immagine o la sotto-area selezionata, partendo da un livello di definizione basso, evitando così di caricare dati superflui se non richiesti. Se l'utente richiede la visualizzazione di un'area più piccola o se decide di applicare lo zoom su quest'area, verrà richiesto un nuovo livello di preview con un dettaglio maggiore. Questo procedimento viene ripetuto fino ad arrivare al massimo livello di dettaglio, che corrisponde al caricamento dell'immagine nelle sue dimensioni reali (si veda Figura 2.6).

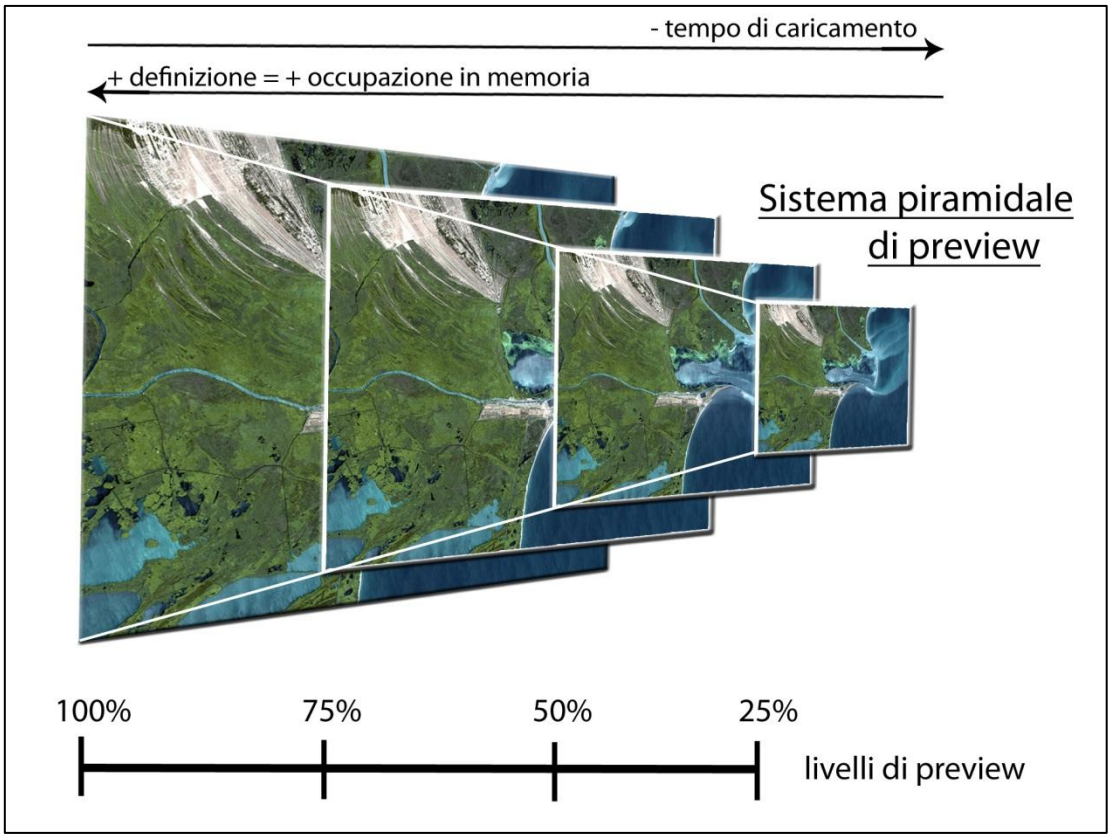


Figura 2.6 - Sistema piramidale di preview.
Mostra il variare di occupazione in memoria e tempo di caricamento, al variare della definizione dei livelli di preview.

I vari livelli di preview visualizzati dall'utente vengono mantenuti in memoria per poter essere recuperati velocemente nel caso sia necessario ripresentare lo stesso livello di dettaglio. (Questo permette di ridurre i tempi di caricamento, e di fornire all'utente una visualizzazione veloce e fluida, indipendentemente dalle dimensioni dell'immagine).

Per ottimizzare le prestazioni ed i tempi di risposta dell'applicazione, si è delegato il compito di caricamento delle immagini iperspettrali ad un "*BackgroundWorker"*: il BackgroundWorker è una classe dell'ambiente .Net e permette l'implementazione di thread secondari, che siano in grado di elaborare processi onerosi, in modo da lasciare libero il thread dedicato all'interfaccia utente.

Questa soluzione implementativa permette di non bloccare l'interfaccia utente durante le fasi di caricamento e di lasciare la possibilità all'utente di svolgere parallelamente attività differenti.

2.4.1.2. Funzioni di Zooming

All'interno del visualizzatore sono state modificate le funzionalità di *zooming*, in particolare le funzioni di "*Zoom In"*, "*Zoom Out"*, e di "*Fit To Window"* già presenti nello UserControl ImageViewer da cui deriva.

Per re-implementare le prime due funzioni di Zoom, ci si è ricondotti all'evento di *MouseWheel*, ovvero all'evento che gestisce lo scorrimento della rotella del mouse, e si è implementato un metodo che richiama la visualizzazione dell'area interessata dell'immagine, opportunamente riscalata al fattore di zoom richiesto (si veda Figura 2.7).

Per la funzione di "*Fit To Window"* è stato implementato un metodo che calcola il massimo fattore di zoom, sfruttando tutta l'area di visualizzazione disponibile, e restituisce la giusta preview al visualizzatore.

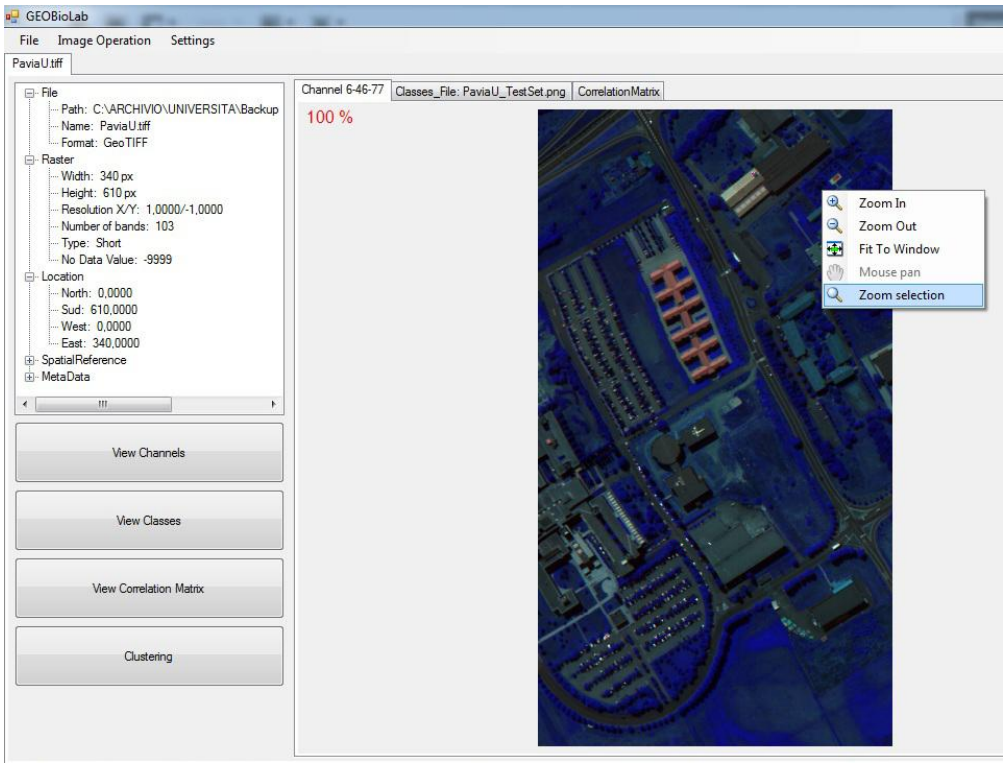
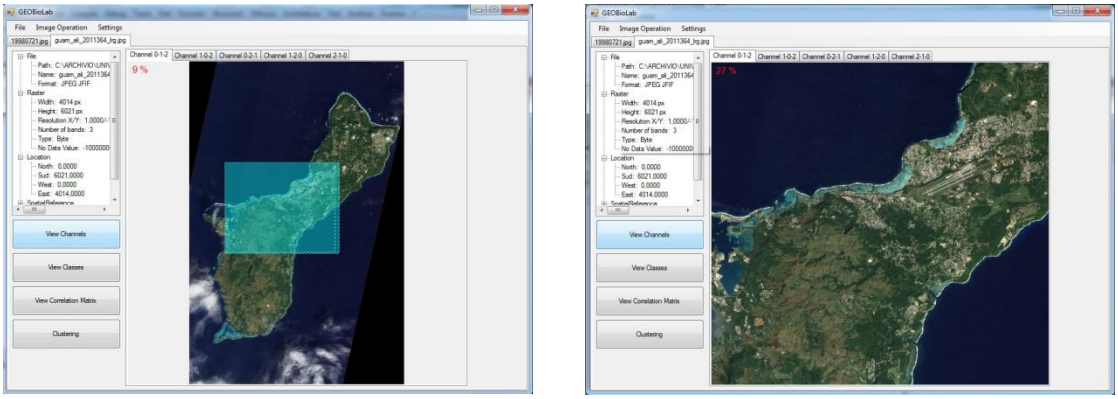


Figura 2.7 - Interfaccia dell'applicazione nella pagina di visualizzazione dei canali. Posizionandosi con il cursore sull'immagine e cliccando con il pulsante destro, si ottiene un menù a tendina con le funzioni di Zooming e di Subregion Selection.

2.4.1.3. Subregion Selection

Per permettere la selezione di una sotto-area di visualizzazione dell'immagine, è stata implementata una funzione che consente all'utente di tracciare graficamente un'area di selezione sul visualizzatore (si veda Figura 2.8). In particolare sono stati gestiti gli eventi del mouse di "click" e "rilascio" del tasto sinistro, in modo da memorizzare le coordinate spaziali del puntatore e poterne fare un mapping con le coordinate spaziali relative all'immagine.



(a) (b)

Figura 2.8 - (a) Selezione della regione di visualizzazione su un'immagine in preview ad un livello di zoom del 9%.
- (b) Visualizzazione della regione selezionata nella figura (a). Livello di zoom del 27%.

2.5. Progettazione logica

Per gestire le diverse immagini iperspettrali è stata predisposta una gerarchia di classi (si veda Figura 2.9), che sfrutta i principi dell' *ereditarietà tra classi* per implementare concetti che partono da un livello astratto molto elevato (superclassi) fino ad arrivare a concetti altamente specializzati (sottoclassi).

Il concetto di *Ereditarietà* [11] viene spiegato come:

> *"... Il meccanismo dell'ereditarietà permette di derivare nuove classi a partire da quelle già definite. Una classe derivata attraverso l'ereditarietà, o sottoclasse, mantiene i metodi e gli attributi delle classi da cui deriva (classi base, o superclassi); inoltre, può definire i propri metodi o attributi, e può ridefinire il*

codice eseguibile di alcuni dei metodi ereditati tramite un meccanismo chiamato overriding. …"

In particolare le sottoclassi appartenenti alla gerarchia dovranno specializzare i metodi relativi alla gestione e al caricamento di immagini tipizzate. Per rappresentare i diversi tipi di dato utilizzati nella memorizzazione di immagini iperspettrali (interi, byte, short e double), è stato creato un tipo di dato *enumerativo* denominato *"Hyperspectral-DataType"* in grado di rappresentarli.

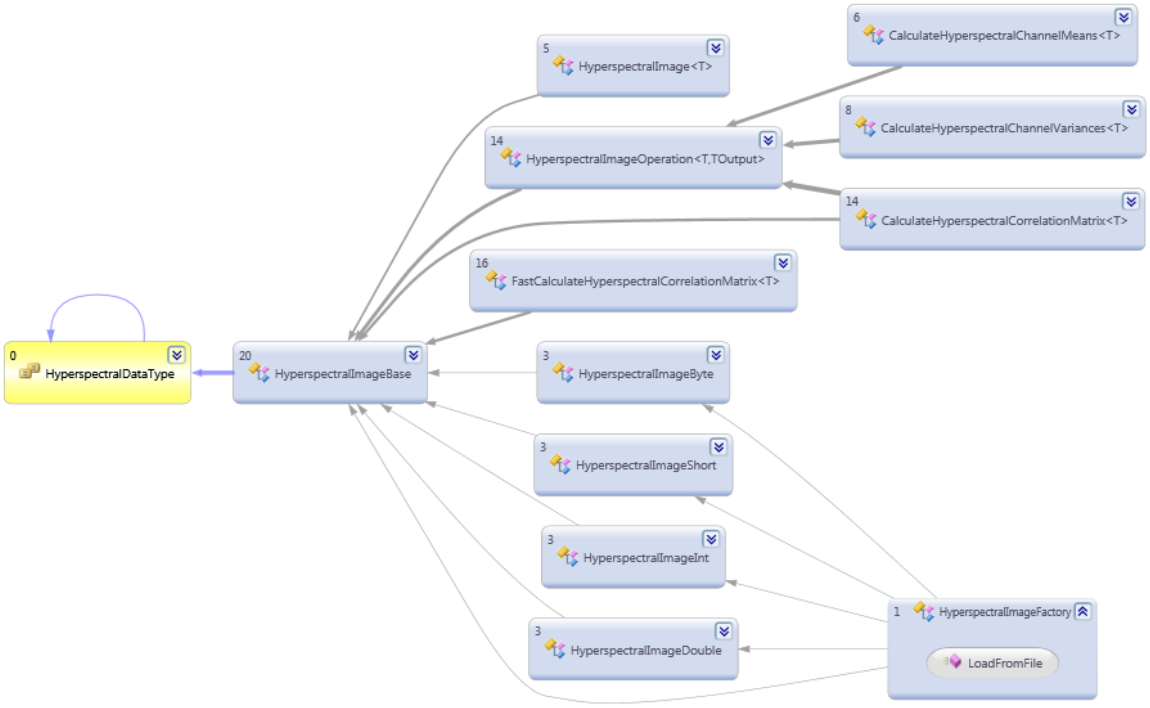


Figura 2.9 - Tipizzazione delle classi di immagini iperspettrali.

A livello implementativo è nata l'esigenza di creare una classe statica denominata *"HyperspectralImageFactory"*, che segue uno dei design pattern fondamentali della programmazione a oggetti, il *Factory Method*.

Il Factory Method predispone una classe "stampo" (o creatrice) che fornisce un'interfaccia pubblica per la creazione di un oggetto, e lascia che le sottoclassi decidano quale oggetto istanziare. In questo modo viene permessa l'istanziazione di un oggetto, senza sapere a priori la sua tipologia di appartenenza, o più in generale la sua classe di appartenenza.

2.5.1. Gerarchia di classi Hyperspectral

Per implementare la gerarchia fin qui introdotta, si è definita una classe astratta chiamata *"HyperspectralImageBase"*, la quale eredita dalla classe astratta (ImageBase) della libreria Biolab. Quest'ultima fornisce metodi e proprietà utili nella gestione generica di immagini (si veda la Tabella 2.2).

La classe che eredita dalla radice della gerarchia è la classe *"HyperspectralImage<T>"*, che sovrascrive i metodi astratti ereditati, e definisce nuovi metodi virtuali in modo che le classi figlie, sfruttando i principi del *polimorfismo*, ridefiniscano questi metodi caratterizzandoli con delle operazioni tipizzate.

Il termine *Polimorfismo* [12] viene definito come:

> *"... In informatica, il termine polimorfismo (dal greco "avere molte forme")... nel contesto della programmazione orientata agli oggetti, si riferisce al fatto che una espressione il cui tipo sia descritto da una classe A può assumere valori di un qualunque tipo descritto da una classe B sottoclasse di A (polimorfismo per inclusione)..."*

Nelle foglie della gerarchia si hanno le quattro classi figlie tipizzate, ovvero: *"HyperspectralImageByte"*, *"HyperspectralImageShort"*, *"HyperspectralImageInt"* e *"HyperspectralImageDouble"*.

Queste classi vengono istanziate dalla classe Factory e sovrascrivono i metodi virtuali ereditati, implementandone la loro versione tipizzata.

Nella seguente tabella è possibile mostrare i metodi e le proprietà che caratterizzano le classi portanti di questa gerarchia (Tabella 2.2):

	ImageBase	**HyperspectralImageBase**	**HyperspectralImage<T>**
Proprietà	*Height* *Pixel Count* *Width*	*HyperspectralDataType* *ChannelCount* *Format* *XResolution* *YResolution* *Coord* *SpatialReference* *MetaData* *NoDataValue*	
Metodi	GetSubImage() LoadFromFile() SaveToFile() SetSubImage()	PixelSizeOf() *abstract* GetChannel() *abstract* GetChannels()	*override* GetChannel() *override* GetChannels() *virtual* ReadBuffer() *virtual* Resize()

Tabella 2.2 - Rappresentazione tabellare delle classi principali della gerarchia iperspettrale.
(Nei *prototipi* dei metodi citati, non sono stati riportati i parametri per ragioni di spazio).

2.5.2. Elaborazione di informazioni statistiche

All'interno della stessa gerarchia iperspettrale, per risolvere il problema delle operazioni tipizzate sulle immagini viene applicata la stessa tecnica vista nel paragrafo precedente.

È stata creata una classe astratta *"HyperspectralImageOperation<T,TOutput>"* che sfrutta i *generics* (ovvero tipi di dati generici) per fornire un'interfaccia astratta alle sue sottoclassi. Le classi che ereditano da questa superclasse servono ad implementare metodi tipizzati per l'estrazione di informazioni statistiche dalle bande delle immagini prese in considerazione.

In particolare c'è un secondo livello di classi astratte (si veda Figura 2.10(b)):

- *"CalculateHyperspectralChannelMeans<T>"*
- *"CalculateHyperspectralChannelVariances<T>"*
- *"CalculateHyperspectralCorrelationMatrix<T>"*

Queste classi provvedono rispettivamente al calcolo delle *medie*, delle *varianze* e della *matrice di correlazione* tra bande, fornendo un'interfaccia a tutte le classi figlie, che implementeranno i metodi specifici dei calcoli richiesti.

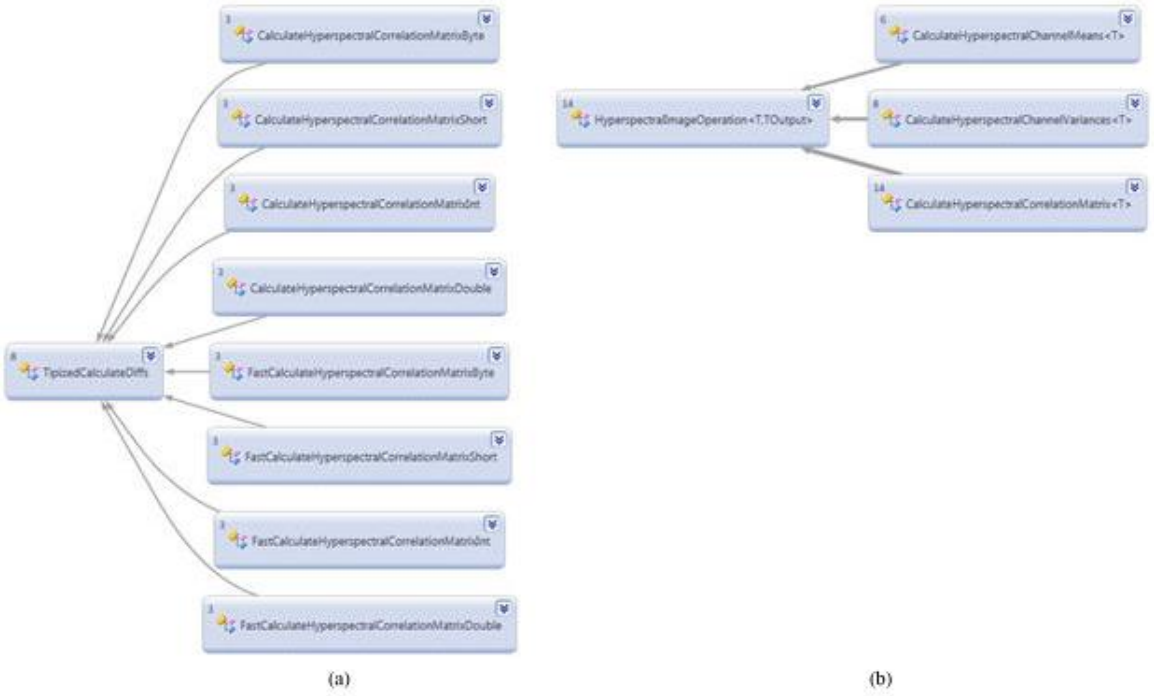


Figura 2.10 - Classi per l'estrazione di informazioni statistiche.
(Per motivi di spazio non sono stati riportati tutti i livelli di astrazione delle gerarchie).

Se si aumenta il livello di dettaglio, e si scende nella gerarchia delle classi iperspettrali, è possibile trovare altri esempi di classi tipizzate. Una di queste è rappresentata in Figura 2.10(a) e descrive l'insieme delle classi specifiche per il calcolo della matrice di correlazione, e la loro interazione con una "classe contenitore" denominata *"TipizedCalculatedDiffs"*. Tutte le classi che calcolano la matrice di correlazione necessitano di interagire con questa classe contenitore, perché richiedono il calcolo oneroso di parecchie differenze tra valori tipizzati. All'interno della classe *"TipizedCalculatedDiffs"* è presente un insieme di classi "operazionali", adibite al solo calcolo tipizzato delle differenze tra i singoli pixel dei canali dell'immagine e il valor medio dei canali stessi. Ogni sottoclasse restituisce dei valori numerici utilizzati nel calcolo degli scarti quadratici medi, all'interno della formula per il calcolo della varianza.

Nei successivi due sotto-paragrafi verranno descritti in dettaglio la *matrice di correlazione tra bande*, e il *grafico delle medie*.

2.5.2.1. La matrice di correlazione tra bande

La *Correlation Matrix* (o matrice di correlazione) viene utilizzata per rappresentare i valori dei coefficienti di correlazione presenti tra ogni singola coppia di canali di un'immagine (si veda Figura 2.11). [13]

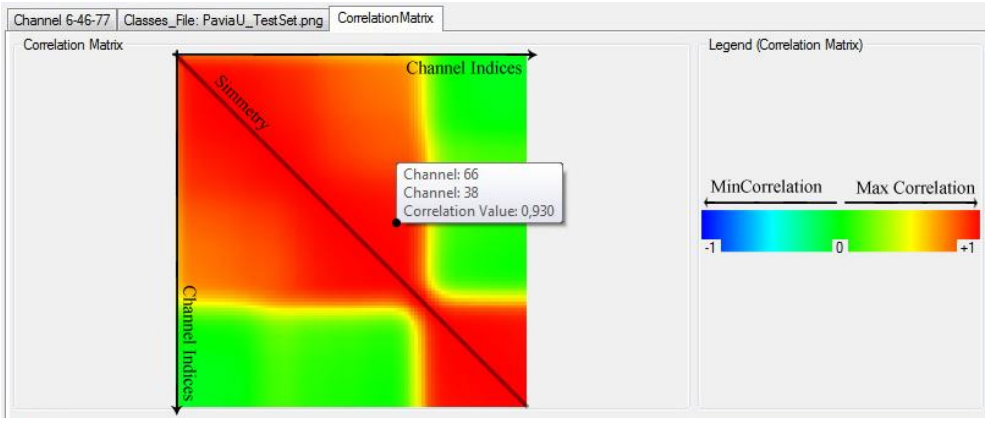


Figura 2.11 - Screenshot dell'applicazione durante la visualizzazione della matrice di correlazione. Nella parte destra della form è possibile notare la legenda dei colori attributi ai valori di correlazione delle bande.

Ogni coefficiente, anche denominato *coefficiente di Pearson*, viene calcolato dividendo la *covarianza*, per il prodotto delle *deviazioni standard* dei due canali coinvolti nei calcoli. I coefficienti di correlazione sono valori compresi tra -1 (indica che i due canali sono *inversamenti correlati*), e +1 (i due canali sono *correlati positivamente o direttamente correlati*). Il valore 0 indica che non c'è correlazione tra i canali coinvolti (si veda la Tabella 2.3).

Canali	1	2	3	4	5	6	7	8	9	10	11	12
1	1.0											
2	0.66	1.0										
3	0.75	0.69	1.0									
4	0.64	0.75	0.69	1.0								
5	0.68	0.70	0.79	0.71	1.0							
6	0.31	0.45	0.45	0.52	0.63	1.0						
7	0.41	0.53	0.57	0.61	0.60	0.71	1.0					
8	0.61	0.75	0.71	0.80	0.82	0.67	0.67	1.0				
9	0.58	0.75	0.67	0.81	0.73	0.59	0.66	0.89	1.0			
10	0.34	0.50	0.52	0.57	0.64	0.79	0.74	0.72	0.67	1.0		
11	-0.45	-0.50	-0.38	-0.51	-0.32	0.07	-0.11	-0.47	-0.59	-0.02	1.0	
12	-0.58	-0.58	-0.56	-0.57	-0.48	-0.01	-0.20	-0.52	-0.59	-0.10	0.75	1.0

Tabella 2.3 - Esempio di una matrice di correlazione di un'immagine a 12 dimensioni.

Per il calcolo della matrice di correlazione sono state predisposte due classi astratte differenti:

- La classe *"CalculateHyperspectralCorrelationMatrix<T>"* calcola il coefficiente di correlazione tra ogni coppia di canali selezionati, e per ognuna di esse ricalcola gli scarti tra i singoli pixel di entrambi i canali e il valor medio dei canali stessi.
- La classe *"FastCalculateHyperspectralCorrelationMatrix<T>"* calcola il coefficiente di correlazione tra i canali, mantenendo in memoria tutti gli scarti di ogni pixel ed il valor medio di ogni canale.

La seconda classe, appositamente denominata "Fast", è nettamente più performante della prima, in quanto memorizza gli scarti dei pixel dal valor medio, ogni qualvolta incontra un nuovo canale, ed evita di ripetere i calcoli ad ogni "accoppiamento" dello stesso canale. La prima classe, a differenza della seconda, riduce notevolmente lo spazio occupato in memoria, in quanto non deve memorizzare tutte le informazioni relative ai calcoli precedenti.

E' stato necessario predisporre entrambe le soluzioni, perché in molti dei casi affrontati, la dimensione delle immagini prese in considerazione non permetteva di utilizzare la versione "Fast" del calcolo, causa l'eccessiva quantità di memoria centrale richiesta.

Per evitare di incorrere in eventuali errori software di questo tipo, viene effettuato un controllo prima che venga richiamato il calcolo della matrice di correlazione. Questo controllo verifica la possibilità di utilizzare l'una o l'altra classe di calcolo, attraverso una stima di occupazione della matrice di correlazione che si sta andando a calcolare.

In entrambi i casi, il calcolo della matrice di correlazione, data l'occupazione in memoria e la quantità di calcoli richiesti, viene demandata ad un *BackgroundWorker* appositamente programmato (con la stessa tecnica vista nel Paragrafo 2.4.1).

2.5.2.2. Il grafico delle medie

Il *Mean Graph* (o grafico delle medie) viene utilizzato per rappresentare graficamente una stima dell'intervallo di appartenenza dei pixel di un'immagine.

In particolare per ogni canale viene calcolato il valor medio, e gli estremi dell'intervallo di appartenenza vengono ricavati aggiungendo e sottraendo la *deviazione standard* a tale valore.

È stato predisposto un visualizzatore in grado di esprimere l'intervallo di appartenenza dei pixel, di ogni canale selezionato, con una sfumatura di colore (si veda Figura 2.12). Inoltre per migliorare l'interpretazione del grafico, e poter tracciare delle linee continue, è stata applicata un'interpolazione grafica dei dati.

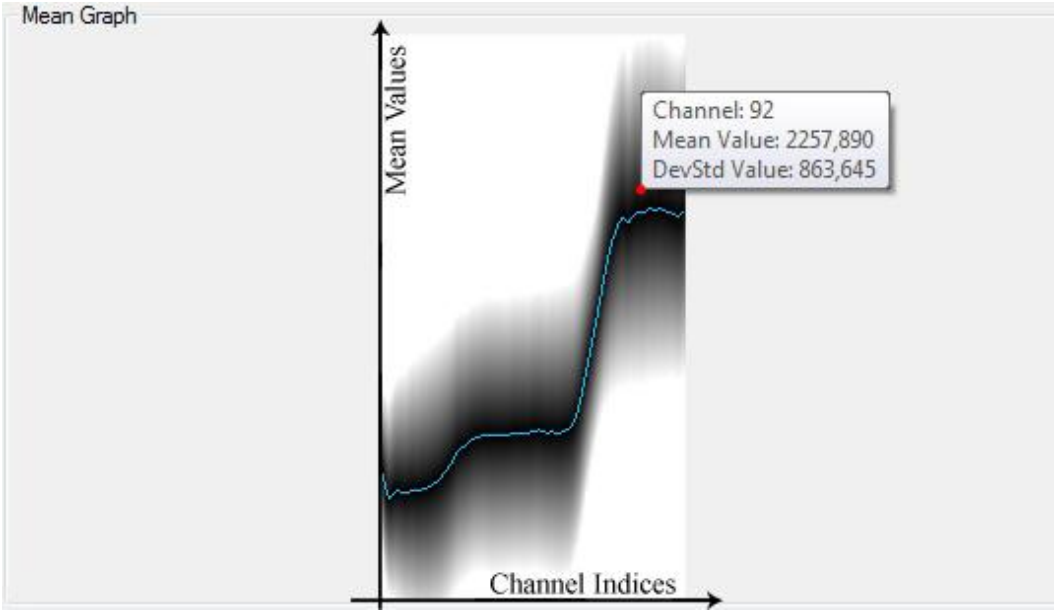


Figura 2.12 - Screenshot dell'applicazione durante la visualizzazione del grafico delle medie. Posizionando il cursore su un punto qualsiasi del grafico, appare un popup contenente l'indice del canale selezionato, il valor medio e la deviazione standard.

2.6. Gestione dei file di progetto

Per dare la possibilità all'utente di memorizzare i dati ottenuti da operazioni svolte sulle immagini iperspettrali, e per poterli riutilizzare in esecuzioni successive, si è deciso di implementare una classe contenitore, che memorizza al suo interno tutte queste informazioni.

La classe *"HyperspectralImageProject"* possiede una struttura complessa (si veda Figura 2.13), in quanto deve rendere fruibili dall'esterno molti metodi utili alla richiesta di operazioni di calcolo o alla modifica di file di progetto, e deve mantenerne privati

altrettanti che svolgono operazioni di scrittura e lettura da file, che effettuano inizializzazione di valori e percorsi, o ricercano elementi precisi all'interno delle cartelle salvate.

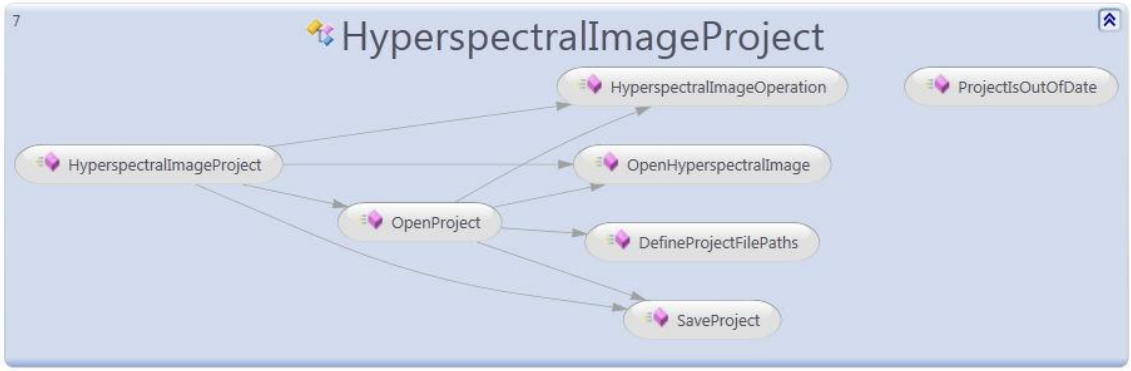


Figura 2.13 - Diagramma semplificato della classe HyperspectralImageProject.
(Sono state rappresentate solamente le dipendenze tra i metodi principali della classe).

In particolare i *Metodi Pubblici* sono quelli necessari al calcolo di informazioni statistiche. Di ognuno di questi metodi sono stati creati diversi *prototipi*, in modo da permettere il calcolo su tutta l'immagine o solamente su alcune sotto-aree, la selezione di una sotto-lista dei canali dell'immagine o la scelta precisa di una singola classe di pixel. Oltre a questi metodi di calcolo, vengono anche forniti metodi relativi al salvataggio e al caricamento di file di progetto, e all'eventuale gestione di file delle classi.

Nei *Metodi Privati* invece viene gestita tutta la logica interna della classe, quindi vengono predisposte e istanziate le giuste strutture dati, atte a contenere le informazioni prelevate dalle immagini o caricate da file. Inoltre vengono fisicamente implementati i metodi di lettura e scrittura su file, distinguendo i file contenenti valori non direttamente leggibili (memorizzati in binario), ed i file disponibili alla consultazione esterna (memorizzati su file di testo).

2.6.1. Gerarchia di cartelle per il salvataggio di file

Per effettuare il salvataggio su disco dei file di progetto, è stata predisposta una gerarchia di directory. Nell'*header file* principale (ovvero il file con estensione *.prj*) vengono inserite delle linee d'intestazione contenenti i percorsi dei file dell'immagine

iperspettrale, e dell'immagine contenente le informazioni sulle classi di appartenenza di ogni pixel (se presente).

Nella cartella con lo stesso nome del file di progetto, sono racchiusi i file (*.dat) contenenti: medie e varianze dei singoli canali, matrice di correlazione (se calcolata su tutti i canali dell'immagine), e il file contenente il mapping tra le etichette e i colori del file delle classi.

Nel caso in cui sia stato caricato un file delle classi, e l'utente abbia svolto attività statistiche su alcune di queste classi, viene predisposta una sotto-gerarchia di cartelle (strutturata come quella dei file di progetto). In questo modo le informazioni statistiche relative ad un precisa classe di pixel, vengono memorizzate in una cartella identificata da un nome univoco rappresentante la classe stessa.

Il *naming* (ovvero la convenzione con la quale si stabiliscono nomi e percorsi di file e cartelle) viene gestito automaticamente e controlla l'univocità dei singoli path, per evitare sovrascritture e problemi di memorizzazione.

Capitolo 3
Classificazione non supervisionata

3.1. Introduzione alla Visione Artificiale

La *Visione Artificiale (nel seguito VA)* è una disciplina informatica che studia le tecniche di acquisizione, analisi, elaborazione e comprensione delle immagini.

Lo *scopo* della *VA* è quello di progettare sistemi artificiali in grado di imitare sistemi visivi e percettivi biologici, presenti tipicamente nell'uomo e in alcune specie animali.

In particolare la *VA* studia tecniche di riconoscimento automatico, o semi-automatico, delle immagini, sfruttando algoritmi che estraggono le informazioni utili dalle immagini stesse.

Esistono diversi campi d'applicazione della Visione Artificiale:

- Processi di controllo (catene di montaggio e macchine industriali).
- Navigazione (veicoli auto-pilotati o tele-guidati).
- Telesorveglianza (controllo di sagome, forme o persone)
- Localizzazione di oggetti (identificazione di forme e colori).
- Tracciamento (o *tracking*) di oggetti ed individuazione di movimenti.
- Ricostruzione 3D di forme e oggetti.

- Riconoscimento (o *Recognition*) di oggetti e contesti.

Uno dei problemi più comuni nel campo della *VA*, è proprio quello del ***Riconoscimento***.

Quest'ultimo può essere scomposto in tanti sotto-problemi limitati a contesti e oggetti ben precisi:

- *Object Recognition*, riconoscimento di oggetti posizionati nello spazio.
- *Content-based Image Retrieval*, riconoscimento di immagini con un contenuto comune.
- *Pose Estimation*, stima della posizione e dell'orientazione di un oggetto nello spazio, rispetto alla camera di rilevamento.
- *Optical Character Recognition*, identificazione di caratteri in testi stampati o scritti manualmente.
- *Byometrics Recognition (facial, fingerprint, iris, human-pose, ecc…)*, tecniche di riconoscimento biometrico per l'identificazione di individui.

3.1.1. Pattern Recognition

Il *Pattern Recognition (PR)* è un particolare problema di apprendimento automatico, e consiste nell'identificazione e classificazione di *pattern* all'interno di dati grezzi.

Il termine *"**pattern**"* viene spesso erroneamente tradotto con il termine "forma", quando invece possiede un significato molto più ampio.

> *Watanabe definisce un pattern come "l'opposto del caos" e come un'entità vagamente definita alla quale può essere dato un nome. [14]*

Nel contesto del *PR* il termine "Recognition" assume il valore di *Classificazione o Categorizzazione*. Lo scopo del *PR* è infatti quello di assegnare delle etichette a delle *classi* di dati multidimensionali, sfruttando i pattern come *classificatori*.

Con il termine "***classe**"* si intende un insieme di entità aventi proprietà comuni. Nell'ambito del riconoscimento di immagini digitali, una classe può essere considerata un insieme di pixel (o punti) aventi caratteristiche simili.

Gli algoritmi di *PR* possono apprendere le classi basandosi su conoscenze pregresse fornitegli dal progettista (apprendimento supervisionato) o su informazioni estratte direttamente dai dati grezzi (apprendimento non supervisionato).

Esistono numerosi algoritmi di Pattern Recognition che si differenziano per la procedura di apprendimento utilizzata, e per la tipologia di output restituito. Tra questi algoritmi se ne possono identificare alcuni non supervisionati e mirati alla categorizzazione di classi, denominati algoritmi di ***Clustering***.

3.2. Teoria sul Clustering

Con il termine *"Clustering" (*che letteralmente significa *"addensamento"* o *"raggruppamento")* si denota una famiglia di tecniche non supervisionate, in grado di individuare raggruppamenti intrinseci (detti *cluster*) tra i dati nello spazio multidimensionale, e di etichettare le classi in corrispondenza di tali raggruppamenti [15].

Un ***cluster*** è formato da un insieme di oggetti che sono *"simili"* tra loro, e sono *"dissimili"* da tutti gli oggetti appartenenti ad altri cluster (si veda Figura 3.1).

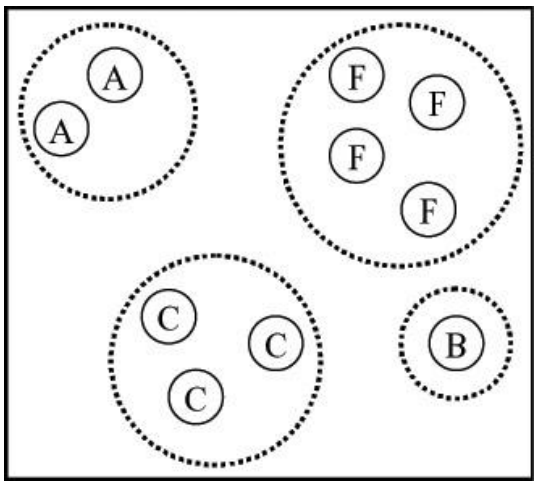


Figura 3.1 – Rappresentazione esemplificativa di Clustering, su gruppi di elementi simili.

Nella maggior parte dei casi, i *criteri di clustering* vengono definiti sulla base delle due seguenti osservazioni:

1) coppie di oggetti presi da uno stesso cluster devono sempre essere più simili, rispetto a coppie prelevate da cluster differenti.
2) i cluster sono costituiti per loro natura da "nuvole" di punti a densità relativamente elevata, separati da zone con densità inferiore.

Spesso nel definire il concetto di clustering si tende a considerare indistintamente il termine "simile" con il termine "vicino". In realtà esistono modelli di clustering che prevedono diversi *criteri di similarità*, dove il concetto di simile spesso non si limita alla semplice vicinanza in uno spazio multidimensionale. Per esempio modelli basati sulla distribuzione dei dati (*Distribution-based Models*), o sulla densità dei dati (*Density-based Models*) considerano un concetto di similarità completamente differente da quello euclideo.

3.2.1. Dicotomia delle tecniche di Clustering

In letteratura esistono diverse tecniche di clustering, che vengono suddivise secondo criteri differenti. Una delle principali categorizzazioni degli algoritmi di clustering è basata sulla scelta del modello di suddivisione dello spazio dei cluster.

- *Clustering Gerarchico (Hierarchical Clustering)*
  Gli algoritmi di clustering appartenenti a questa categoria organizzano i dati in sequenze innestate di gruppi, le quali vanno a formare una gerarchia di dati. Gli strumenti grafici più utilizzati per la rappresentazione di questa tecnica sono alberi e dendrogrammi (si veda Figura 3.2).

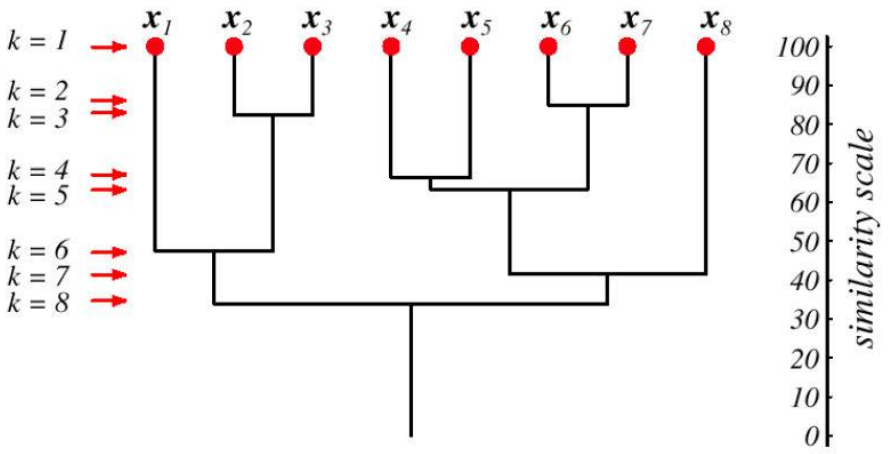


Figura 3.2 - Esempio di *dendrogramma* per la rappresentazione di Clustering Gerarchico. Sull'asse delle ascisse si trova la distanza logica tra i cluster, mentre su quello delle ordinate è presente il livello gerarchico di aggregazione (con la relativa similarità).

Le fasi intermedie del processo di clustering gerarchico, ovvero le fasi che portano dallo stato iniziale (dati grezzi, non raggruppati) allo stato finale (dati suddivisi in cluster), possono essere caratterizzate da due "filosofie di raggruppamento" diverse.

- *Metodo aggregativo (o Bottom-Up).*

  Questo metodo prevede che inizialmente tutti gli elementi siano considerati cluster indipendenti, e che procedendo nelle fasi dell'algoritmo, i cluster più vicini si uniscano. Gli algoritmi *Bottom-Up* procedono fino al raggiungimento del numero di cluster prefissato, o fino a che la "distanza minima" tra i cluster non supera un determinato valore di soglia.

- *Metodo divisivo (Top-Down)*

  L'algoritmo prevede che allo stato iniziale ogni elemento appartenga a un unico cluster, e procedendo nell'esecuzione dell'algoritmo, i cluster vengano divisi in gruppi di elementi sempre più piccoli e più omogenei. Gli algoritmi *Top-Down* necessitano di conoscere un numero di cluster prefissato, per poterlo utilizzare come condizione di arresto.

- *Clustering Partizionale (Partitional Clustering)*

  I metodi appartenenti a questa categoria cercano di suddividere gli elementi del *dataset* in un insieme di cluster disgiunti (si veda Figura 3.3). Il partizionamento dei dati avviene per mezzo di algoritmi euristici iterativi, che cercano di minimizzare i criteri di dissimilarità dei dati.

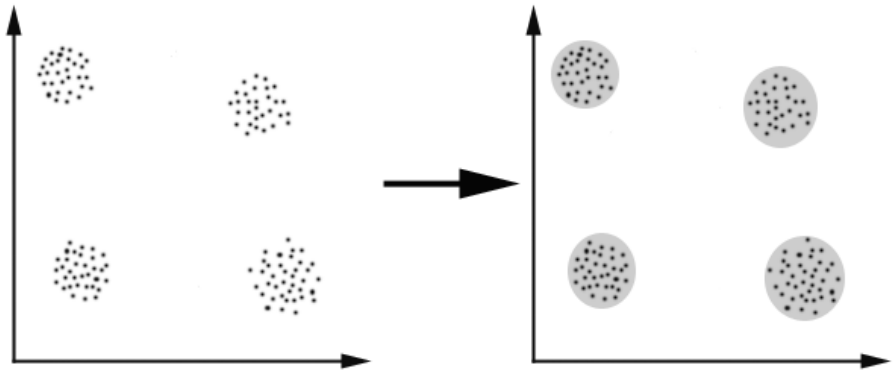


Figura 3.3 - Esempio di *Partitional Clustering*, su 4 classi facilmente distinguibili.

Un ottimo esempio di algoritmo partizionale è l'algoritmo ***K-Means*** (che verrà descritto in dettaglio nel Paragrafo 3.3).

Un'altra categorizzazione importante delle tecniche di clustering dipende dalla possibilità che un elemento possa essere o meno assegnato a cluster differenti:

- *Clustering Deterministico (o esclusivo)*

  Si parla di clustering esclusivo o ***Hard Clustering***, quando ogni elemento può appartenere a uno e un solo cluster. (Con questa soluzione i cluster non possono avere elementi in comune).
- *Clustering Non-deterministico (o non-esclusivo)*

  Si parla di clustering non-esclusivo o ***Soft Clustering***, quando ogni elemento può appartenere a più cluster differenti, con gradi di appartenenza diversi.

  (Da questa tipologia di algoritmi di clustering nasce la *logica fuzzy*, che verrà illustrata in dettaglio nel Paragrafo 3.4).

3.3. Algoritmo K-Means

L'algoritmo *K-Means (KM)* [16] appartiene alla famiglia degli algoritmi partizionali esclusivi (o di ***Hard Clustering)***. L'obiettivo di questo algoritmo è quello di effettuare una classificazione non supervisionata, minimizzando le varianze intra-cluster.

Questo algoritmo necessita di alcune definizioni preliminari:

- sia $X = \{X_1, X_2, .., X_N\}$, $dove\ X_k \epsilon \Re^p,\ k = 1, .., N$, l'insieme degli N oggetti (p-dimensionali) su cui effettuare il clustering;
- sia $V = \{V_1, V_2, \dots, V_C\}$ il vettore dei *C centroidi* p-dimensionali, dove il termine "*centroide"* rappresenta il punto medio o punto centrale di un cluster;
- sia *U* la *matrice di membership (o matrice di appartenenza)*, di dimensioni CxN, nella quale viene rappresentata l'appartenenza dei punti ai diversi cluster. Nel caso specifico, trattandosi di un algoritmo esclusivo, i valori presenti nella matrice saranno valori binari (0 o 1);
- si definisce "*partizione hard"*, una famiglia di cluster $\mathcal{P} = \{A_1, A_2, \dots, A_C\}$ tali che ogni oggetto può appartenere ad uno e un solo cluster;
- si definisce *distanza euclidea multidimensionale*, la distanza tra due punti nello spazio multidimensionale.

3.3.1. Pseudocodifica (KM)

L'algoritmo richiede in input i dati su cui effettuare il clustering *(X)*, il numero di cluster *(C)* e una soluzione iniziale (*P)* che può anche essere ricavata a partire da *C* centroidi (generati casualmente o per mezzo di euristiche).

Ad ogni iterazione effettua il calcolo di nuovi "prototipi" di centroidi, ottenuti come media tra gli elementi p-dimensionali appartenenti ai singoli cluster, e aggiorna la soluzione attuale andando a riassegnare i punti ai cluster rappresentati dal centroide più vicino.

L'algoritmo termina al raggiungimento di almeno una delle seguenti condizioni di *convergenza*:

I. la procedura di aggiornamento non modifica la matrice di appartenenza;
II. il superamento del numero massimo di iterazioni (se prefissato);

Pseudocodice (K-Means)

```
(Input: X, N, V)(*)
Inizializza X, N, V, C, P;
Ripeti{
      Aggiorna centroidi();
      Aggiorna soluzione();
} finchè (U viene modificata && n_iterazioni < max_iterazioni)
(Output: P)
```

(*) È possibile specificare in input il numero massimo di iterazioni.

È possibile definire lo stesso algoritmo come un problema di ottimizzazione, definendo la funzione obiettivo:

$$\boldsymbol{J_{KM}(U,V)} = \sum_{i=1}^{C}\sum_{k=1}^{N} u_{ik} \cdot \|X_k - V_i\|^2 \qquad (1)$$

$$dove\ \|X_k - V_i\|\ \text{è}\ la\ norma\ euclidea.$$

La partizione ottima è quella U^* che riesce a minimizzare la funzione obiettivo.

3.3.2. Limiti dell'Hard Clustering

In termini di *tempi*, l'algoritmo riesce sempre ad ottenere un risultato in un tempo medio accettabile (sempre inferiore agli altri algoritmi appartenenti alla stessa famiglia).

In termini di *qualità* delle soluzioni, l'algoritmo non garantisce il raggiungimento dell'ottimo globale. L'ottimizzazione è iterativa e locale, perciò il metodo può solo convergere verso massimi locali della soluzione. Per ottenere risultati qualitativamente migliori, è possibile lanciare l'algoritmo più volte, facendolo partire da situazioni iniziali differenti, generando diversi insiemi di centroidi casuali.

L'algoritmo presenta inoltre diverse limitazioni legate alla conformazione dei cluster da identificare:

- Se i cluster possiedono *dimensioni differenti* (in termini di quantità di punti da classificare), l'algoritmo può non effettuare una suddivisione corretta, andando a scomporre le classi di grandezza maggiore (si veda Figura 3.4).

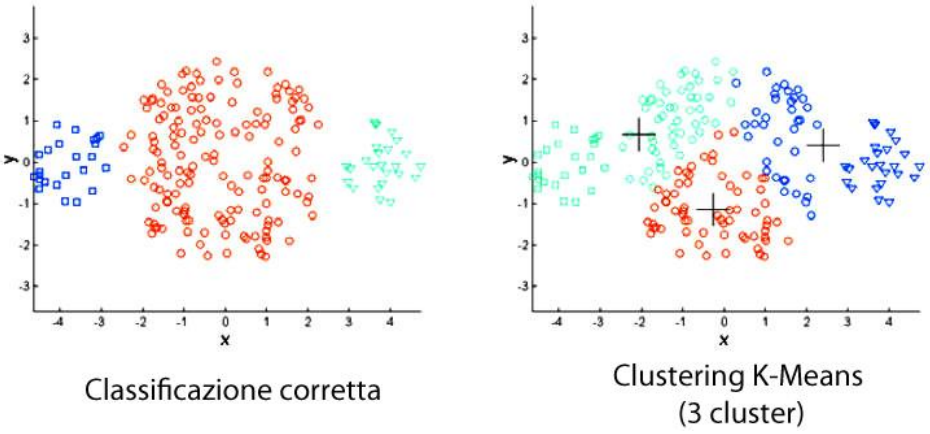


Figura 3.4 - Clustering KM con una classe di taglia maggiore delle altre.

- Se i cluster possiedono *densità differenti*, l'algoritmo difficilmente manterrà uniti cluster sparsi e con una bassa densità (si veda Figura 3.5).

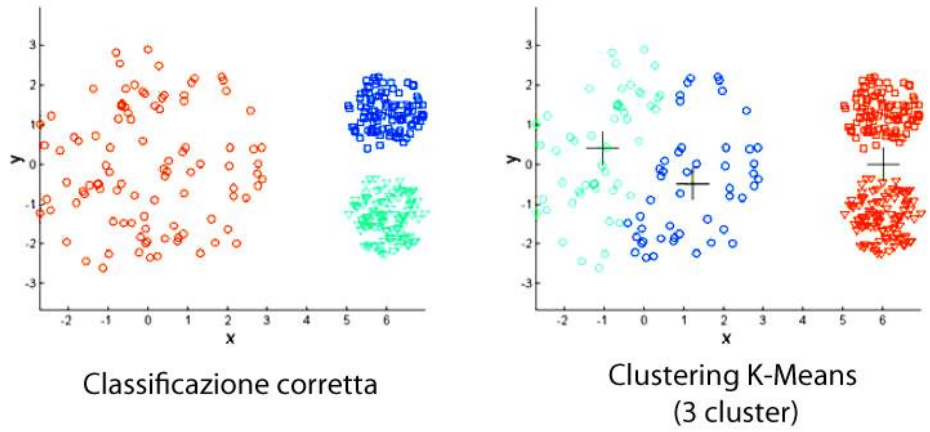


Figura 3.5 - Clustering KM applicato su cluster con densità differenti.

- Se i cluster possiedono *forme "non globulari"*, ovvero se la composizione assunta dai vari cluster possiede forme non regolari e che possono confondersi tra loro, la classificazione non avrà un esito positivo (si veda Figura 3.6).

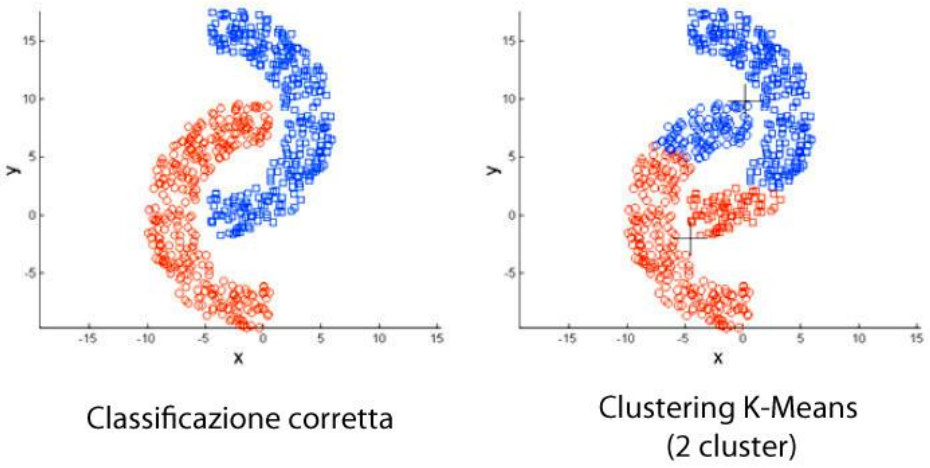


Figura 3.6 - Clustering KM applicato a cluster di forme non omogenee.

3.4. Algoritmo Fuzzy C-Means

L'algoritmo *Fuzzy C-Means (FCM)* è una particolare variante dell'algoritmo K-Means. [17]

Gli algoritmi di *Hard Clustering*, come l'algoritmo KM precedentemente descritto (si veda Paragrafo 3.3), possiedono numerose limitazioni e non riescono a gestire i cosiddetti *"borderline cases"* (o casi limite). Nell'insieme dei punti da classificare è possibile trovare punti, la cui appartenenza ad un determinato cluster sia incerta. Gli algoritmi esclusivi non consentono casi di pluri-appartenenza, per questo motivo ogni punto viene assegnato al cluster con valore di similarità maggiore. Spesso questa scelta porta ad errori di classificazione, che possono essere risolti introducendo una componente non deterministica nell'algoritmo di clustering.

L'algoritmo FCM appartiene alla famiglia degli algoritmi partizionali non-esclusivi (ovvero di ***Soft Clustering***). Questo algoritmo è caratterizzato da una *"logica fuzzy"* (il termine "fuzzy" letteralmente significa "sfocatura"), ovvero consente di attribuire ai dati valori probabilistici di appartenenza ai cluster.

Gli algoritmi di *fuzzy clustering* assegnano ai cluster valori di appartenenza reali compresi tra 0 (indica nessuna similarità con il cluster, e quindi nessuna probabilità di appartenenza) e 1 (completa similarità).

Per questo motivo:

- ogni dato può parzialmente appartenere a cluster differenti;
- la somma della appartenenze, relative ad un punto, rispetto ad ogni possibile cluster deve essere uguale a 1;
- la somma delle appartenenze all'interno della matrice di membership U, non deve superare il valore totale di *N*.

L'algoritmo FCM mantiene tutti i termini introdotti nella definizione dell'algoritmo KM (si veda Paragrafo 3.3) e aggiunge i seguenti termini utili:

- si definisce "*partizione fuzzy"*, una famiglia di cluster $\mathcal{P} = \{A_1, A_2, \dots, A_C\}$, tali che ogni oggetto può avere un'appartenenza parziale rispetto a molteplici cluster.
- si definisce l'esponente *m* come *"fuzziness factor"*, ovvero un'esponente pesato che consente di attribuire maggior rilevanza ai valori di membership contenuti nella matrice *U*. (Aumentando il valore di *m*, si aumenta la "spaccatura" tra i cluster di appartenenza e i cluster rifiutati da ogni singolo punto). Il valore consigliato in letteratura è 2. [18]

3.4.1. Pseudocodifica (FCM)

L'algoritmo FCM mantiene la stessa struttura esecutiva dell'algoritmo KM, aggiungendo alcune procedure di controllo e aggiornamento. In particolare vengono inserite le chiamate al calcolo della funzione obiettivo, in quanto quest'ultima entra a far parte dei criteri di convergenza utili alla terminazione dell'algoritmo.

$$\boldsymbol{J_{FCM}(U,V)} = \sum_{i=1}^{C}\sum_{k=1}^{N} u_{ik}^{m} \cdot \|X_k - V_i\|^2 = \sum_{i=1}^{C}\sum_{k=1}^{N} u_{ik}^{m} \cdot (d_{ik})^2 \qquad (2)$$

Inoltre vengono modificate le procedure di aggiornamento della *matrice di membership (3)* e dei nuovi prototipi di *centroidi (4).*

$$\boldsymbol{u_{ik}} = \frac{1}{\sum_{j=1}^{C}\left(\frac{\|X_k - V_i\|^2}{\|X_k - V_j\|^2}\right)^{2/(m-1)}} = \frac{1}{\sum_{j=1}^{C}\left(\frac{d_{ik}}{d_{ij}}\right)^{2/(m-1)}} \tag{3}$$

$$con\ \ i = 1, \dots, C\ \ e\ k = 1, \dots, N$$

$$\boldsymbol{V_i} = \frac{\sum_{k=1}^{N} u_{ik}^{m} \cdot X_k}{\sum_{k=1}^{N} u_{ik}^{m}} \tag{4}$$

$$con\ \ u_{ik} \in [0,1], \quad \sum_{j=1}^{C} u_{ik} = 1 \ \ \forall k, \quad 0 < \sum_{j=1}^{C} u_{ik} < N \ \ \forall i$$

Analogamente all'algoritmo KM, l'FCM termina la sua esecuzione al raggiungimento di almeno una delle seguenti condizioni di *convergenza*:

I. *(vedi Paragrafo 3.3)*

II. *(vedi Paragrafo 3.3)*

III. la differenza della funzione obiettivo, calcolata su due iterazioni consecutive, produce un valore minore del valore di soglia predefinito.

Pseudocodice (Fuzzy C-Means)

```
(Input: X, N, V)(*)
Inizializza X, N, V, C, Ƥ, U;
Ripeti{
      F = Calcola funzione obiettivo();
      Aggiorna centroidi();
      Aggiorna membership();
      Aggiorna soluzione();
      Fnew = Calcola funzione obiettivo();
} finché (U viene modificata && n_iterazioni < max_iterazioni
            && (F-Fnew) > Soglia_accuracy )
(Output: Ƥ)
```

(*) Eventuali input facoltativi possono essere: il numero massimo di iterazioni, il valore di soglia per la funzione obiettivo, il valore per il fuzziness factor.

3.4.2. Variante: Algoritmo Conditional Fuzzy C-Means

Esiste un'ulteriore variante dell'algoritmo FCM, chiamata *Conditional Fuzzy C-Means*, che contempla casi in cui si conoscano informazioni aggiuntive riguardanti i cluster.

In particolare, questa variante C-*FCM* [19], permette di modificare la funzione di membership (3) aggiungendo una condizione ausiliaria f_k, che ha il compito di dare maggior peso a determinati elementi del dataset, di cui si conoscano informazioni aggiuntive (ad esempio, se si conoscono punti che possono essere poco rilevanti ai fini della classificazione, è possibile "penalizzarli" nella valutazione di membership, assegnando un valore di f_k molto piccolo).

La funzione di membership dell'algoritmo C-FCM, risulta così modificata:

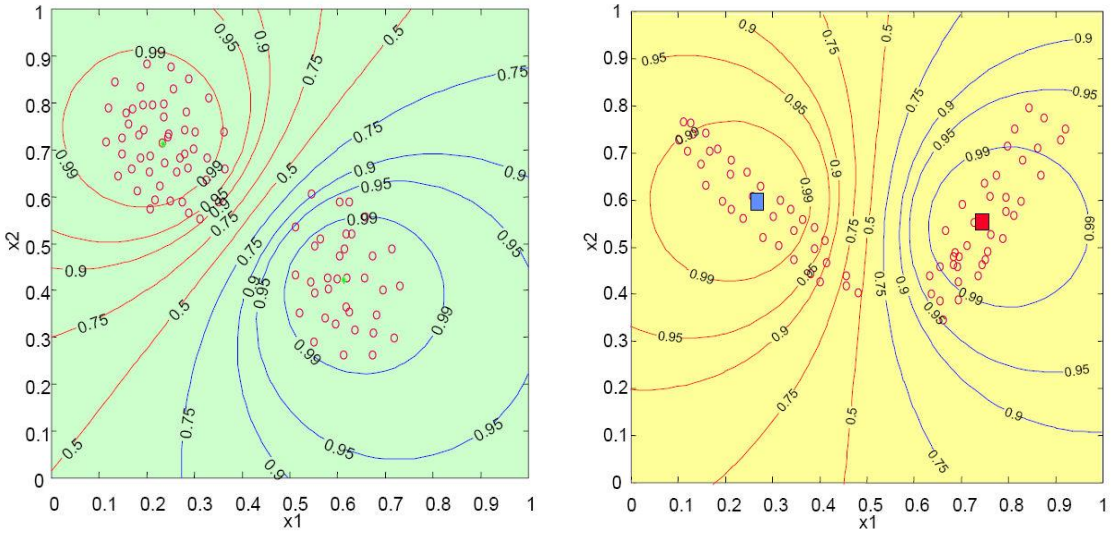
$$\boldsymbol{u_{ik}} = \frac{f_k}{\sum_{j=1}^{C}\left(\frac{\|X_k - V_i\|^2}{\|X_k - V_j\|^2}\right)^{2/(m-1)}} = \frac{f_k}{\sum_{j=1}^{C}\left(\frac{d_{ik}}{d_{ij}}\right)^{2/(m-1)}} \qquad (5)$$

$$con\ \ i = 1, \dots, C\ , \qquad k = 1, \dots, N, \qquad f_k \in [0,1], \qquad \sum_{j=1}^{C} u_{ik} = f_k$$

3.4.3. Limiti del Soft Clustering

La variante *fuzzy* dell'algoritmo *K-Means*, permette di risolvere diversi dei problemi incontrati con l'algoritmo KM (per esempio i casi limite affrontati nel Paragrafo 3.3.2, legati a *densità e grandezza* dei cluster). Inoltre nella maggior parte dei casi, permette di ottenere una convergenza più robusta e più consolidata.

Algoritmi come il FCM continuano ad avere problemi (si veda Figura 3.7) con situazioni dove i cluster possiedono forme allungate o schiacciate rispetto alla composizione circolare (o globulare tipica):



(a) (b)

Figura 3.7 - (a) Essendo FCM un algoritmo basato sulla *distanza euclidea*, e avendo aggiunto un criterio di appartenenza ai cluster non-esclusivo, produce risultati molto soddisfacenti in situazioni in cui i cluster possiedono forme sferiche o "globulari".
- (b) L'algoritmo continua ad avere problemi con cluster disposti in insiemi "allungati".
(I valori rappresentati sulle linee rappresentano i valori di appartenenza al cluster).

3.5. Algoritmo Geometrically Guided C-FCM

L'algoritmo *Geometrically Guided Conditional Fuzzy C-Means (GGC-FCM)* è una particolare variante dell'algoritmo *C-FCM*, che utilizza le informazioni spaziali (ovvero le coordinate dei singoli punti sul piano cartesiano) nel processo di costruzione dei prototipi dei cluster. [20]

Se si considera il contesto delle immagini, per calcolare il contributo di un singolo pixel all'interno di un determinato cluster, occorrerà tener conto sia delle informazioni spettrali (valori contenuti nelle bande multidimensionali) che delle informazioni spaziali (coordinate dei punti sul piano bidimensionale) del pixel.

L'*idea* fondante dell'algoritmo è quella che, preso un dato pixel dell'immagine, esista una correlazione con i suoi pixel vicini, e che la classe di appartenenza maggiormente rappresentata nelle sue vicinanze, sia anche la classe del pixel in questione.

3.5.1. Pseudocodifica (GGC-FCM)

Come per i precedenti algoritmi, occorre definire alcuni termini che caratterizzano questa particolare variante (per un elenco completo, si veda Paragrafo 3.4).

- Si definisce il concetto di *"partition image"*, ovvero l'insieme di tutti i valori di appartenenza u_{ij} legati ad un singolo cluster. Nella matrice di appartenenza *U* (definita nel Paragrafo 3.3) di grandezza *NxC*, esistono *C partition image*, ognuna relativa ad uno dei diversi cluster dell'immagine.

- Si definisce $\boldsymbol{W}$, come la *"neighbourhood window"* di un dato pixel X_i (letteralmente significa *"finestra di vicini"*). Questa finestra di pixel consente di verificare il "matching" (ovvero la similarità) tra la classe di appartenenza del pixel X_i, e la *"majority class"* dei pixel in *W*. Questo confronto basato su valori di membership fornisce un "valore di condizionamento", che aumenta se i pixel nella finestra *W* hanno valori di membership simili a quelli di X_i, e diminuisce se questi valori sono differenti.
  - Sia $\boldsymbol{sum\_u_{rc}}$, il vettore contenente le somme dei valori di membership per ogni *partition image* (sum_u_1), tale che:

$$\boldsymbol{sum_{u_{rc}}} = \left(sum_{u_1}, \dots, sum_{u_C}\right) \tag{6}$$

$$dove \quad sum\_u_1 = \sum_{r'c' \in W} u[r'c', i] \,, \quad \forall i = 1, \dots, C$$

  - Sia $\boldsymbol{iMax}$, il cluster rappresentante la *"majority class"*(letteralmente "classe di maggioranza"), ovvero la classe più rappresentata dai pixel appartenenti alla finestra *W*. Solo questa classe deve essere considerata nel calcolo di similarità tra la finestra di vicini *W* e il pixel centrale X_i, in quanto questa dovrebbe essere, in linea teorica, la classe di appartenenza del pixel dato. Utilizzando la formula (6) si ottiene:

$$\boldsymbol{iMax} = i | \max_i \left(sum_{u_{rc}}\right) \tag{7}$$

- Sia $\Delta m_{rc,iMax}$ la *"mean membership deviation"* (media degli scarti dei valori di membership presenti nella finestra *W*, rispetto al pixel di centro (r, c)), tale che:

$$\Delta m_{rc,iMax} = \frac{1}{s^2 - 1} \sum_{r'c' \in W} |u[rc, iMax] - u[r'c', iMax]| \tag{8}$$

- Si definisce f_{rc}, come l'*indice di condizionamento* del pixel centrale della finestra *W*, che rappresenta il valore di membership pesato rispetto al valor medio degli scarti (definito nella formula (7)).

$$f_{rc} = \Delta m_{rc,iMax} \cdot u[rc, iMax] \tag{9}$$

- Si introduce il concetto di *"Edge-preserving threshold"* (che letteralmente significa "soglia di conservazione dei bordi"). Questo valore di soglia permette di identificare i pixel di confine (nel caso in cui la finestra *W* sia posizionata sulla transizione tra due piani dell'immagine), che potrebbero portare a falsi incrementi della $\Delta m_{rc,iMax}$ nella formula (8).

$$EdgeThreshold = \frac{floor(s/2) \cdot s}{s^2 - 1} \tag{10}$$

Pseudocodice (Geometrically Guided Conditional Fuzzy C-Means)

```
(Input: X, N, V, W)(*)
Inizializza X, N, V, C, P, U;
Calcola EdgeThreshold (10);
Ripeti{
      F = Calcola funzione obiettivo();
      Aggiorna centroidi();
      Aggiorna membership();
      Aggiorna valori rispetto alla finestra W {
            Calcola somme di membership (6);
            Calcola classe di maggioranza iMax (7);
            Calcola la media degli scarti di U in W (8);
```

```
            Se Δm_{rc,iMax} > EdgeThreshold
                  Calcola indice di condizionamento f_rc (9);
      }
      Aggiorna soluzione();
      Fnew = Calcola funzione obiettivo();
} finché (U viene modificata && n_iterazioni < max_iterazioni
            && (F-Fnew) > Soglia_accuracy )
(**)[Applica Defuzzification()];
(Output: Ƥ)
```

(*) Eventuali input facoltativi possono essere: il numero massimo di iterazioni, il valore di soglia per la funzione obiettivo, il valore per il fuzziness factor.
(**) La procedura di Defuzzification permette di restituire una partizione hard dei cluster, e permette di stabilire una classe di rifiuto per i pixel con indice di condizionamento ridotto.

3.5.2. Defuzzification (Soft to Hard Clustering)

Al raggiungimento di una delle tre *condizioni di convergenza* (si veda Paragrafo 3.4.1), occorre applicare una procedura di *Defuzzification* alla matrice di appartenenza *U*, ovvero una procedura in grado di restituire una *partizione hard* da *U*, per ottenere la classificazione finale dei pixel.

La procedure di *Defuzzification* più comune è detta: *Maximum Membership*. Questa procedura assegna ad ogni pixel la classe con il valore di membership più elevato, prelevandola dalla rispettiva riga nella matrice *U*.

Questa procedura non permette di discriminare i pixel "incerti", ovvero quei pixel che possiedono un indice di condizionamento molto basso. Per questo motivo occorre definire un valore di soglia, in grado di distinguere questi pixel. In letteratura viene denominato *Outlier Threshold* e viene definito come segue:

$$OutlierThreshol = EdgeThreshold \cdot 0{,}5 \qquad (11)$$

Questo valore di soglia permette di effettuare una corretta classificazione esclusiva, andando ad identificare i pixel "rifiutati" in una classe specifica denominata *Reject Class (o classe di rifiuto)*.

Capitolo 4

Test e sperimentazioni

4.1. Casi di studio: "ROSIS Data – Campaign over Pavia"

Nell'ambito dei test sugli algoritmi di clustering, sono stati scelti due casi di studio relativi alla città di Pavia. In particolare sono state analizzate due immagini iperspettrali utilizzate in numerosi articoli in ambito scientifico, per valutare le prestazioni di algoritmi di classificazione. Le due immagini riguardanti la città di Pavia vengono comunemente chiamate ***Pavia University*** e ***Pavia Centre***, e ritraggono rispettivamente il centro universitario della facoltà di ingegneria, e il centro cittadino. [21]

Queste due immagini [22] sono state acquisite dal sensore *ROSIS (Reflective Optics System Imaging Spectrometer)* durante una "campagna di telerilevamento" aerea, effettuata sulla città di Pavia, nel nord Italia (45° 11' N, 9° 9' E), in data: 8 luglio 2002.

Entrambe possiedono delle zone di pixel che non contengono informazioni, e che sono frutto di una rilevazione effettuata con un errato *field of View* del sensore. (Nel campo del telerilevamento, il *field of View o "campo visivo"* di un sensore, indica la zona territoriale coperta dalla visuale del sensore). Per questo motivo, entrambe le immagini necessitano di un processo di rifinitura, che permetta di scartare le porzioni d'immagine non significative.

4.1.1. Pavia University

L'immagine di *Pavia University (PaviaU)* possiede le seguenti caratteristiche:

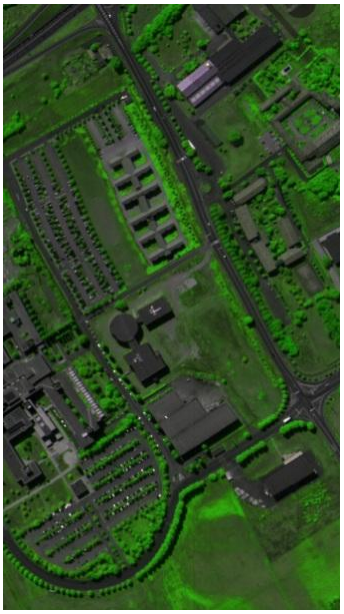
- Dimensioni del file: 41MB;
- Luogo rilevato: Università di Pavia (zona della facoltà di Ingegneria);
- Numero di bande: 103;
- Dimensioni (in pixel): 340x610;
- Risoluzione spaziale: 1,3 metri;
- Numero di classi: 9 (Alberi, Asfalto, Bitume, Ghiaia, Metallo, Ombra, Mattoni, Prato, Terra).

Di seguito verranno rappresentate alcune bande significative di PaviaU (si veda Figura 4.1), e alcune combinazioni false-color (si veda Figura 4.2).

(a) (b) (c)

Figura 4.1 : - (a) Banda 90 in tonalità di grigio.
- (b) Banda 60 in tonalità di grigio.
- (c) Banda 40 in tonalità di grigio.
(Si veda Figura 4.2 per una visualizzazione in false-color delle stesse tre bande).

(a)

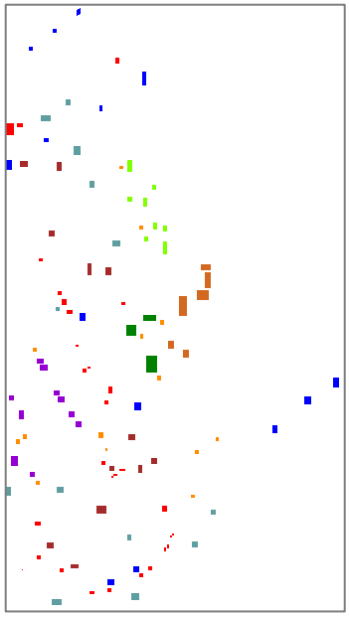


(b)

Figura 4.2 - (a) Bande 90-60-40 (rgb, false-color).
Per enfatizzare in rosso, le zone verdi di Pavia University.

- (b) Bande 70-90-50 (rgb, false-color).
Per ottenere un risultato più foto-realistico.

Per questa immagine sono stati forniti due file denominati: *PaviaU_TrainingSet* e *PaviaU_TestSet* contenenti, per alcuni dei pixel dell'immagine, l'informazione sulla classe di appartenenza (si vedano Figura 4.3 (a) e (b)).



(a)

(b)

Figura 4.3 - (a) Rappresentazione di *PaviaU_TrainingSet*. - (b) Rappresentazione di *PaviaU_TestSet*.

Osservazione I:

Analizzando i file di training e test set, è stato possibile costruire una tabella contenente le informazioni relative ad ogni classe (si veda Tabella 4.1*). Nella tabella sono state riportate le informazioni relative al tipo di materiale rappresentatato, al numero di pixel presenti nei file sopracitati, e al colore utilizzato per la rappresentazione grafica.*

Classi	***Tipo di copertura***	***Punti (Training Set)***	***Punti (Test Set)***	***Colore***
1	Alberi	524	3064	
2	Asfalto	548	6631	
3	Bitume	375	1330	
4	Ghiaia	392	2099	
5	Metallo	265	1345	
6	Ombra	231	947	
7	Mattoni	514	3682	
8	Prato	540	18649	
9	Terra	532	5029	

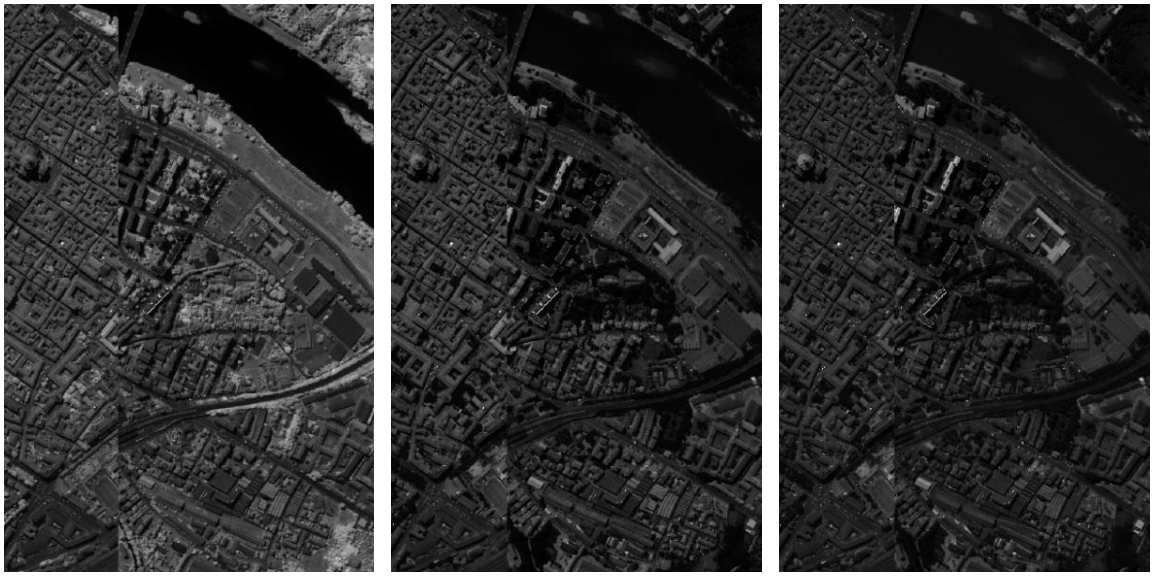
Tabella 4.1 – Rappresenta il contenuto informativo legato ad ogni classe identificata nel *groundtruth*. Dai dati in tabella è possibile ricavare il totale dei punti classificati nell'immagini di *Training Set (3921)* e nell'immagine di *Test Set (42776).*

4.1.2. Pavia Centre

L'immagine di *Pavia Centre (PaviaC)* possiede le seguenti caratteristiche:

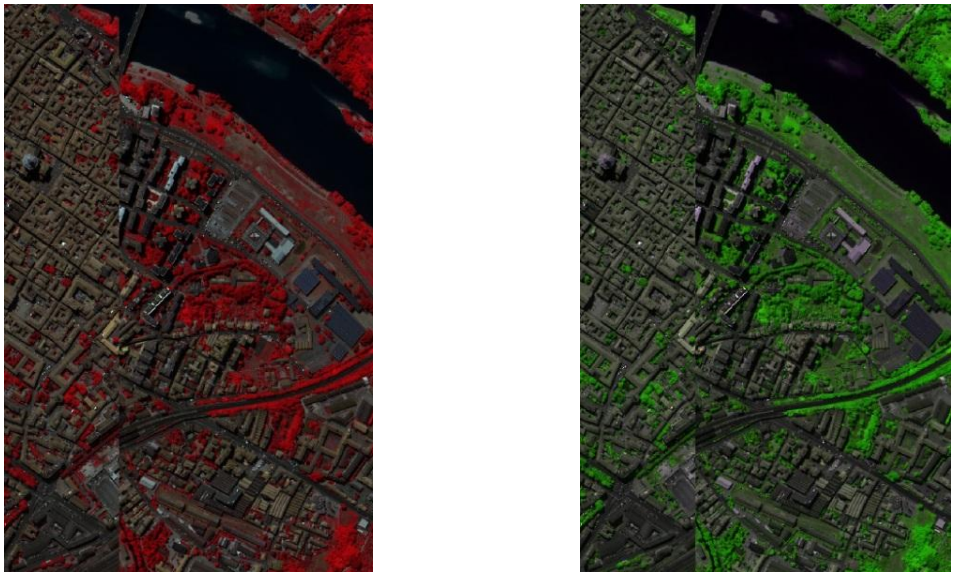
- Dimensioni del file: 152MB;
- Luogo rilevato: Centro urbano della città di Pavia;
- Numero di bande iperspettrali: 102;
- Dimensioni (in pixel): 715x1096;
- Risoluzione spaziale: 1,3 metri;
- Numero di classi: 9 (Acqua, Alberi, Asfalto, Mattoni, Bitume, Piastrelle, Ombra, Prato, Terra).

Vengono presentate di seguito alcune delle bande significative per l'immagine di PaviaC (si veda Figura 4.4) e alcune delle combinazioni false-color più rappresentative (si veda Figura 4.5).



(a) (b) (c)

Figura 4.4 - (a) Banda 90 in tonalità di grigio.
- (b) Banda 60 in tonalità di grigio.
- (c) Banda 40 in tonalità di grigio.
(Si veda Figura 4.5 per una visualizzazione in false-color delle stesse tre bande)

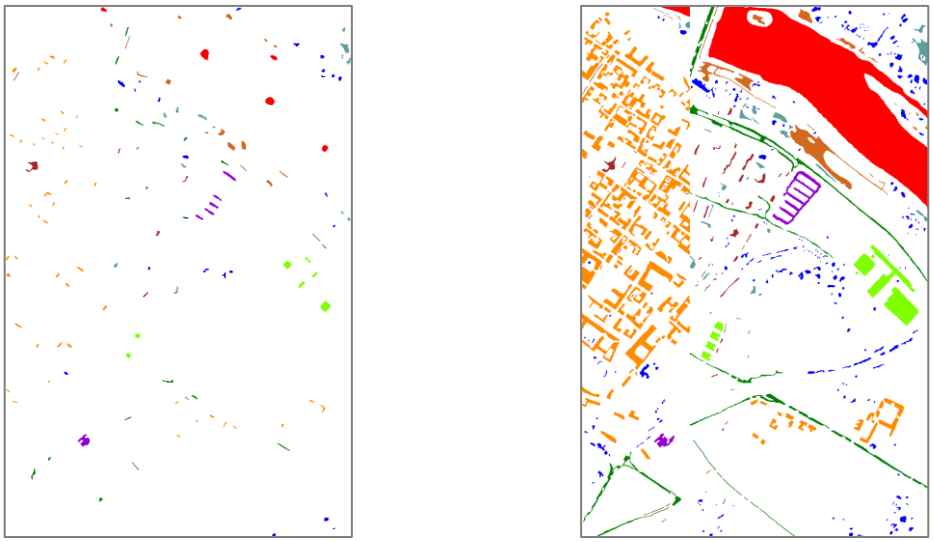


(a) (b)

Figura 4.5 - (a) Bande 90-60-40 (rgb, false-color). Per enfatizzare in rosso, le zone verdi di Pavia Centre.

- (b) Bande 70-90-50 (rgb, false-color). Per ottenere un risultato più foto-realistico.

Analogamente all'immagine di *PaviaU*, vengono visuallizzati i file d'immagine: *PaviaC_TrainingSet* (Figura 4.6 (a)) e *PaviaC_TestSet* (Figura 4.6 (b)).



(a) (b)

Figura 4.6 - (a) Rappresentazione di *PaviaC_TrainingSet*. - (b) Rappresentazione di *PaviaC_TestSet*.

La Tabella 4.2 riporta l'elenco delle classi presenti nei file *PaviaC_TrainingSet* (e *PaviaC_TestSet* con alcune informazioni statistiche.

Classi	***Tipo di copertura***	***Punti (Training Set)***	***Punti (Test Set)***	***Colore***
1	Acqua	824	65971	
2	Alberi	820	7598	
3	Asfalto	816	9248	
4	Mattoni	808	2685	
5	Bitume	808	7287	
6	Piastrelle	1260	42826	
7	Ombra	476	2863	
8	Prato	824	3090	
9	Terra	820	6584	

Tabella 4.2 – Rappresenta il contenuto informativo legato ad ogni classe identificata nel groundtruth. Dai dati in tabella è possibile ricavare il totale dei punti classificati nell'immagini di *Training Set (7456)* e nell'immagine di *Test Set (148152).*

4.2. Clustering Validation

Per poter valutare correttamente i risultati ottenuti dal processo di classificazione non supervisionata, occorre definire delle metriche di valutazione degli algoritmi di clustering.

Per *Clustering Validation* si intende quell'insieme di *indici* e *misure* utilizzati per valutare l'accuratezza di un algoritmo di clustering.

In particolare nelle tecniche di *External Evaluation Clustering*, vengono confrontati i risultati del clustering con i valori attesi, ottenuti da file esterni che possiedono una classificazione corretta dell'immagine (ad esempio nel contesto delle immagini di *Pavia* (Paragrafo 4.1), le immagini *Training* e *Test* possono essere utilizzate per questo scopo).

Di seguito verranno introdotti alcune *misure* di "valutazione esterna".

4.2.1. Rand Measure

La *Rand Misure (anche detta "Indice di Rand" o "Misura di Rand")* è stata introdotta da M.Rand nel 1971. [23]

Questo indice di valutazione confronta l'insieme dei cluster ottenuti dall'algoritmo di clustering analizzato (nel seguito *X*), con quello dei cluster attesi (nel seguito *Y*), e consente di verificare il valore di similarità tra questi due insiemi. In particolare con questo indice è possibile verificare la percentuale di scelte corrette che sono state compiute dall'algoritmo, per identificare l'insieme *X*.

Formulazione matematica:

$$RandIndex(X,Y) = \frac{TP + TN}{TP + FP + TN + FN} \quad (1)$$

$dove\ TP, FP, TN, FN\ contano\ le\ "coppie\ di\ dati"\ che:$
$TP = TruePositive,\quad appartengono\ allo\ stesso\ cluster\ sia\ in\ X\ che\ in\ Y;$
$FP = FalsePositive,\quad appartengono\ allo\ stesso\ cluster\ in\ Y, e\ a\ diversi\ cluster\ in\ X;$
$TN = TrueNegative,\quad appartengono\ a\ cluster\ diversi\ sia\ in\ X\ che\ in\ Y;$
$FN = FalseNegative,\quad appartengono\ allo\ stesso\ cluster\ in\ X, e\ a\ divers\ cluster\ in\ Y;$

4.2.2. F-Measure

Il criterio denominato *F-Measure (o anche $F_\beta score$)* si basa su due concetti teorici di pattern recognition: *precision (p) e recall (r)*. Questo indice può infatti essere considerato una media pesata tra *p* e *r*, con peso β.

- Il concetto di *precision*, rappresenta la frazione di dati correttamente classificati (*TP*), sulla totalità di dati da classificare (*TP+FP*):

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

- La *recall* invece rappresenta la frazione di dati correttamente classificati (*TP*), sulla totalità di classi da identificare (*TP*+*FN*):

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

Sfruttando le formule (2) e (3) è possibile definire *F-Measure* come:

$$FMeasure = F_\beta = (1 + \beta^2) \cdot \frac{Precision \cdot Recall}{Precision + Recall} \tag{4}$$

Tipicamente viene utilizzata la formula (4) con valori di β tra 0 e 2 (ad esempio: $F_{0.5}$, F_1 o F_2).

4.2.3. Jaccard Index

L'*indice di Jaccard (o Jaccard Index)* mette a confronto due dataset, e fornisce un valore di similarità compreso tra 0 e 1 (dove 0 indica che i due dataset non hanno elementi in comune, e 1 indica che i due dataset sono identici).

Questo indice può essere calcolato come il rapporto tra l'intersezione degli elementi unici appartenenti a entrambi i dataset, e l'insieme degli elementi unici totali. Formalmente:

$$JaccardIndex(D_1, D_2) = \frac{|unique(D_1 \cap D_2)|}{|unique(D_1 \cup D_2)|} = \frac{TP}{TP + FP + FN} \tag{5}$$

4.2.4. Fowlkes-Mallows Index

L'*Indice di Fowlkes-Mallows* è stato introdotto da E.B.Fowlkes e C.L.Mallows nel1983.

Questo indice rappresenta il valore di similarità tra due cluster (dove più alto è il valore di indice, e più alta sarà la similarità dei due cluster).

Per definire l'indice vengono utilizzati i concetti di *precision* e *recall* visti in precedenza (si vedano (2) e (3)):

$$FMindex = \sqrt{\frac{TP}{TP + FP} \cdot \frac{TP}{TP + FN}} \quad (6)$$

4.3. Sperimentazioni (KM,FCM,GGC)

In questo paragrafo vengono riportati i risultati dei test effettuati sugli algoritmi di Clustering introdotti nel Capitolo 3.

Gli algoritmi di classificazione non supervisionata sono stati applicati alle immagini di Pavia University e Pavia Centre, definite nel Paragrafo 4.1, su entrambi i *dataset* presenti (*training set*, e *test set*).

Tutte le diverse tipologie di test affrontate, sono state ripetute numerose volte, utilizzando centroidi *random* differenti (ovvero generati casualmente, con semi di generazione diversi). I risultati riportati nelle seguenti tabelle sono stati ottenuti calcolando la media aritmetica dei risultati ottenuti nelle ripetizioni. Questo procedimento permette di ottenere risultati attendibili, evitando di pregiudicare i risultati che potrebbero essere inficiati da una particolare conformazione di centroidi random.

4.3.1. Test: PaviaU TrainingSet

Nella seguente tabella (si veda Tabella 4.3) sono riportati i risultati ottenuti sull'immagine di *PaviaU_TrainingSet.*

Alg.	Iteration (#)	Rand Misure	Jaccard Index	FM Index	Fm(0.5)	Fm(1.0)	Fm(2.0)	Unknown (%)
KM	65	0,847	0,281	0,442	0,475	0,439	0,408	0
FCM	41	0,864	0,318	0,485	0,513	0,483	0,456	0
GGC	37	0,864	0,320	0,487	0,516	0,484	0,456	4,4

Tabella 4.3 - Valutazione degli indici relativi al dataset: *PavaU_TrainingSet*.

Dai successivi grafici, ricavati dalle dieci ripetizioni dei test, sono stati messi a confronto: il numero di iterazioni compiute dai tre diversi algoritmi (si veda Figura 4.7),

e il valore di uno degli indici di *Validation Clustering*, più significativi, l'*Indice di Rand (si veda* Figura 4.8*)*:

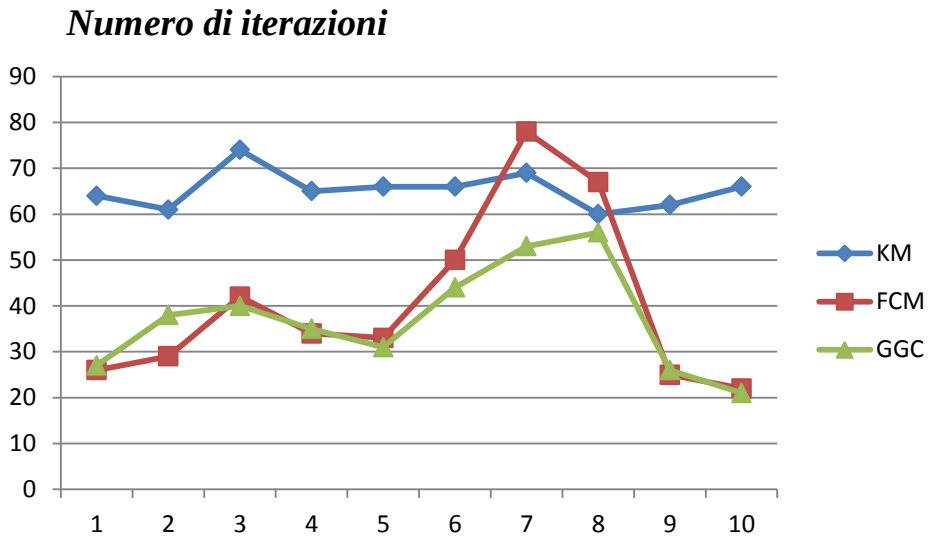


Figura 4.7 - Grafico raffigurante l'andamento del numero di iterazioni per *PaviaU_TrainingSet.*

Dal grafico si può facilmente notare che, considerando il caso medio, gli algoritmi FCM e GGC-FCM necessitano di un minor numero di iterazioni.

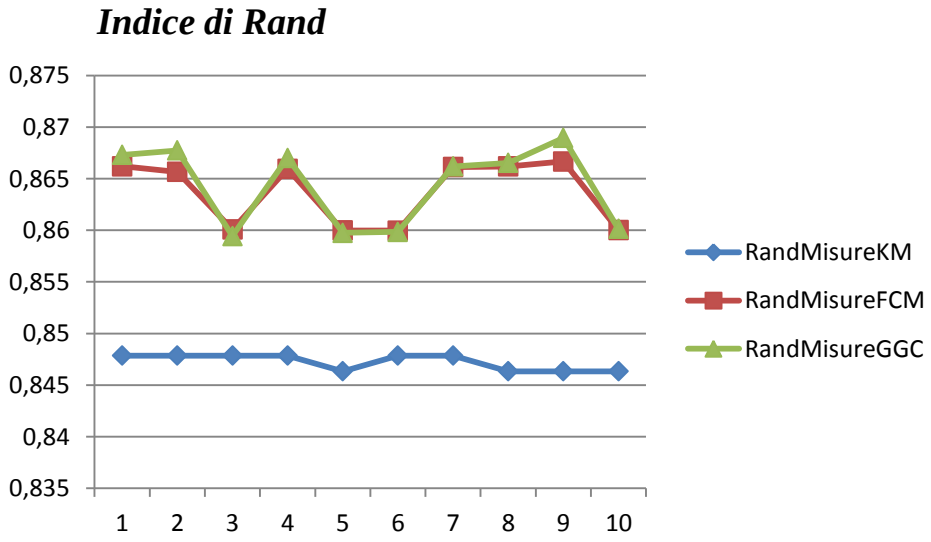


Figura 4.8 - Grafico raffigurante l'andamento *dell'Indice di Rand* per *PaviaU_TrainingSet.*

Gli algoritmi FCM e GGC-FCM, su un dataset come PaviaU_TrainingSet, che risulta essere particolarmente ridotto e con insiemi di pixel ben raggruppati,

ottengono sempre precisioni superiori a quelle dell'algoritmo KM (sull'indice di Rand analizzato, e su tutti i restanti indici).

4.3.2. Test: PaviaU TestSet

Nella Tabella 4.4 sono riportati i risultati dei ottenuti sull'immagine di *PaviaU_TestSet.*

Alg.	Iteration (#)	Rand Misure	Jaccard Index	FM Index	Fm(0.5)	Fm(1.0)	Fm(2.0)	Unknown (%)
KM	38	0,758	0,394	0,566	0,601	0,562	0,529	0
FCM	99	0,767	0,233	0,398	0,319	0,378	0,465	0
GGC	106	0,771	0,247	0,412	0,338	0,395	0,476	9,0

Tabella 4.4 - Valutazione degli indici relativi al dataset: *PavaU_TestSet.*

Dalla Tabella 4.4 si può facilmente evincere come, al cambiamento dell'immagine di dataset, siano cambiate drasticamente le prestazioni. Con l'immagine di PaviaU_TestSet, che risulta essere molto più grande e complessa di PaviaU_TrainingSet (possedendo numerosi gruppi di pixel sparsi, e con forme non omogenee) gli algoritmi FCM e GGC-FCM ottengono un'efficienza media peggiore dell'algoritmo KM. In particolare sono cresciute le iterazioni richieste, e sono calate le prestazioni sugli indici di Jaccard, Fowlkes-Mallows, e F-Measure.

Di seguito verranno rappresentati i grafici risultanti dalle ripetizioni dei dieci test effettuati su *PaviaU_TestSet (si vedano* Figura 4.9 *e* Figura 4.10*).*

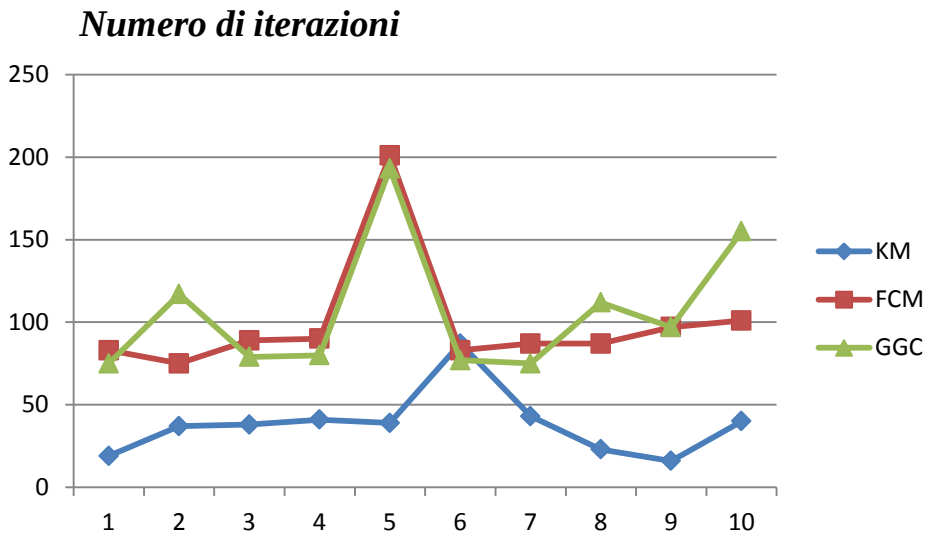


Figura 4.9 - Grafico raffigurante l'andamento del numero di iterazioni per *PaviaU_TestSet.*

Dal grafico si può notare che, sul dataset di PaviaU_TestSet, gli algoritmi della famiglia Fuzzy richiedono mediamente un numero di iterazioni superiore a quello dell'algorimo KM (con picchi di valori molto più elevati del valor medio).

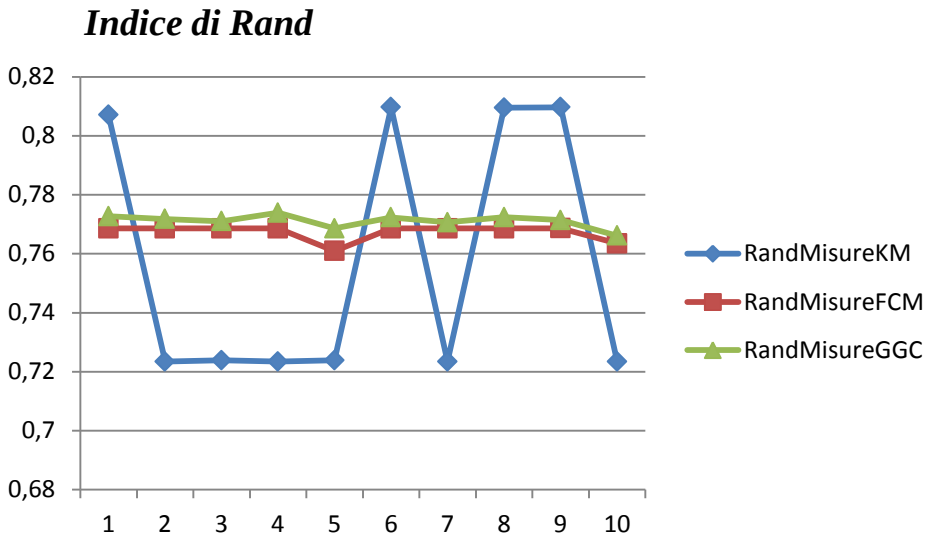


Figura 4.10 - Grafico raffigurante l'andamento *dell'Indice di Rand* per *PaviaU_TestSet.*

Dal grafico si evince che gli algoritmi FCM e GGC-FCM mantengono prestazioni più costanti, su tutte le esecuzioni casuali effettuate. L'algoritmo KM alterna ottimi valori di accuratezza, a valori decisamente peggiori.

4.3.3. Test: PaviaC TrainingSet

Nella Tabella 4.5 sono riportati i risultati ottenuti sull'immagine di *PaviaC_TrainingSet.*

Alg.	Iteration (#)	Rand Misure	Jaccard Index	FM Index	Fm(0.5)	Fm(1.0)	Fm(2.0)	Unknown (%)
KM	53	0,818	0,287	0,465	0,537	0,445	0,381	0
FCM	36	0,892	0,382	0,553	0,564	0,552	0,541	0
GGC	36	0,894	0,394	0,565	0,579	0,565	0,551	5,0

Tabella 4.5 - Valutazione degli indici relativi al dataset: *PavaC_TrainingSet*.

Analogamente al caso di studio di PaviaU, nel dataset di PaviaC_TrainingSet, gli algoritmi FCM e GGC-FCM ottengono i migliori risultati, sia dal punto di vista del numero di iterazioni, sia per la qualità dei risultati ottenuti.

I grafici seguenti riportano l'andamento del numero di iterazioni (Figura 4.11) e dell'*Indice di Rand* (Figura 4.12).

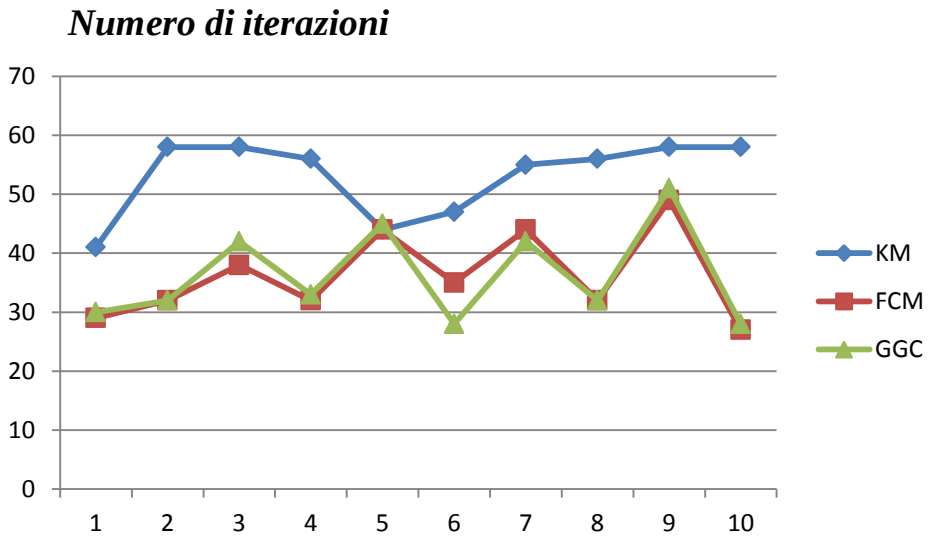


Figura 4.11 - Grafico raffigurante l'andamento del numero di iterazioni per *PaviaC_TrainingSet.*

Analogamente all'immagine di PaviaU_TrainingSet, gli algoritmi FCM e GGC-FCM necessitano di un minor numero di iterazioni medie, rispetto a KM.

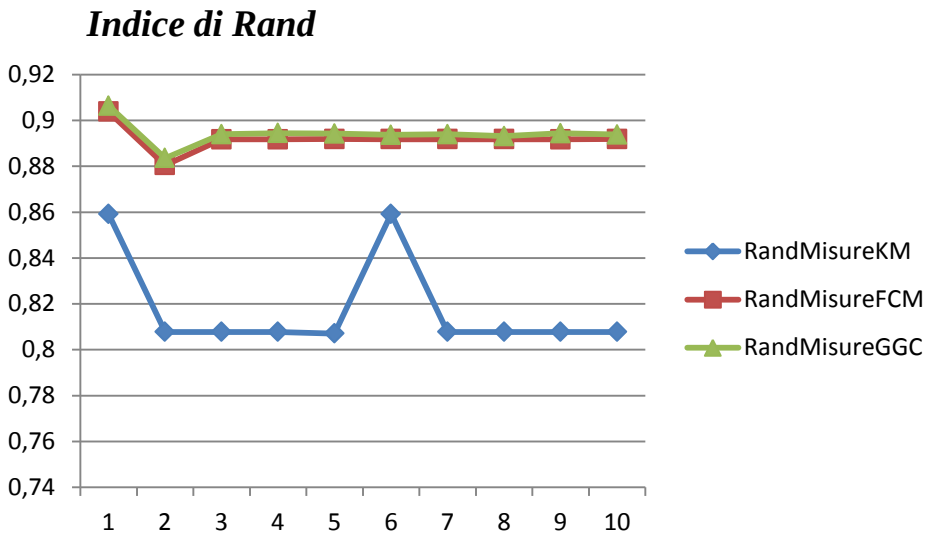


Figura 4.12 - Grafico raffigurante l'andamento *dell'Indice di Rand* per *PaviaC_TrainingSet*.

Analogamente al caso dell'immagine di PaviaU_TrainingSet, gli algoritmi FCM e GGC-FCM ottengono sempre precisioni superiori a quelle dell'algoritmo KM.

4.3.4. Test: PaviaC TestSet

Nella Tabella 4.6 sono riportati i risultati ottenuti sull'immagine di *PaviaC_TestSet.*

Alg.	Iteration (#)	Rand Misure	Jaccard Index	FM Index	Fm(0.5)	Fm(1.0)	Fm(2.0)	Unknown (%)
KM	89	0,927	0,765	0,869	0,833	0,867	0,903	0
FCM	75	0,871	0,599	0,751	0,687	0,740	0,808	0
GGC	70	0,886	0,659	0,791	0,747	0,783	0,829	8,9

Tabella 4.6 - Valutazione degli indici relativi al dataset: *PavaC_TestSet.*

Confrontando la precedente tabella, con quella dei risultati ottenuti sul caso di studio di PaviaU_TestSet (si veda Tabella 4.4), è possibile confermare il calo prestazionale degli algoritmi fuzzy (FCM e GGC-FCM) sulle immagini di TestSet. Ad ogni modo, in questo particolare caso di studio, le iterazioni richieste dall'algoritmo KM rimangono superiori alle altre, e gli indici di

valutazione dimostrano risultati qualitativamente accettabili anche nei casi di FCM e GGC-FCM.

Di seguito verranno riportati i grafici relativi all'andamento del numero di iterazioni (Figura 4.13) e dell'*Indice di Rand* (Figura 4.14) con il *dataset* di *PaviaC_TestSet.*

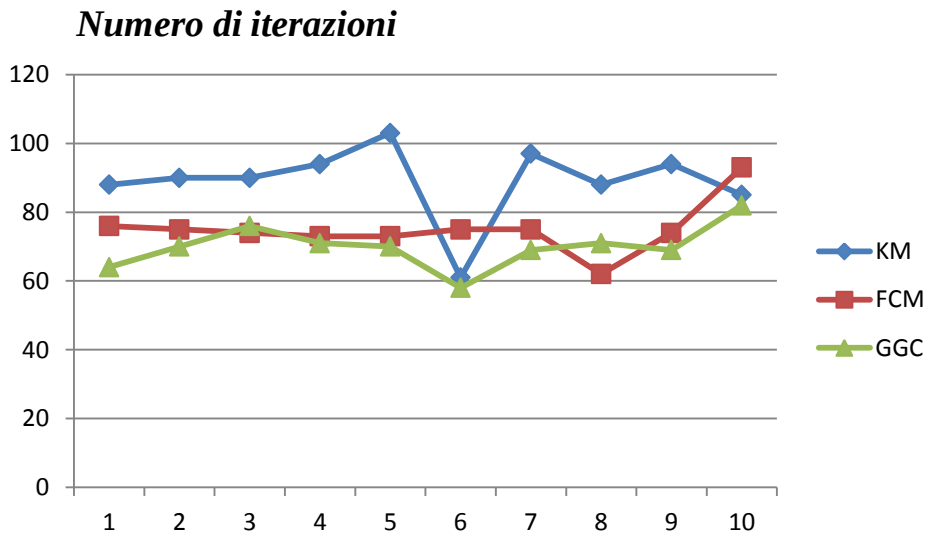


Figura 4.13 - Grafico raffigurante l'andamento del numero di iterazioni per *PaviaC_TestSet.*

Dal grafico si può notare che, sul dataset di PaviaC_TestSet, l'algoritmo KM richiede un maggior numero medio di iterazioni rispetto agli algoritmi Fuzzy.

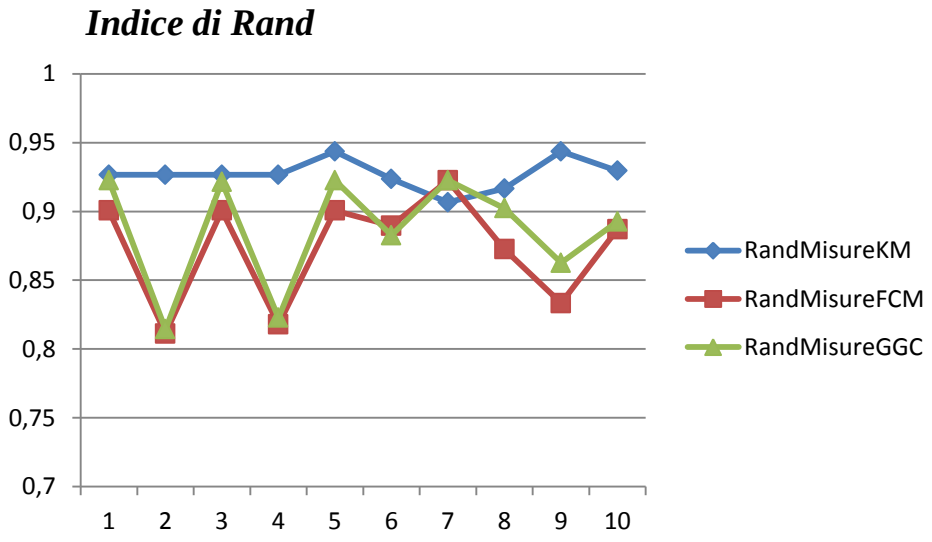


Figura 4.14 - Grafico raffigurante l'andamento *dell'Indice di Rand* per *PaviaC_TestSet*.

Dal grafico si evince che su questo particolare dataset, gli algoritmi FCM e GGC-FCM ottengono valori medi di accuratezza minori rispetto a quelli ottenuti dall'algoritmo KM.

Conclusioni

Nello sviluppo della tesi proposta sono stati affrontati i problemi di *caricamento*, *analisi* ed *elaborazione* delle immagini iperspettrali. Inoltre sono state affrontate tecniche per l'*estrazione* d'informazioni statistiche dalle immagini, e la *categorizzazione* di zone di territorio secondo particolari classi prestabilite dall'utente.

Per risolvere questi problemi è stata sviluppata un'applicazione in grado di fornire all'utente un ambiente grafico per la gestione e l'analisi di contenuti iperspettrali. In particolare sono state implementate diverse funzionalità relative alla visualizzazione di immagini satellitari, consentendo all'utente di scegliere tra diverse possibili tipologie di visualizzazione. È stato risolto il problema del caricamento delle immagini iperspettrali, ponendo particolare attenzione alla gestione della memoria, per garantire stabilità e usabilità del programma. Inoltre sono state implementate tecniche di estrazione delle informazioni statistiche, e algoritmi di classificazione non supervisionata delle immagini. Infine sono stati svolti dei test per confrontare gli algoritmi di clustering implementati, per enfatizzarne pregi e difetti.

I. Lavoro svolto

Il lavoro svolto in questo progetto di tesi, può essere suddiviso in quattro fasi ben delineate:

Nella parte iniziale è stato effettuato uno studio accurato sull'attuale stato dell'arte relativo al *Telerilevamento* e alle tecniche di analisi delle immagini satellitari.

Nella seconda fase è stata studiata la libreria *GDAL* per la gestione e il caricamento di *immagini iperspettrali*. Inoltre è stata progettata concettualmente e architetturalmente l'*applicazione* richiesta, dall'interfaccia grafica del front-end alla logica applicativa dei moduli operazionali. In questa fase è stata data particolare attenzione all'implementazione delle gerarchie di classi per la memorizzazione e l'elaborazione delle immagini iperspettrali.

Nella terza fase sono state studiate le tecniche di classificazione delle immagini. In particolare è stato affrontato il problema della *classificazione non supervisionata* delle immagini multidimensionali. Sono stati studiati e implementati algoritmi di clustering partizionali allo stato dell'arte, quali l'*Algoritmo K-Means* [16] e l'*Algoritmo Fuzzy C-Means* [17] (nelle sue possibili varianti).

Nella quarta e ultima fase, sono stati presi in considerazione due *casi di studio* relativi alla città di Pavia, che sono stati utilizzati per svolgere i *test* di confronto tra gli algoritmi di clustering implementati. In questa fase finale sono state applicate tecniche di *Clustering Validation* per confrontare l'accuratezza dei risultati ottenuti.

II. Sviluppi futuri

Al termine di questo progetto di tesi è giusto e doveroso suggerire un proseguimento nello studio delle tecniche di telerilevamento iperspettrale. Come è stato dimostrato in questo lavoro di tesi, questa "giovane" branca possiede ancora molte potenzialità inespresse. Ricercando e applicando nuove soluzioni di classificazione delle immagini, è possibile ottenere grandi risultati in innumerevoli campi di applicazione, dall'analisi socio-economica delle zone territoriali d'interesse allo studio idro-geologico di fenomeni nel tempo. Continuando a condividere gratuitamente immagini e informazioni satellitari su scala mondiale, e seguendo la strada intrapresa da alcuni enti geo-spaziali che stanno promuovendo le attività di telerilevamento, si potrebbe ampliare enormemente il bacino d'utenza del campo iperspettrale.

L’applicazione sviluppata nel progetto di tesi, è stata progettata basandosi sulle regole della “buona programmazione a oggetti”, che hanno garantito proprietà quali l’*ereditarietà* dei componenti*, l’estendibilità e la riusabilità* del codice. Il modello architetturale adottato, ha portato allo sviluppo di moduli indipendenti e intercambiabili.

Per questo motivo è possibile suggerire alcuni sviluppi futuri dell’applicazione:

- Migliorare le *performance* dell’applicazione su macchine a 64 bit, andando a ricompilare le librerie satellitari *GDAL* (attualmente viene utilizzata la versione a 32 bit).
- Aumentare il numero di formati geosatellitari supportati.
- Estendere gli algoritmi di clustering implementati, permettendo la classificazione di immagini con un numero di cluster maggiori del numero di classi presenti. Questa tecnica può migliorare sensibilmente l’accuratezza degli algoritmi su immagini iperspettrali complesse come quelle prese in esame.

Bibliografia

[1] «EarthExplorer.USGS.Gov,» USGS - United States Geological Survey, [Online]. Available: http://earthexplorer.usgs.gov/.

[2] «Landsat.USGS.Gov,» NASA Landsat Project - USGS - U.S. Geological Survey, [Online]. Available: http://landsat.usgs.gov/.

[3] «NASA.Gov.Gallery,» NASA - National Aeronautics and Space Administration, [Online]. Available: http://www.nasa.gov/.

[4] «Glovis.USGS.Gov,» USGS - EROS - Earth Resources Observation and Science Center, [Online]. Available: http://glovis.usgs.gov/.

[5] «SatImagingCorp.Gallery,» SIC - Satellite Imaging Corporation, [Online]. Available: http://www.satimagingcorp.com/gallery.html.

[6] «EyeFind.RapidEye,» RapidEye - Online Archive Discovery Tool, [Online]. Available: http://eyefind.rapideye.com/.

[7] «GeoEye.Gallery,» GeoEye Foundation, [Online]. Available: http://www.geoeye.com/CorpSite/gallery/.

[8] «ChangeMatters.Esri.Com,» ESRI - Environmental Systems Research Institute, [Online]. Available: http://changematters.esri.com/compare.

[9] «BioLab.Csr.Unibo,» Biometric System Laboratory - DEIS University of Bologna, [Online]. Available: http://biolab.csr.unibo.it.

[10] «GDAL,» GDAL - Geospatial Data Abstraction Library, [Online]. Available: http://www.gdal.org/.

[11] «Wikipedia "Programmazione orientata agli oggetti",» [Online]. Available: http://it.wikipedia.org/wiki/Programmazione_orientata_agli_oggetti.

[12] «Wikipedia "Polimorfismo",» [Online]. Available: http://it.wikipedia.org/wiki/Polimorfismo_(informatica).

[13] D. A. Landgrebe, Signal Theory Methods in Multispectral Remote Sensing, Wiley, 2003.

[14] S. Watanabe, Pattern Recognition: Human and Mechanical, New York: Wiley, 1985.

[15] M. P. A.K.Jain, Data Clustering: A Review, ACM Computing Surveys, 1999.

[16] A. K. Jain e R. C. Dubes, Algorithms for Clustering Data, Prentice Hall, 1988.

[17] J. C. Noordam, Chemometrics in multispectral imaging for quality inspection of postharverst products, Radboud University, Nijmegen, 2005.

[18] J. Bezdek, Pattern recognition with fuzzy objective function algorithms, New York: Plenum Press, 1981.

[19] W. Pedrycz, Pattern Recognition Letters - Conditional Fuzzy C-Means, Manitoba (Canada): Department of Electrical & Computer Engineering, University of Manitoba, 1996.

[20] J. C. Noordam e v. d. W. Broek, «Multivariate image segmentation based on geometrically guided fuzzy C-means clustering,» *Journal of Chemometrics,* pp. 1-11, 2000.

[21] «TLCLab.UniPv,» Telecommunications & Remote Sensing Laboratory - Dep. of Electronics - University of Pavia, [Online]. Available: http://tlclab.unipv.it/.

[22] P. Gamba, «A Collection of Data for Urban Area Characterization.,» in *Geoscience and Remote Sensing Symposium, 2004.(IGARSS '04)*, Anchorage(USA), 2004.

[23] W. M. Rand, «Objective criteria for the evaluation of clustering methods.,» *Journal of the American Statistical Association, Vol. 66, Nr. 336*, pp. 846-850, 1971.

[24] H. Xin e Z. Liangpei, «A comparative study of spatial approaches for urban mapping using hyperspectral ROSIS iamges over Pavia City, northern Italy,» in *International Journal of Remote Sensing*, Wuhan University, P.R. China, 2008.

NOTA: tutte le citazioni da testi in lingua straniera sono state effettuate traducendo o riassumendo nel modo semanticamente più fedele possibile.