Basi di dati per la gestione di immagini

Applicazioni tipiche

- Applicazioni mediche
  - raggi X, risonanza magnetica, tomografie computerizzate
- GIS (Geographical Information Systems)
  - immagini satellitari, mappe
- applicazioni di polizia
  - impronte
  - fotografie

Problematiche

- La realizzazione di un sistema per la gestione di immagini deve affrontare le seguenti problematiche
  - rappresentazione delle immagini
    - come si possono rappresentare le immagini tramite dati alfanumerici?
    - quali proprietà delle immagini devono essere rappresentate?
    - quali proprietà invarianti deve soddisfare la rappresentazione?
    - come è possibile ottenere (automaticamente o semi automaticamente) tale rappresentazione?
  - Misura della similarità
    - dato uno schema di rappresentazione, come dovrebbero essere confrontate due immagini?
    - quale misura deve essere utilizzata per determinare una similarità visiva?
  - Metodi di accesso e ritrovamento
    - quale metodo di indicizzazione dovrebbe essere usato per ritrovare efficientemente un'immagine nel database?

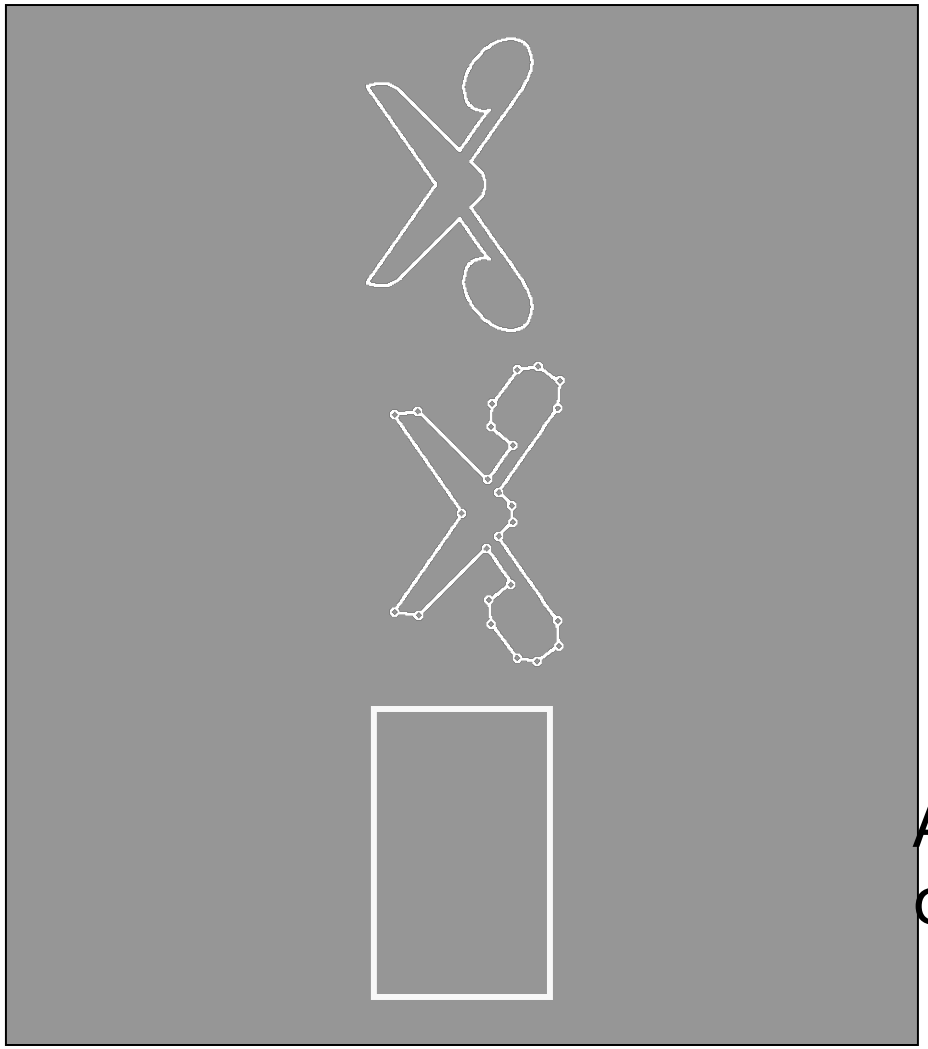
Rappresentazione

Quali features e attributi?

Features

- Forme (shape)
  - l'idea è quella di estrarre delle forme dall'immagine
  - possono rappresentare contorni o approssimazione di contorni
  - tipicamente ogni shape è rappresentata come combinazione di 16 shape primitive (linea, arco, ecc.), per ciascuna delle quali deve essere fornita la geometria (rappresentazione vettoriale)
    - alcune informazioni possono essere ridondanti ma rendere più efficiente le interrogazioni in alcuni casi
  - l'approssimazione di una shape limita la porzione di immagine di interesse

Features



contorno

Approssimazione
contorno

Features

- Oggetti semantici
  - con oggetto semantico si intende un oggetto significativo dal punto di vista dell'applicazione
    - casa, viso, strada
  - se gli oggetti sono chiaramente identificabili e possono essere facilmente riconosciuti, il ritrovamento può essere basato su tali oggetto
  - tipicamente richiedono intervento umano
  - approccio limitato dalle tecniche esistenti di analisi delle immagini

Features

- Texture (struttura)
  - rappresentano informazioni relative alla struttura dell'immagine
  - si basano sulle cosiddette Tamura features
    - concentrazione (scala)
    - contrasto
    - direzionalità
    - regolarità
    - rugosità
  - è stato dimostrato che solo le prime tre proprietà sono significative

Features

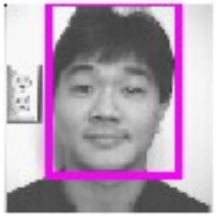
- Colore
  - proprietà globale che non richiede conoscenza degli oggetti contenuti nelle immagini
  - può essere determinato in modo automatico quindi è molto utilizzato
  - spesso si usano istogrammi che rappresentano la composizione di colori in un'immagine
  - nello spazio di colori RGB (Red, Green, Blue) l'istogramma è suddiviso in una serie di scomparti, ciascuno dei quali corrisponde ad un colore, ottenuto come combinazione di rosso, verde e blue
  - per rappresentare un'immagine tramite il colore, si associa ad ogni scomparto il numero di pixel dell'immagine con il colore considerato
  - si ha uno scomparto per ogni colore (es. 256)
  - quindi in generale il colore può essere rappresentato come un punto in uno spazio multidimensionale

Un possibile surrogato per le immagini

- In generale, per rappresentare in modo sufficientemente dettagliato un'immagine è necessario utilizzare più features
- In generale, ogni immagine può essere interpretata come
  - un insieme di oggetti interessanti, ciascuno caratterizzato da *un descrittore di forma*, che rappresenta la shape dell'oggetto e/o la zona dell'immagine nella quale l'oggetto è collocato (approssimazione)
  - *un descrittore di proprietà*, che descrive le proprietà di un insieme di pixel nell'immagine (esempio, RGB, livelli di grigio per immagini in bianco e nero, texture)
    - in genere non si associano proprietà singoli pixel per ovvi motivi computazionali
- le proprietà sono rappresentate da un nome (Red, Green, Blue) e da un dominio (ad esempio {0,…,8})

Esempio

pic1.gif



pic2.gif

pic3.gif

pic4.gif

pic5.gif

pic6.gif

pic7.gif

Esempio

- Si consideri l'immagine pic1.gif
- l'immagine contiene due oggetti di interesse - o1 e o2 -
  - le shape di questi oggetti sono rappresentate dai rettangoli presenti nella figura
  - il descrittore di proprietà associato ad un insieme di pixel potrebbe avere la seguente forma
    - Red = 5
    - Green = 1
    - Blue = 3

Shape

- Una shape per un oggetto può essere rappresentata in vari modi
- ad esempio, un contorno può sempre essere visto come una sequenza di punti p1,…,pn in uno spazio bidimensionale tali che:
  - ogni p_i, p_{i+1} rappresenta una porzione della shape
  - a ciascuna porzione vengono associate determinate proprietà, ad esempio, il tipo (linea, arco, ecc.) e la geometria
    - alcune informazioni possono essere ridondanti ma rendere più efficiente le interrogazioni in alcuni casi
  - nel caso più semplice, si suppone che tutte le porzioni siano linee e quindi i punti sono sufficienti a descrivere la geometria delle shape

Descrittori di proprietà

- Le proprietà vengono in genere associate ad insiemi di pixel
- come determinare questi insiemi?
- In genere, ogni immagine viene associata ad una coppia di interi positivi (m,n), chiamata la *griglia di risoluzione* dell'immagine
- una griglia (m,n) divide l'immagine in (m x n) celle di uguale dimensione, chiamate *griglia dell'immagine*
- ogni cella è costituita da un insieme di pixel
- le proprietà possono quindi essere associate a ciascuna cella
- ogni proprietà può essere interpretata come una tripla (Nome, Valore, Metodo) dove
  - Nome rappresenta il nome della proprietà (es. bwcolor per colore bianco/nero)
  - Valore rappresenta il range di valori per l'attributo
  - Metodo rappresenta l'identificazione di un algoritmo che permette di determinare la proprietà considerata

Osservazione

- Le shape si possono interpretare come proprietà globali dell'immagine
- le proprietà sono invece locali a ciascuna singola cella
- dalle proprietà di ogni singola cella si possono ovviamente inferire le proprietà delle celle che rappresentano i vari oggetti identificati

Esempio

- Consideriamo immagini in bianco e nero
- una possibile proprietà da associare ad una cella è

  (bwcolor, {b,w}, bwalgo}

  dove
  - bwcolor è il nome della proprietà
  - b e w sono i possibili valori che la proprietà può assumere
  - bwalgo identifica un algoritmo che prende una cella e restituisce b o w in relazione al numero di pixel bianchi o neri presenti nella cella
- consideriamo adesso un'immagine in livelli di grigio
- in questo caso, una possibile proprietà è

  (graylevel, [0,1], grayalgo}
  - in questo caso l'algoritmo dopo avere analizzato la cella restituisce un valore tra 0 e 1 che rappresenta il livello di grigio della cella

Definizione di base di dati di immagini

- Una base di dati di immagini è una tripla (GI, Prop, Rec) dove:
  - GI è un insieme di immagini a cui è stata associata una griglia
    - ogni immagine si può quindi vedere come una terna (Immagine,m,n), dove (m,n) rappresenta la griglia di risoluzione
  - Prop è un insieme di proprietà di celle
  - Rec è una funzione che associa ad ogni immagine un insieme di shape, in base ad una qualche rappresentazione
- Prop rappresenta le proprietà locali delle immagini
- Rec rappresenta le proprietà globali delle immagini

Problematiche

- Compressione
  - Come ridurre il numero di pixel di un’immagine per occupare meno spazio disco?
  - Come utilizzare questa rappresentazione nelle query?
- Segmentazione
  - come partizionare l’immagine in un insieme di regioni in modo che ogni regione sia omogenea rispetto a qualche proprietà?
- Interrogazioni
  - come determinare quando due immagini sono simili?

Compressione

- ogni immagine è composta da un insieme molto grande di pixel
- memorizzare un'immagine in questa forma può non essere conveniente dal punto di vista dello spazio occupato su disco
- si ha quindi la necessità di utilizzare degli algoritmi di compressione che permettano di ridurre il numero di pixel dell'immagine
- ogni algoritmo si può vedere come una funzione che prende un'immagine, restituisce la rappresentazione compressa e soddisfa le seguenti proprietà:
  - invertibilità:
    - deve esistere la funzione inversa alla funzione di compressione (si deve potere riottenere esattamente l'immagine di partenza)
  - preservazione distanze
    - volendo utilizzare la forma compressa nelle interrogazioni, la trasformazione deve preservare le distanze Euclidee dei pixel contenuti nell'immagine (vedi oltre)

Compressione

- Esempi di funzioni di trasformazione
  - DFT: Discrete Fourier Transform
    - non invertibile
    - viene sempre associata ad altre trasformazioni in modo che la combinazione di queste trasformazioni garantisca l'invertibilità
    - preserva le distanze
  - DCT: Discrete Cosine Transform
    - invertibile
    - preserva distanze

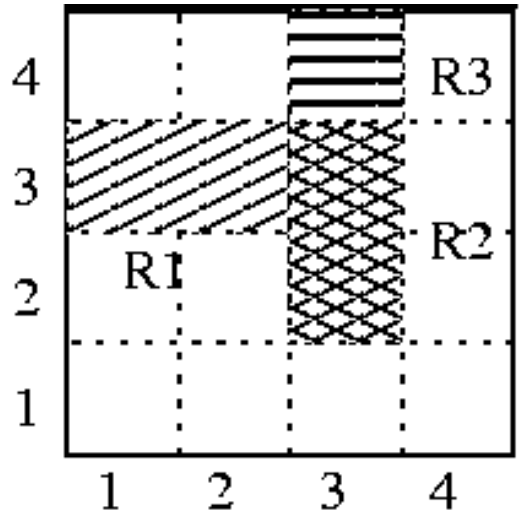
Segmentazione (informalmente)

- Informalmente, il problema è quello di prendere in input un'immagine e produrre in output un insieme di regioni *connesse* che *partizionano* l'immagine in modo che ogni regione sia *omogenea* rispetto ad una qualche proprietà
- questa operazione permette di associare proprietà da associare a gruppi di celle
- il retrieval può quindi basarsi non solo sulle proprietà di una singola cella ma anche sulle proprietà delle regioni

Segmentazione (formalmente)

- Supponiamo che l'immagine I contenga (m x n) celle
- Una regione per I è un sottoinsieme di celle
- Una regione R *connessa* è un insieme di celle tali che se (x1,y1), (x2,y2) $\in$ R allora esiste una sequenza di celle $C_1,\ldots,C_n$ in R tali che:
  - C_1 = (x1,y1)
  - C_n = (x2,y2)
  - la distanza Euclidea tra C_i e C_{ni+1} è 1, per qualunque 1 <= i < n

Esempio di regione connessa



- R1, R2, R3 sono regioni connesse
- (R1 U R2) è connessa
- (R2 U R3) è connessa
- (R1 U R2 U R3) è connessa
- (R1 U R3) non è connessa:

  R1 = (2,3) R3 = (3,4) d(R1,R3) >1

Segmentazione

- Un predicato di omogeneità associato ad un’immagine I è una funzione che prende in input una regione connessa R in I e restituisce vero o falso in relazione al fatto che la regione soddisfi una certa proprietà

Segmentazione

- Esempio
  - supponiamo che e sia un numero reale tra 0 e 1, inclusi, e supponiamo di considerare immagini in bianco e nero
  - un predicato di omogeneità può essere definito come segue
    - $H^{bw}_e(R)$ = vero se più del 100e % delle celle nella regione R hanno lo stesso colore
  - Supponiamo di considerare tre regioni. caratterizzate come segue:

Region	Num. of Black Pixels	Num. of White Pixels
R_1	800	200
R_2	900	100
R_3	100	900

  - Supponiamo di considerare tre diversi predicati: $H^{bw}_{0.8,}$ $H^{bw}_{0.89,}$ $H^{bw}_{0.92}$
  - la tabella seguente illustra i risultati di omogeneità per le regioni considerate

Region	$H^{bw}_{0.8}$	$H^{bw}_{0.89}$	$H^{bw}_{0.92}$
R_1	true	false	false
R_2	true	true	false
R_3	true	true	false

Segmentazione

- Data un'immagine I associata ad una griglia (m x n), definiamo segmentazione dell'immagine rispetto ad un predicato di omogeneità H come un insieme di regioni $R_1,\ldots,R_k$ tali che:
  - $R_i \cap R_j = \{\}$ per ogni i diverso da j, 1 <= i,j <= k
  - $I = R_i \cup \ldots \cup R_k$
  - $H(R_i)$ = vero per ogni 1 <= i <= k
  - se $R_i \cup R_j$ è una regione connessa, allora $H(R_i \cup R_j)$ = falso

Esempio

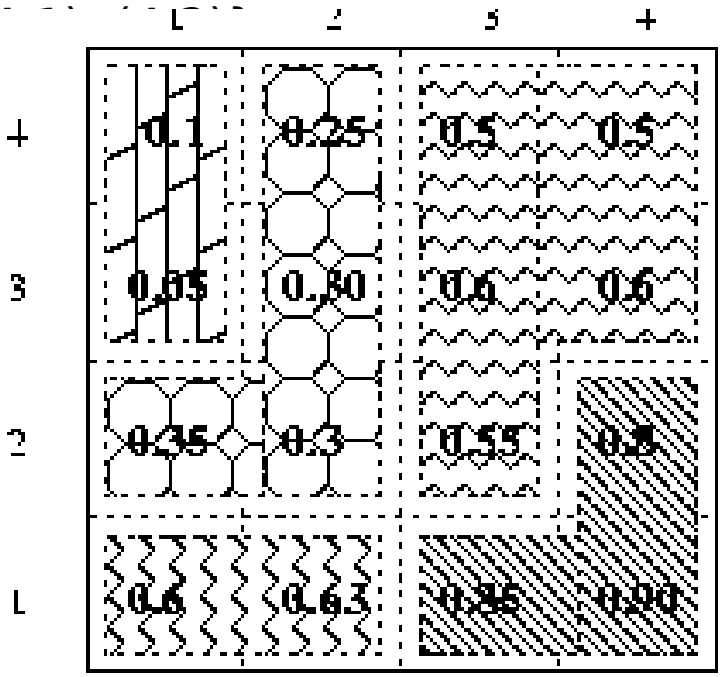
- Si consideri una griglia di dimensione (4 x 4) contenente i seguenti livelli di bianco/nero per ogni cella

Row/Col	1	2	3	4
1	0.1	0.25	0.5	0.5
2	0.05	0.30	0.6	0.6
3	0.35	0.30	0.55	0.8
4	0.6	0.65	0.85	0.90

- si consideri il predicato di omogeneità $H^{dyn}_{0.03}$ tale che presa una regione R restituisce vero se e solo se esiste un valore reale r tale che ogni cella nella regione ha un bwlevel v tale che |v-r| <= 0.03

Esempio

- Le seguenti 5 regioni rappresentano una valida segmentazione dell'immagine sottostante rispetto al predicato considerato
  - R1 = {(1,1),(1,2)}
  - R2 = {(1,3), (2,1), (2,2,), (2,3)}
  - R3 = {(3,1), (3,2), (3,3), (
  - R4 = {(3,4), (4,3), (4,4)}
  - R5 = {(1,4), (2,4)}



Algoritmo di segmentazione

- Input
  - immagine I da segmentare
- Due fasi
  - fase di split
    - se I è omogenea, restituiamo un'unica regione rapresentata dall'immagine stessa
    - altrimenti, dividiamo l'imagine in due parti e ripetiamo ricorsivamente questo processo, finché non troviamo un insieme di regioni R1,…,Rk omogenee che soddisfano tutte le condizioni della segmentazione, esclusa l'ultima
  - fase di merge
    - Siano R1,…,Rk le regioni restituite dalla fase precedente
    - verifichiamo quali regioni possono essere combinate
    - siano R'1,…,R'h le regioni finali
  - Output
    - R'1,…,R'h

Query per similitudine

- Le query possono coinvolgere sia proprietà locali che proprietà globali
- problema: come determino la similitudine tra due immagini?
- Approccio fondamentale: approccio metrico
  - si assume che esista una distanza metrica d con la quale confrontare ogni coppia di oggetti
  - più bassa è la distanza, più simili sono gli oggetti

Approccio metrico

- Una funzione da un insieme X a [0,1] è una funzione distanza se soddisfa i seguenti assiomi per ogni x,y,z in X:
  - d(x,y) = d(y,x)
  - d(x,y) <= d(x,z) + d(z,y)
  - d(x,x) = 0
- per ogni proprietà associata ad un'immagine può essere definita una funzione distanza
- la distanza tra due immagini si ottiene quindi combinando le distanze tra le varie proprietà (locali e/o globali), ad esempio applicando la distanza Euclidea

Esempio applicato alle proprietà globali

- Si consideri un insieme Obj di immagini (256 x 256). Si supponga che ogni cella sia associata a tre attributi (red, green, blue) che assumono un valore in {0,...,8}

- un esempio di funzioni distanza tra due immagini o1 e o2 rispetto alle tre proprietà considerate è il seguente:
  - diffr[i,j] = (o1[i,j].red - o2[i,j].red)^2
  - diffg[i,j] = (o1[i,j].green - o2[i,j].green) ^{2}
  - diffb[i,j] = (o1[i,j].blue - o2[i,j].blue) ^{2}
  - d(o1,o2) = ($\sum_{i=1,256}$ $\sum_{i=1,256}$ (diffr[i,j] + diffg[i,j] + diffb[i,j]))$^{1/2}$

Approccio metrico per proprietà locali

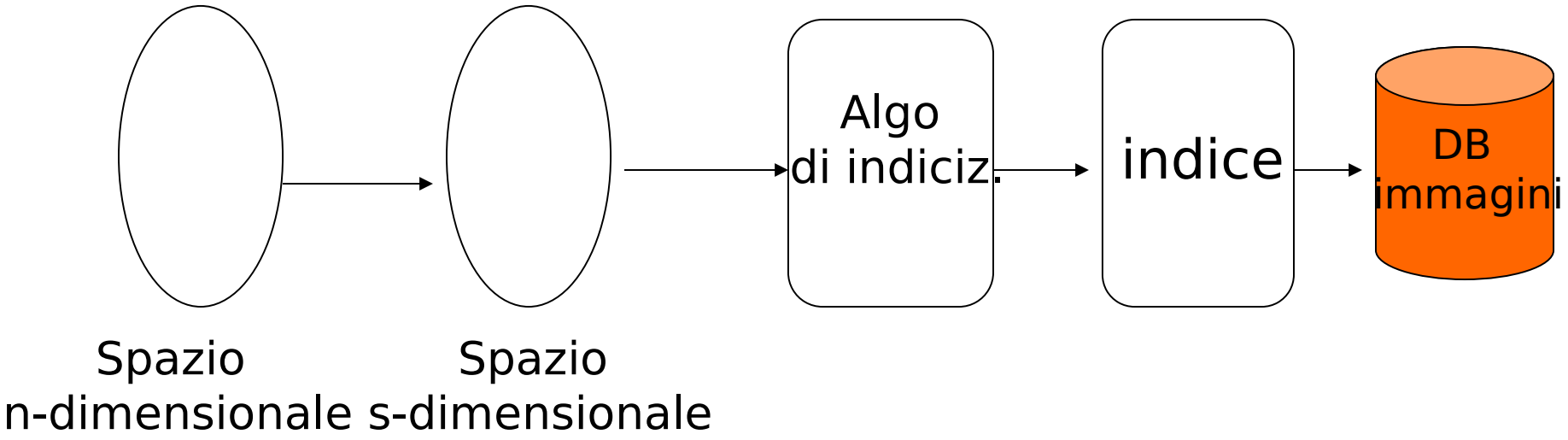
- La distanza precedente rappresenta la distanza Euclidea tra i due oggetti
- se si considerano n proprietà locali e t =h x k celle, in base all'approccio metrico ogni immagine è vista come un insieme di m punti n-dimensionali
  - f(I) = {P1,...,Pt} Pi = (xi1,...,xin)
- per determinare se le immagini sono simili in base alla distanza Euclidea:
  - f(I1) = {P1,...,Pt} Pi = (xi1,...,xin)
  - f(I2) = {Q1,...,Qt} Qi = (yi1,...,yin)
  - $d(I1,I2) = (\sum_{i=1,t} d'(Pi,Qi))^{1/2}$
  - $d'(Pi,Qi) = (xi1 - yi1)^2 + \ldots + (xin - yin)^2$

Approccio metrico per proprietà locali

- L’approccio metrico richiede che per ogni proprietà locale venga calcolata la distanza tra celle corrispondenti in immagini distinte
- ogni immagine è quindi interpretata come un insieme di punti
  - ogni punto è dato dai valori per le proprietà locali di una certa cella
  - se le proprietà sono n, i punti sono n-dimensionali
  - ogni immagine sarà caratterizzata da tanti punti quante sono le celle
- poiché il numero delle celle spesso è alto esistono approcci (che non vediamo) per mappare le immagini NON in un insieme di punti MA in un singolo punto in uno spazio s-dimensionale, con s <= n
- Esempio: consideriamo la media degli n punti
  - in questo caso s coincide con n
- si noti che il punto ottenuto si può interpretare come una proprietà globale dell’immagine

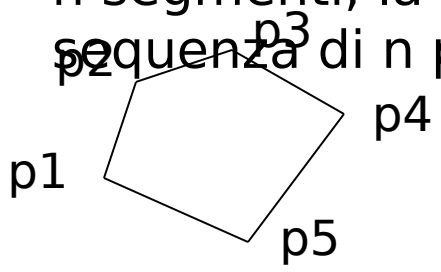
Indicizzazione per proprietà locali

- I punti ottenuti possono essere utilizzati come base per definire le tecniche di indicizzazione
  - nel caso limite in cui s = 1, si potrebbero usare I B-tree
  - nel caso in cui s > 1, è necessario utilizzare specifici indici multidimensionali
  - questi indici sono stati definiti nel contesto delle basi di dati spaziali



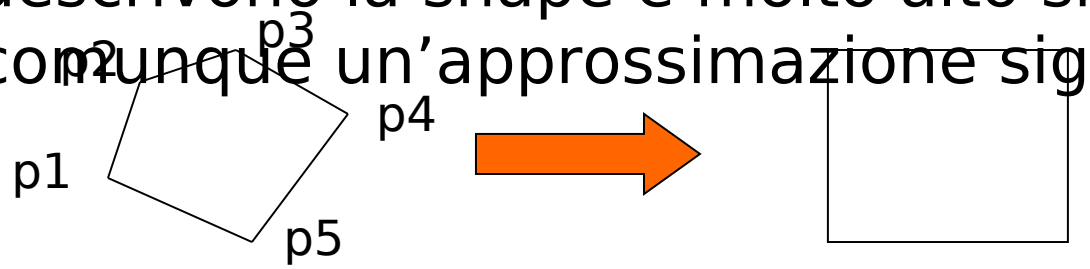
Approccio metrico per proprietà globali

- Anche per gli oggetti (regioni) associati ad un'immagine è possibile applicare un approccio metrico
- in questo caso la distanza deve essere definita sulle shape
- anche le shape possono essere rappresentate come un insieme di punti
- Esempio
  - se la shape è rappresentata da una spezzata costituita da n segmenti, la spezzata può essere descritta da una sequenza di n punti bidimensionali

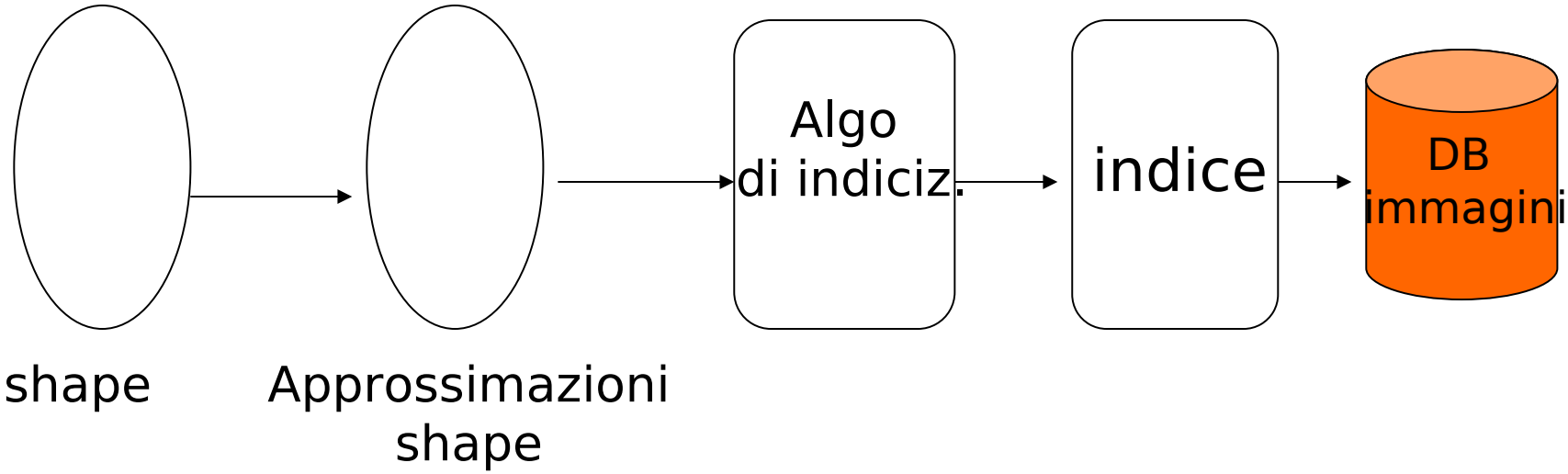


Approccio metrico per proprietà globali

- In questo caso è possibile passare da n punti a 2 punti considerando il minimo rettangolo che contiene la shape (minimum bounding box - MBB)
- 2 punti si possono vedere come un punto in uno spazio 4-dimensionale (s = 4)
- in questo caso quindi s = 4 è maggiore di n=2 ma poichè in genere il numero di punti che descrivono la shape è molto alto si ritiene comunque un'approssimazione significativa



Indicizzazione per shape



Notazione

- Spesso I punti nello spazio s-dimensionale vengono chiamati *feature* mentre l'insieme dei punti da cui si parte si definisce *surrogato*

Uso della distanza nelle query

- L’utilizzo della funzione distanza permette di risolvere in modo formalmente semplice (ma non semplice dal punto di vista di un’ottimizzazione efficiente) le query per similitudine
- quattro tipi di query fondamentali
  - match completo preciso
  - match parziale preciso
  - match completo impreciso
  - march parziale impreciso
- completo/totale si riferisce agli oggetti che considero nella query
  - intere immagini
  - oggetti
- preciso/impreciso si riferisce a quante immagini voglio ritrovare rispetto alla distanza

Match preciso

- Completo
  - “trova l’immagine più simile all’immagine I”
  - è necessario avere generato i surrogati per le immagini nel database e per l’immagine I
  - per ogni immagine I’ nel database si calcola d(I,I’)
  - si restituisce I’ che minimizza d(I,I’)
  - in pratica si trova l’immagine “più vicina” ad I
  - sono necessari indici per determinare l’oggetto più simile, cioè più vicino all’oggetto dato (nearest neighbor query)

Match preciso

- Parziale
  - “trova l’immagini che contiene l’oggetto più simile all’immagine I”
  - è necessario avere generato i surrogati per le immagini nel database e per l’immagine I
  - per ogni immagine I’ nel database si considerano gli oggetti/regioni o riconosciuti in I e si calcola d(I,o)
  - si restituisce l’immagine I’ contenente un oggetto o che minimizza d(I,o)

Match impreciso

- completo
  - “trova le immagini più simili all’immagine I rispetto ad una tolleranza e data”
  - in questo caso l’input è rappresentato da I e anche da una tolleranza di errore e
  - è necessario avere generato i surrogati per le immagini nel database e per l’immagine I
  - per ogni immagine I’ nel database si calcola d(I,I’)
  - si restituiscono tutte le I’ tali che d(I,I’) <= e

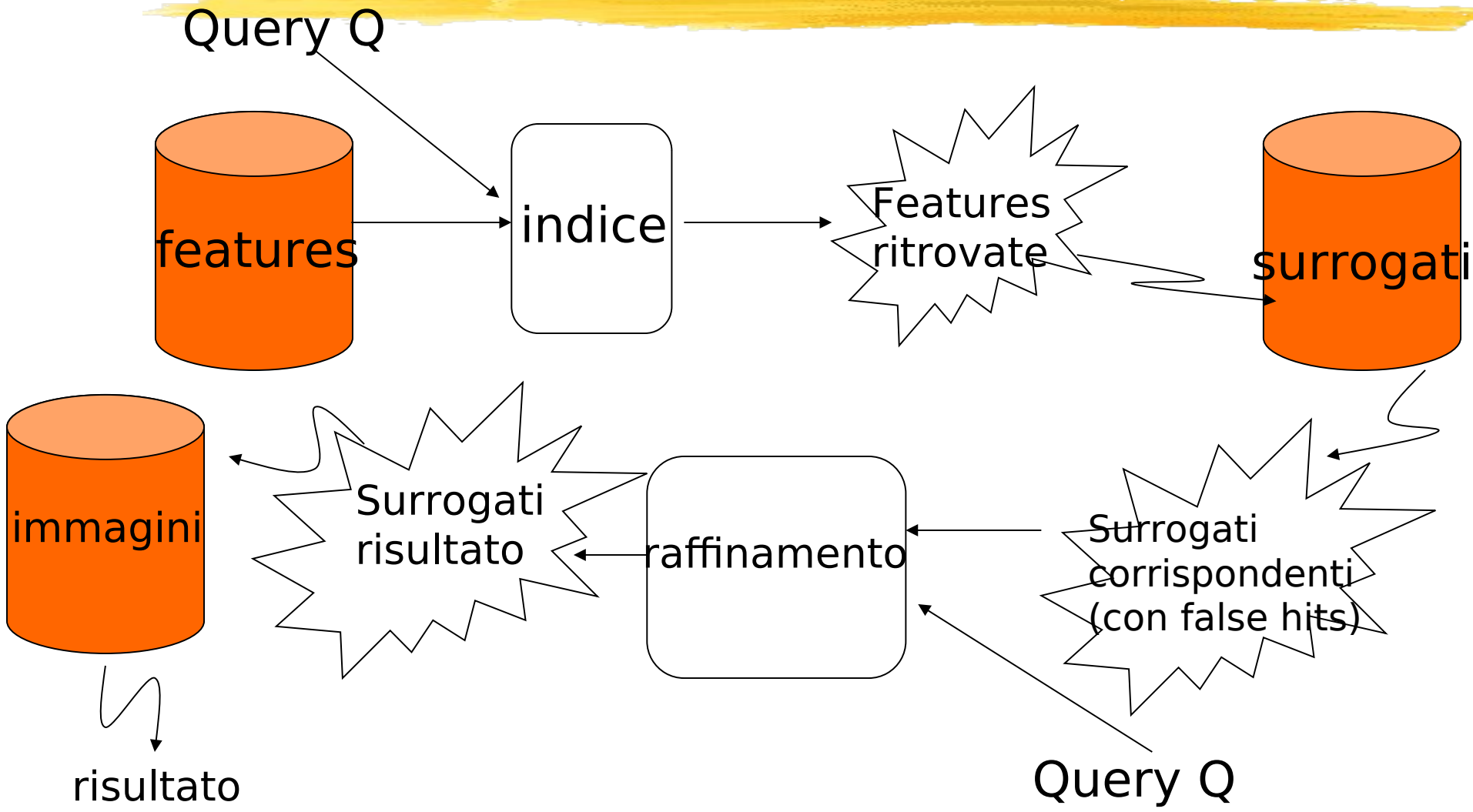
Match impreciso

- Parziale
  - “trova le immagini che contengono oggetti simili all’immagine I, rispetto ad una tolleranza e”
  - in questo caso l’input è rappresentato da I e anche da una tolleranza di errore e
  - è necessario avere generato i surrogati per le immagini nel database e per l’immagine I
  - per ogni immagine I’ nel database si considerano gli oggetti/regioni o riconosciuti in I e si calcola d(I,o)
  - si restituiscono tutte le immagini I’ contenenti un oggetto o tale che d(I,o) <= e

Esecuzione query

- Nel contesto relazionale, i B-tree data una selezione permettevano di trovare tutte le tuple che la verificavano
- nel contesto multimediali questo non accade
- cioè, data una condizione di selezione e un indice è possibile selezionare in genere solo un soprainsieme delle features che soddisfano la query (*fase di filtering*)
- questo perché le feature approssimano il surrogato associato ad una immagine
- è poi necessario considerare i surrogati associati alle feature ottenute per scartare quelli che non soddisfano la query, applicando direttamente la nozione di distanza considerata (*fase di raffinamento*)
  - gli oggetti (in questo caso, surrogati) scartati vengono chiamati *false hits*

Esecuzione query



Approccio trasformazionale

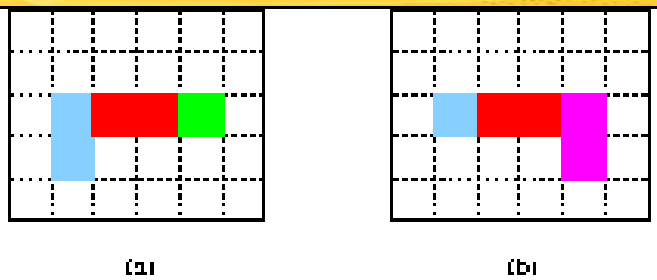
- Oltre all'approccio metrico esiste un altro approccio per gestire query per similitudine chiamato approccio trasformazionale
- Si basa sul principio per cui dati due oggetti o1 e o2, il livello di dissimilarità tra o1 e o2 è proporzionale al costo minimo di trasformare l'oggetto o1 nell'oggetto o2 o viceversa
- la trasformazione si ottiene combinando un insieme di operatori di trasformazione di base:
  - traslazioni
  - rotazioni
  - scalatura
  - incisioni (taglia parte di un'immagine)

Approccio trasformazionale

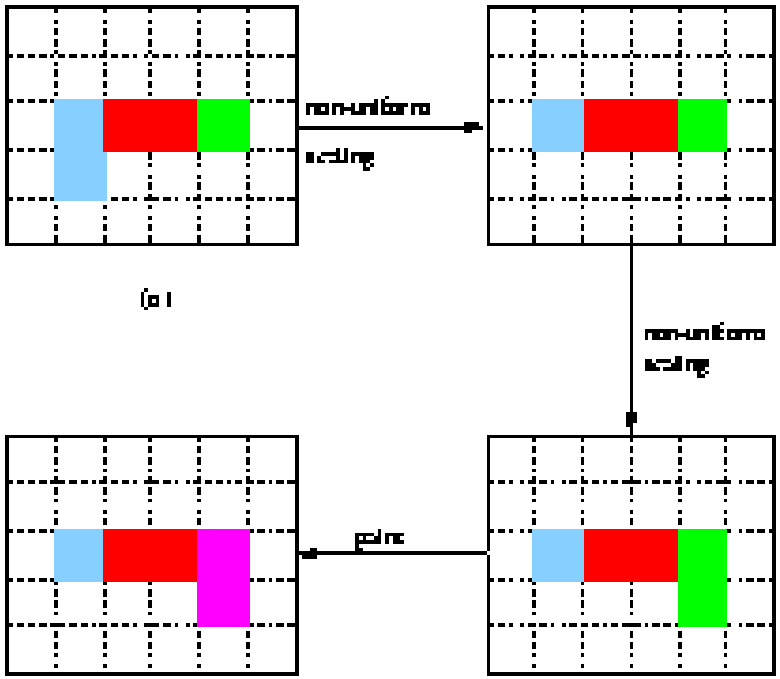
- La trasformazione di un oggetto o in un oggetto o’ è una sequenza di operazioni di trasformazione $to_1,\ldots,to_{r,}\ to_{r+1}$ e una sequenza di oggetti $o_1,\ldots,o_r$ tale che:
  - $to_1\ (o) = o_1$
  - $to_i(o_{i-1}) = o_i$
  - $to_{r+1}(o_r) = o'$
- il costo della sequenza di trasformazione TS è dato da
  - $cost(TS) = \sum_{i=1,r} cost(to_i)$
- data la sequenza di trasformazione TS, la dissimilarità tra o e o’, indicata con dis(o,o’) rispetto agli operatori di trasformazione considerati e all’insieme di funzioni di costo fissate è data da:
  - dis(o,o’) = min {cost(TS) | TS in TSeq(o,o’) U TS in TSeq(o’,o)}

dove Tseq(o,o’) indica tutte le possibili trasformazioni che convertono o in o’

Esempio



- Per passare dall'oggetto a) all'oggetto b), si può applicare la seguente trasformazione:



Confronto

- Vantaggi del modello trasformazionale sul modello metrico
  - la nozione di similarità dipende dagli operatori di trasformazione, che possono essere definiti dall'utente
  - le funzioni di costo possono essere definite dall'utente
- Vantaggi del modello metrico sul modello trasformazionale
  - avendo una sola nozione di similarità l'indicizzazione dei dati è semplificata mentre nel modello trasformazionale dipende dalle scelte dell'utente e non sempre è possibile indicizzare

Rappresentazione immagini nei DBMS

- La gestione delle immagini nei vari DBMS dipende dal sistema prescelto
- in genere, viene permessa una duplice rappresentazione
  - riferimento esterno:
    - il DB contiene riferimento ai file che contengono le immagini
  - rappresentazione interna:
    - le immagini vengono memorizzati nel DBMS di tipo BLOB (binary large object)
- i sistemi mettono quindi a disposizione tool per generare i surrogati delle immagini (il tipo di surrogato dipende dal DBMS e dal tool prescelto),per gestire le immagini e per interrogarle

Rappresentazione immagini nei DBMS

- Quindi, per rappresentare un database di immagini IDB=(GI, Prop, Rec) in un DBMS
  - ogni I in GI viene memorizzata in base all'approccio prescelto
  - ogni proprietà in Prop verrà implementata dai tool messi a disposizione dal sistema per determinare i valori associati a specifiche proprietà (che dipendono dal sistema scelto)
  - Rec rappresenta le shape degli oggetti riconosciuti utilizzando i tool messi a disposizione del sistema
- le proprietà possono riferirsi alle celle o all'immagine nel suo complesso e rappresentare, ad esempio, informazioni quali il nome dell'immagine, la data di caricamento, ecc.
- queste informazioni possono essere rese disponibili all'utente per effettuare interrogazioni
- Nel caso dell'approccio metrico, il sistema supporterà adeguati indici multidimensionali per supportare le query per similitudine

Gestione immagini in Oracle 8i

Due approcci

- Le immagini in Oracle 8i possono essere gestite in due modi diversi:
  - usando interMedia
  - usando Oracle Visual Information Retrieval

Uso di interMedia

- Prodotto fornito con Oracle8i che permette di memorizzare, ritrovare, manipolare e gestire testi, immagini, video e audio
- le immagini sorgente possono essere memorizzate in campi BLOB o BFILE
- in interMedia le immagini (ma anche gli altri media) vengono rappresentate utilizzando specifici tipi oggetto
  - nel caso delle immagini: Tipo ORDImage
- un'istanza del tipo ORDImage è costituita da:
  - attributi, che rappresentano il dato sorgente, cioè l'immagine, e alcune proprietà globali associate all'immagine tra cui:
    - lunghezza, larghezza, dimensione
    - tipo file (es. TIFF)
    - tipo di compressione (es. JPEG)
    - tipo di contenuto (es. Monocromatico)
  - metodi, che permettono di gestire le immagini

Uso di interMedia

- Se le immagini sorgenti sono memorizzate come BLOB, vengono gestite dal DBMS
- gli attributi e i metodi associati alle immagini vengono invece gestiti da interMedia
  - generati
  - ritrovati
- interMedia si può quindi vedere come un “building block” sul quale costruire applicazioni piuttosto che un’applicazione utente
- Limitazione:
  - non supporta ricerche complesse per similarità ma soltanto ricerche basate sul valore degli attributi
  - non permette di indicizzare le immagini rispetto a qualche feature
- Esempio di query supportata: determina le immagini di dimensione > 32

Oracle Visual Information Retrieval

- Estensione di Oracle 8i al supporto della rappresentazione, memorizzazione, indicizzazione, ritrovamento, manipolazione di immagini
- permette di eseguire query di similarità
- anche in questo caso le immagini sono rappresentate come istanze di un certo tipo oggetto che estende il tipo ORDImage supportato da interMedia con un attributo *signature*
- questo attributo rappresenta la feature sulla quale basare le query sul contenuto ed eventualmente l'indicizzazione
- questa tecnologia è stata sviluppata da Virage Inc., specializzata nel ritrovamento per similarità
- è prevista l'estensione del prodotto con nuovi attributi da associare alle immagini ma viene garantita la compatibilità con le versioni precedenti

Tipi oggetto

```
CREATE TYPE ORDVIR AS OBJECT
(
    -- TYPE ATTRIBUTES
            image ORDImage,
            signature RAW(2000)

    -- METHOD DECLARATION

    -- CONSTRUCTORS
        STATIC FUNCTION init( ) RETURN ORDVir,
        STATIC FUNCTION init(srcType IN VARCHAR2,srcLocation IN
          VARCHAR2, srcName IN VARCHAR2) RETURN ORDVir,
```

Tipi oggetto

-- IMAGE COPY METHOD
MEMBER PROCEDURE copy(dest IN OUT ORDVir),
-- IMAGE PROCESSING RELATED METHODS
MEMBER PROCEDURE process(SELF IN OUT ORDVir, command IN VARCHAR2),
...
-- IMAGE PROPERTY SET AND CHECK METHODS
MEMBER PROCEDURE setProperties(SELF IN OUT ORDVir),
....
-- IMAGE ATTRIBUTE ACCESSOR METHODS
MEMBER FUNCTION getHeight RETURN INTEGER,
....
--SOURCE/CONTENT RELATED METHODS
MEMBER PROCEDURE importFrom(ctx IN OUT RAW, source_type IN VARCHAR2, source_name IN VARCHAR2)
MEMBER PROCEDURE export(ctx IN OUT RAW, source_type IN VARCHAR2, source_name IN VARCHAR2)

Attributi

Table 4–1 ORDImage Object Type

Attribute	Data Type	Description
source	ORDSource	Points to the source of the stored image data
height	INTEGER	Height of the image in pixels
width	INTEGER	Width of the image in pixels
contentLength	INTEGER	Size of the on-disk image file in bytes
fileFormat	VARCHAR2(4000)	File type of image (such as, TIFF or JFIF)
contentFormat	VARCHAR2(4000)	Type of image (such as, monochrome or 8-bit grayscale
compressionFormat	VARCHAR2(4000)	Compression type of the image
mimeType	VARCHAR2(4000)	MIME type of the image

Attributi

Table 4–2 ORDSource Object Type

Attribute	Data Type	Description
localData	BLOB	Locally stored image data.
srcType	VARCHAR2(4000)	Identifies the source type as one of the following: ▪ NULL ▪ 'FILE' -- source is an external file ▪ 'HTTP' -- HTTP server name ▪ '<name>' -- name of another source
srcLocation	VARCHAR2(4000)	Identifies where the data can be found based on the srcType. See Table 4–3.
srcName	VARCHAR2(4000)	Identifies the data object name based on the source type. See Table 4–4.
updateTime	DATE	The time at which the data was last updated.
local	NUMBER	Flag to determine if the data is local (1) or not (0).

Attributi

Table 4–3 srcLocation Values

srcType	srcLocation
NULL	NULL or not accessed
FILE	<DIR> or name of the directory object

Table 4–3 srcLocation Values (Cont.)

srcType	srcLocation
HTTP	<SourceBase> or URL needed to find the base directory
<name>	<iden> or identifier string required to access another server

Table 4–4 Valid srcName Values

srcType	srcName
NULL	NULL or not accessed
FILE	<FILE> or name of the file
HTTP	<Source> or name of the object
<name>	<object name> or name of the object

Esempio

- Si supponga di volere creare una tabella contenente delle fotografie
- in particolare la tabella deve contenere:
  - id fotografia
  - nome fotografo
  - descrizione fotografia
  - la fotografia
- una possibile dichiarazione di questa tabella è la seguente:

  - CREATE TABLE stockphotos (photo_id NUMBER, photographer VARCHAR(64), annotation VARCHAR(255), photo ORDSYS.ORDVir);

Costruttori

- Init() return ORDVir
- inizializza l'oggetto ponendo tutti gli attributi a NULL esclusi:
  - source.localdata=empty_blob()
    - empty_blob(): inizializza un campo di tipo BLOB
  - source.local = 1
  - source.updateTime = SYSDATE
- Esempio
  - insert into stockphotos (1, "John Ross", NULL, ORDSYS.OrdVir.init());

Costruttori

- Init(srcType IN VARCHAR2,srcLocation IN VARCHAR2, srcName IN VARCHAR2) RETURN ORDVir,
- inizializza l'oggetto ponendo tutti gli attributi a NULL esclusi:
  - source.localdata=empty_blob()
  - source.local = 1
  - source.updateTime = SYSDATE
  - source.srcType, source.srcLocation, srcName, posti uguali ai valori in input
- Esempio
  - insert into stockphotos (1, "John Ross", NULL, ORDSYS.OrdVir.init('FILE','VIRDIR','image1.gif'));

Caricamento

- I costruttori effettuano solo l'inizializzazione degli attributi ma non caricano l'immagine nel campo BLOB
- sono previsti metodi per il caricamento da un file o l'esportazione
- vediamo solo un caso di import
- MEMBER PROCEDURE importFrom(ctx IN OUT RAW, source_type IN VARCHAR2, source_name IN VARCHAR2)
- RAW: dati che non vengono interpretati da Oracle
  - simile a LOB, ma con varie restrizioni a livello di gestione

Esempio

```
    Image ORDSYS.ORDVir;
    ctx   RAW(4000):=NULL;
BEGIN
    -- select the image to be imported
    SELECT photo INTO Image FROM stockphotos
           WHERE photo_id = 1 FOR UPDATE;

    -- import the image into the database
    Image.importFrom(ctx,
                     'FILE',
                     'ORDVIRDIR',
                     'virdemo1.dat');
```

Metodi per la manipolazione

- Si dividono in diversi gruppi tra cui:
  - metodi di copia
  - metodi di processamento
  - metodi di settaggio attributi
  - metodi di accesso agli attributi (circa uno per attributo)
- supportano manipolazioni di base, basate sull'uso degli attributi

Metodi di manipolazione

- Metodo di copia:
  - Copy (dest IN OUT ORDVir)
  - copia l'immagine in dest, inclusi i valori di tutti gli attributi
- metodi di processamento:
  - process(command IN VARCHAR2)
  - command rappresenta una lista di tecniche di image processing da applicare alle immagini

Metodi di manipolazione

- Metodi di settaggio attributi:
  - setproperties()
  - determina i valori per i seguenti attributi:
    - height,width,file type, image type, compression type
- Metodi di accesso:
  - circa uno per ogni attributi (si veda il manuale)
  - getHeight() RETURN INTEGER
  - restituisce il valore associato all'attributo HEIGHT

Operatori per visual information retrieval

- Supportano tre operazioni fondamentali:
  - analyze: per generare la feature associata ad un'immagine (attributo signature)
  - VIRsimilar: prese due immagini, stabilisce se sono simili, confrontando le segnature
  - VIRscore: restituisce il valore di similarità per due immagini
- la funzione analyze è definita nel package VIR
  - ORDSYS.VIR.analyze
  - ORDSYS.VIRScore
  - ORDSYS.VIRSimilar

Analyze

- Analyze(Image IN OUT ORDVir)
- crea la feature da associare all'immagine Image
- la feature viene inserita nel campo signature
- NOTA: non setta le altre proprietà, è necessario usare setproperties

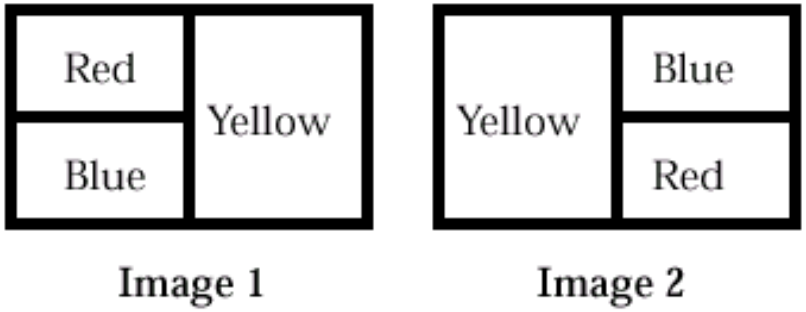
La segnatura

- La segnatura contiene informazioni sulle seguenti proprietà:
  - *colore globale*: distribuzione di colore nell'intera immagine
    - considera solo quanti pixel ci sono per ogni colore e non la loro distribuzione
  - *colore locale*: rappresenta la distribuzione di colore nell'immagine, tenendo in considerazione la posizione in cui il colore compare
  - *texture*: struttura
  - *structure*: rappresenta gli oggetti (tipicamente forme geometriche), può considerare anche la posizione e la dimensione
    - ottenute ad esempio per segmentazione
- I valori associati alle precedenti proprietà rappresentano un punto multidimensionale
- la somma di tali valori per un'immagine è 100

La segnatura: esempi

- Image1 e Image2 sono simili rispetto al colore globale ma non rispetto al colore locale

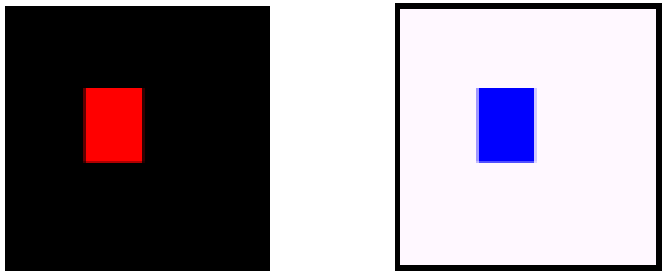
Figure 2–1 Image Comparison: Global Color and Local Color



La segnatura: esempi

Figure 2–4 Fabric Images with Similar Texture

Figure 2–5 Images with Very Similar Structure



Similar

- VIRSimilar(signature IN RAW,
  querysignature IN RAW,
  weightstring IN VARCHAR2,
  threshold in FLOAT,
  [referencetoScore IN NUMBER])
- weightstring ha la forma
  - ‘globalcolor = “val” localcolor = “val” texture = “val” structure = “val”
- confronta signature e query_signature
- calcola la media delle distanze di ogni proprietà considerata, pesandola rispetto ai pesi specificati in weightstring (che devono essere compresi tra 0 e 1)
- restituisce 1 se la distanza è minore di threshold (valore tra 0 e 100)
- referencetoScore assegna un identificatore alla chiamata della funzione
- potrà poi essere utilizzato nella chiamata della funzione VIRScore, nel contesto della stessa query

Score

- VIRScore(referencetoSimilar IN NUMBER)
  - la query che invoca questo metodo deve anche invocare VIRSimilar
    - posso calcolare lo scoro solo se ho determinato quali oggetti sono simili
  - referencetoSimilar è un intero che rappresenta l'identificatore di una chiamata a VIRSimilar
  - lo score viene calcolato rispetto agli oggetti sui quali è stata applicata VIRSimilar

Esempio

- Supponiamo di volere determinare tutte le fotografie simile ad una fotografia F data
- nel determinare la similitudine, vogliano attribuire un peso maggiore al local color, in particolare vogliamo fissare i pesi come segue:
  - global color = 0.1
  - local color = 0.6
  - texture = 0.2
  - structure = 0.1
- vogliano inoltre fissare un threshold a 50
- vogliamo restituire gli identificatori delle fotografie che soddisfano la query insieme allo score calcolato

Esempio di match completo impreciso

- Supponendo di considerare una variabile
  - F ORDVir
- ecco la query:

```
SELECT photo_id,ORDSYS.VIRScore(12)
FROM stockphotos
WHERE

ORDSYS.VIRSimilar(photo.signature,F.signature,
        ‘globalcolor = “0.1” localcolor =“0.6” texture
= “0.2”                 structure = “0.1” , 50, 12) = 1
```

Esempio

- Si consideri adesso un'immagine con i seguenti valori per gli attributi visuali:
  - globalcolor = 15
  - localcolor = 90
  - texture = 5
  - structure = 50
- la distanza è calcolata come segue:
  - 0.1 * 15 + 0.6 * 90 + 0.2 * 5 + 0.1 * 50 = 61.5
- poiché il threshold è 50, questa immagine non verrebbe restituita
- supponiamo adesso di invertire i valori associati al globalcolor e localcolor, la distanza diventa:
  - 0.6 * 15 + 0.1 * 90 + 0.2 * 5 + 0.1 * 50 = 24.0
- in questo caso l'immagine verrebbe restituita

Esempio di matching completo preciso

- Supponiamo di volere determinare la fotografica più simile a F

```
SELECT photo_id
FROM
(SELECT photo_id,ORDSYS.VIRScore(12),
    RANK() OVER (ORDER BY ORDSYS.VIRScore(12) ASC) AS rank;
FROM stockphotos
WHERE
ORDSYS.VIRSimilar(photo.signature,query.signature,
‘globalcolor = “0.1” localcolor =“0.6” texture = “0.2”    structure = “0.1” ,
50, 12) = 1)
WHERE rank = 1;
```

Quindi…

- Giocando sui pesi, le immagini possono essere confrontate in modo diverso
  - pesi maggiori aumentano l’importanza di un certo attributo
- valori di threshold più alti riducono la precisione
  - si possono generare molti false hits

Indici

- La creazione degli indici non è immediata
- il tipo dell'indice che deve essere creato è ORDSYS.ORDViridx
- Esempio: per creare un indice sull'attributo photo della tabella stockphotograph:

```
CREATE INDEX imgindex on stockphotograph(photo.signature)
INDEXTYPE IS ORDSYS.ORDViridx;
```

In conclusione …

- In Oracle 8i un'immagine è rappresentata secondo tre componenti:
  - immagine stessa (BLOB o BFILE)
    - se la si vuole trattare come BFILE, l'operazione di inizializzazione initFrom() è sufficiente ad inizializzare l'immagine
    - se la si vuole trattare come BLOB, è necessario prima inizializzarla e poi importarla (init() + import()-importFrom())
  - attributi descrittivi
    - determinati con metodo setProperties()
  - signatura
    - calcolata con metodo Analyze()