Graphs for image analysis and image understanding

Isabelle Bloch

Sorbonne Université, CNRS, LIP6
isabelle.bloch@sorbonne-universite.fr

and Florence Tupin (LTCI, Télécom Paris)

2025

Outline

1. Definitions and representation models
2. Representing image content by a graph for:
   - Segmentation or labeling
   - Object recognition
3. Graph matching
   - Graph or subgraph isomorphisms
   - Error tolerant graph-matching
   - Approximate algorithms (inexact matching)

A useful reference:
O. Lezoray, L. Grady - Image Processing and Analysis with Graphs: Theory and Practice - CRC Press, 2012.

Why using graphs?

- Usefulness: compact, structured and complete representation, easy to handle
- Applications:
  - Image analysis: segmentation, boundary detection
  - Image understanding and pattern recognition: printed characters, objects (buildings, brain structures...), faces...
  - Image registration
  - Understanding of structured scenes
  - ...

Shape and spatial relations

1. Definitions and representations models

Definitions

$$\textit{Graph}: \quad G = (X, E)$$

- X set of vertices ($|X|$ = order of the graph)
- E set of edges ($|E|$ = size of the graph), $E \subseteq X \times X$
- complete graph (size $\frac{n(n-1)}{2}$)
- partial graph $G = (X, E')$ with $E' \subseteq E$
- subgraph $F = (Y, E')$, $Y \subseteq X$ and $E' \subseteq E$
- degree of vertex x: $d(x)$ = number of edges having x as a vertex
- connected graph: any two vertices are linked by a chain of edges
- tree: connected graph without cycles
- clique: complete sub-graph
- dual graph: exchange the roles of vertices and edges
- hypergraph (n-ary relations: a hyperedge connects any number of vertices)
- weighted graphs: weights on the edges

Adjacency and Laplacian matrices

- Weight of an edge linking vertices i and j: w_{ij}
- Adjacency matrix W (of size $|X| \times |X|$):

$$W_{ij} = \begin{cases} w_{ij} & \text{if } e_{ij} = (i,j) \in E \\ 0 & \text{otherwise} \end{cases}$$

  for undirected edges W is symmetric
- Laplacian matrix of an undirected graph:

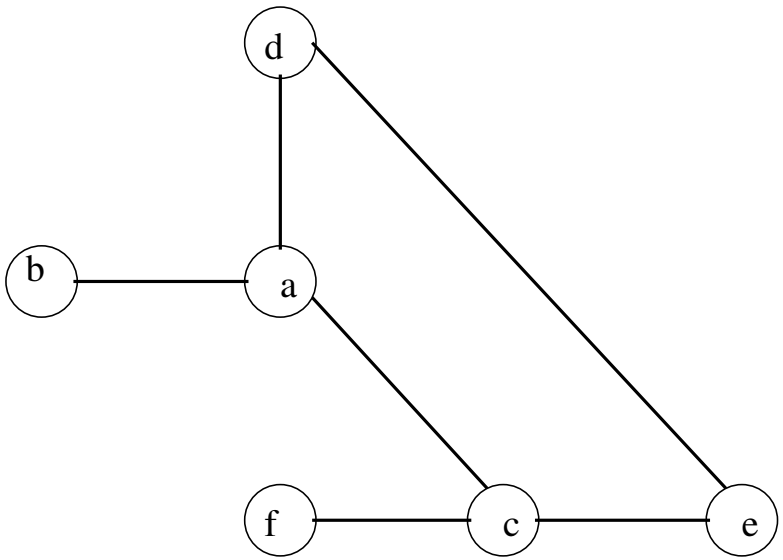
$$d_i = \sum_{e_{ij} \in E} w_{ij}$$

$$L_{ij} = \begin{cases} d_i & \text{if } i = j \\ -w_{ij} & \text{if } e_{ij} \in E \\ 0 & \text{otherwise} \end{cases}$$

$$\boxed{\text{L= D - W}}$$

  with $D_{ii} = d_i$ (D degree matrix)

Example



	a	b	c	d	e	f
a	0	1	1	1	0	0
b	1	0	0	0	0	0
c	1	0	0	0	1	1
d	1	0	0	0	1	0
e	0	0	1	1	0	0
f	0	0	1	0	0	0

Notion of complexity

Upper bound of the number of elementary operations needed by an algorithm to obtain a result.

- polynomial, exponential...
- NP-complete: decision problem whose solutions can be *verified* in polynomial time (NP), but there is no known way to *find* a solution quickly (NP-hard).

Examples of graphs

- Relational attributed graph (RAG): $G = (X, E, \mu, \nu)$
  - $\mu : X \to L_X$ vertex interpreter (L_X = attributes or vertices)
  - $\nu : E \to L_E$ edge interpreter (L_E = attributes of edges)

  Examples:
  - graph of pixels, with basic neighborhood relations
  - region adjacency graph
  - Voronoï diagram / Delaunay triangulation
  - graph of primitives with more complex relations
- Random graph: edges and vertices are random variables
- Fuzzy graph: $G = (X, E = X \times X, \mu_f, \nu_f)$
  - $\mu_f : X \to [0, 1]$
  - $\nu_f : E \to [0, 1]$
  - with $\forall (u, v) \in X \times X,\ \nu_f(u, v) \leq \mu_f(u)\mu_f(v)$ or $\nu_f(u, v) \leq \min[\mu_f(u)\mu_f(v)]$

- Fuzzy attributed graph: RAG with attributes defined as fuzzy values
- Hierarchical graph:
  multi-level graph and bi-partite graph between two levels (multi-level approaches, object grouping...)

  Examples :
  - quadtrees, octrees
  - hierarchical representation of the brain
- Graph for reasoning:
  decision tree, matching graph...
- Aspect graph

Some classical algorithms on graphs

Minimum spanning tree

- Kruskal algorithm: $O(n^2 + mlog_2(m))$ $(n = |E|, m = |X|)$
- Prim algorithm: $O(n^2)$

Shortest path

- positive weights: Dijkstra algorithm: $O(n^2)$
- arbitrary weights and no cycle: Bellman algorithm $O(n^2)$

Max flow and min cut

- $G = (X, E)$
- partition in two subsets A and B $(A \cup B = X,\ A \cap B = \emptyset)$
- $cut(A, B) = \sum_{x \in A, y \in B} w(x, y)$
- Ford and Fulkerson algorithm

Maximal cliques

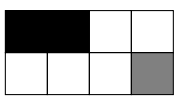
- decision tree
- cut of already explored branches

2. Representing image content as a graph

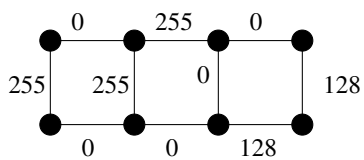
Segmentation by minimum spanning tree

Constantinidès (1986)

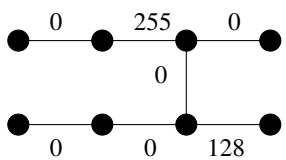
- graph of pixels weighted by the gray levels (or colors) (weights = distances)
- search of the minimum spanning tree
- spanning tree $\Rightarrow$ partitioning by suppressing the most costly edges



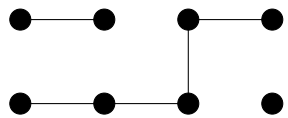
image



graphe des pixels attribué



arbre couvrant de poids minimal



suppression des arêtes
les plus coûteuses

Computation of the minimum spanning tree

Kruskal algorithm

- Starting from a partial graph without any edge, iterate $(n-1)$ times: choose the edge of minimum weight creating no cycle in the graph with the previously chosen edges.
- In practice:
  1. sort edges by increasing weights
  2. while the number of edges is less than $(n-1)$ do:
     - ⋆ select the first edge not already examined
     - ⋆ if cycle, reject
     - ⋆ otherwise, add the edge in the graph
- Complexity: $O(n^2 + m \log_2 m)$

Prim algorithm

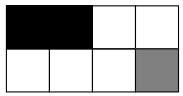
- Progressive extension of the current tree
- Complexity: $O(n^2)$

Segmentation by graph cut

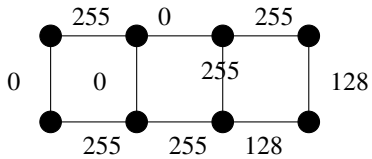
Wu et Leahy (1993)

- Graph of pixels, edge weighted by a similarity function $w(x, y)$
- Segmentation: partition of the graph
- Principle: search for the best partition, minimizing the cut
  - partition into A and B ($A \cup B = X$, $A \cap B = \emptyset$)
  - $cut(A, B) = \sum_{x \in A, y \in B} w(x, y)$
- Example: Ford and Fulkerson algorithm: augmenting a chain to increase the flow along the chain (decrease the cut)

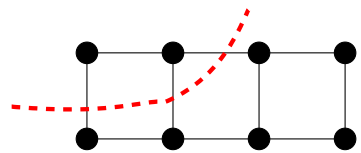
A simple example



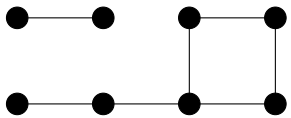
image



graphe des pixels attribué

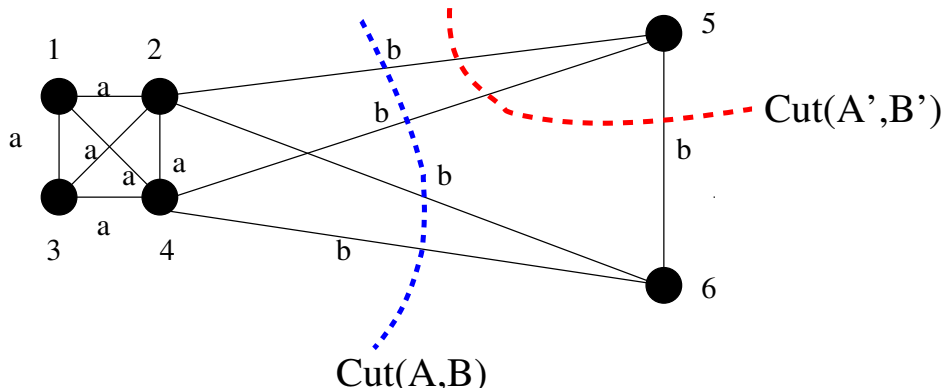


coupe de capacité minimale



partition

Influence of the number of vertices in the cut: $Cut(A, B) = 4b$, $Cut(A', B') = 3b$



$\Rightarrow$ normalized cut

Segmentation using normalized cut

Shi and Malik (2000)

$$Ncut(A,B) = \frac{cut(A,B)}{assoc(A,X)} + \frac{cut(A,B)}{assoc(B,X)}$$

$$assoc(A,X) = \sum_{a \in A, x \in X} w(a,x) = vol(A)$$

Measure of the connectivity withing a cluster:

$$Nassoc(A,B) = \frac{assoc(A,A)}{assoc(A,X)} + \frac{assoc(B,B)}{assoc(B,X)}$$

$$Ncut(A,B) = 2 - Nassoc(A,B)$$

minimizing the cut $\Leftrightarrow$ maximizing group connectivity

- NP complete problem
- Optimization:
  $x = (x_i)$ with $x_i = 1$ if $i \in A$, $x_i = -1$ otherwise
  $D = diag(d_i)$ with $d_i = \sum_j w(i,j)$
  $W = (w_{ij})$

$$\min_x Ncut(x) = \min_y \frac{y^T(D-W)y}{y^T Dy}$$

with $y(i) \in \{1, -b\}$ $(b = \frac{\sum_{x_i > 0} d_i}{\sum_{x_i < 0} d_i})$

Solution from eigenvectors and eigenvalues:

$$(D-W)y = \lambda y \quad \text{or} \quad D^{-\frac{1}{2}}(D-W)D^{-\frac{1}{2}}z = \lambda z$$

$(z = D^{-\frac{1}{2}}y)$

Spectral graph clustering

Von Luxburg (2007)

- Compute the k eigenvectors u_k of L associated with the k smallest eigenvalues.
- Matrix $U = (u_1, ...u_k)$, each row is a k dimensional embedding of vertex i.
- Clustering using k-means based on embeddings.

Modularity graph clustering

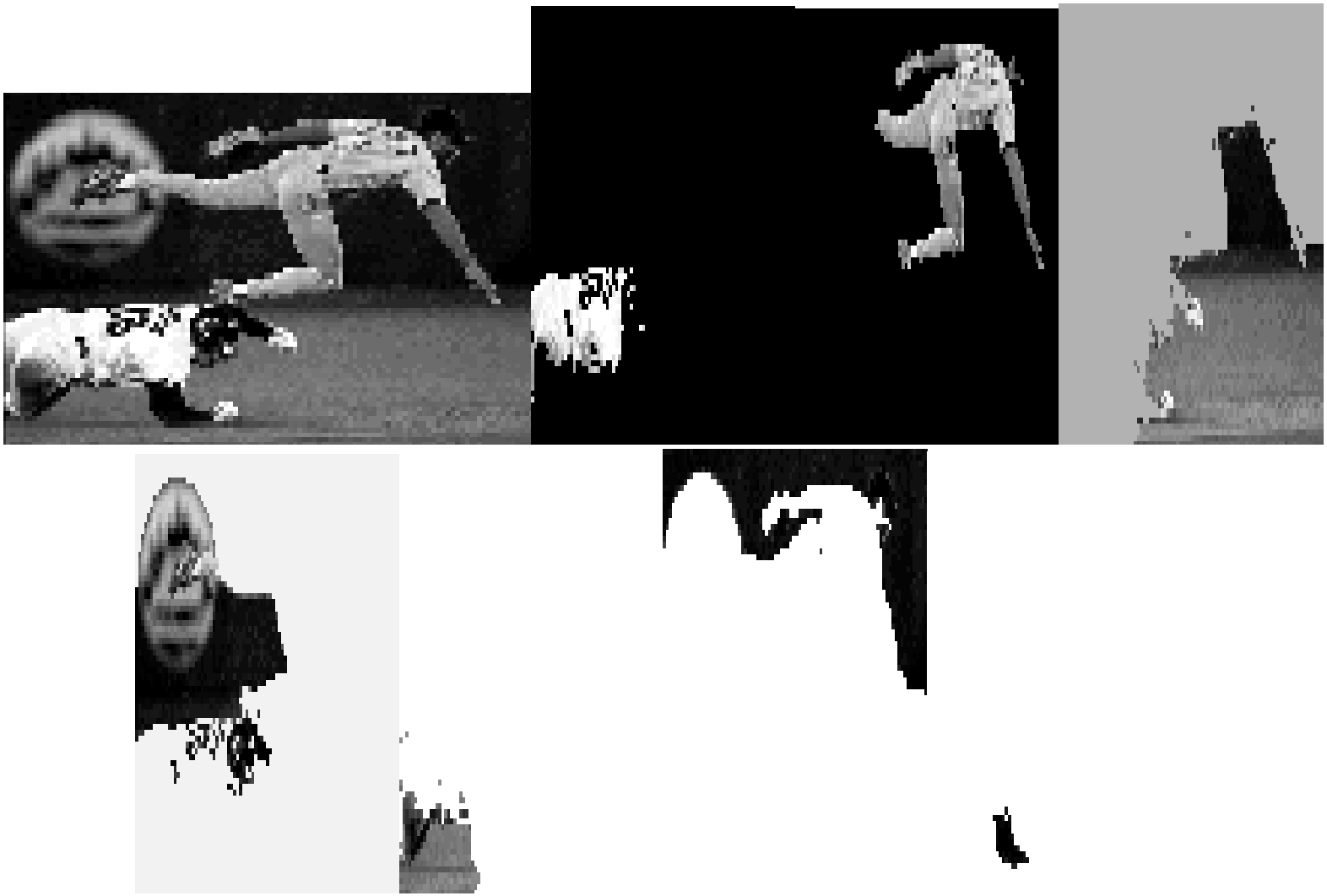
Newman (2006)

- Maximize a quality function:

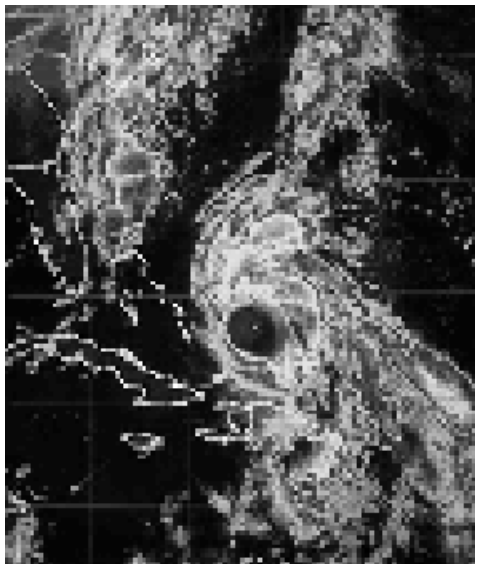
$$Q = \frac{1}{2|E|} \sum_{i=1}^{|X|} \sum_{j=1}^{|X|} \left(w_{ij} - \frac{d_i d_j}{2|E|} \right) \delta(C_i, C_j)$$

  with C_i cluster to which vertex i belongs,
  $\delta(C_i, C_j) = 1$ if $C_i = C_j$, 0 otherwise
- Interpretation: maximize the intra-cluster connections and minimize the inter-cluster connections.
- Greedy optimization (e.g. Louvain algorithm).

Examples (univ. Berkeley)



http://www.cs.berkeley.edu/projects/vision/Grouping/







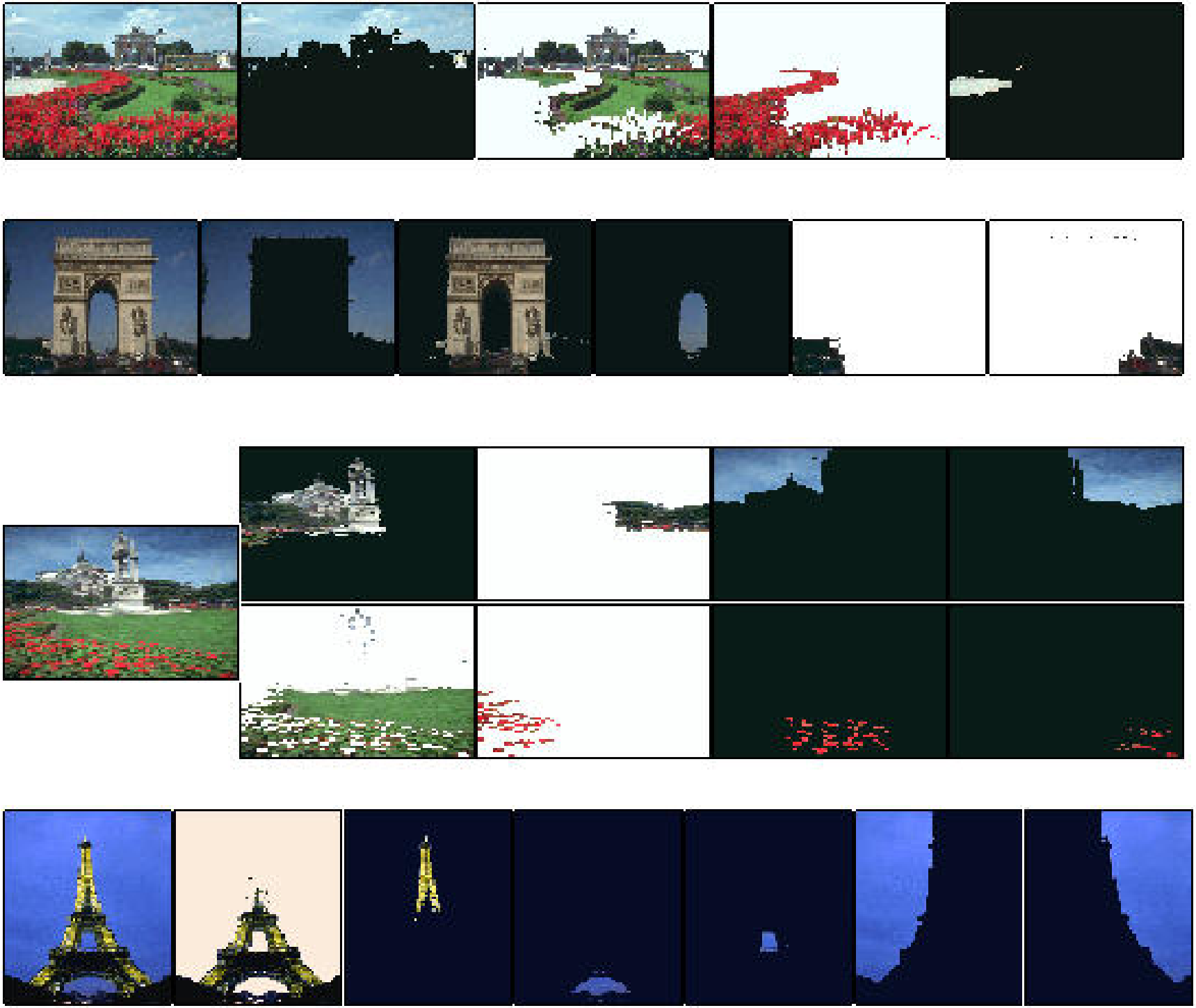


Image understanding

Tenenbaum and Barrow (1977)

- Segmentation into regions
- Building a region adjacency graph
- Labeling using a set of rules (expert system):
  1. on objects (size, color, texture...)
  2. on contextual relationships between objects (above, inside, near...)

Generalization using fuzzy attributed graphs

Markovian labeling (random graphs)

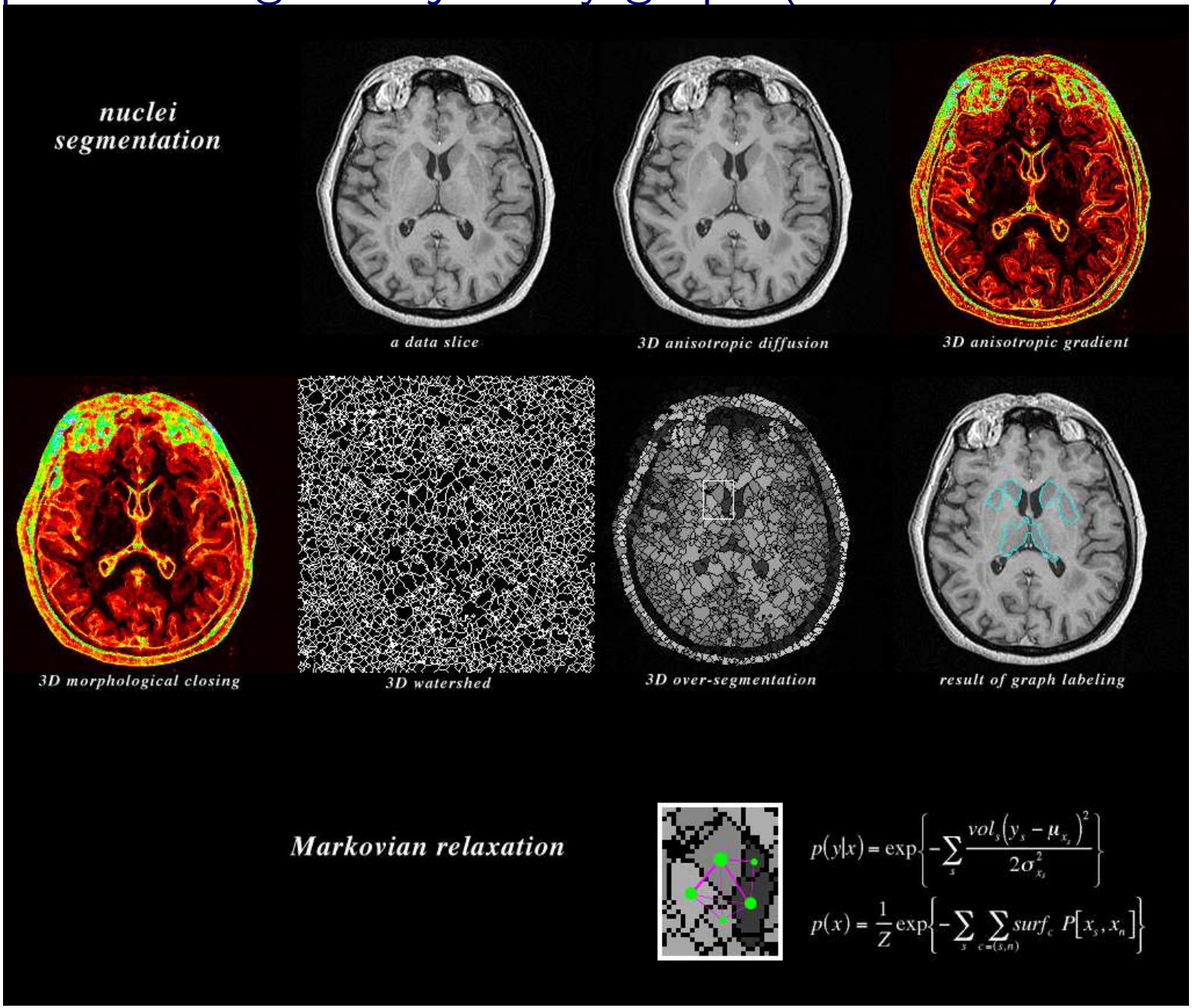
$$U(x \mid y) = U(y \mid x) + U(x) = U(y \mid x) + \sum_{c \in \mathcal{C}} V_c(x)$$

$x = \textit{labeling}, y = \textit{data}$

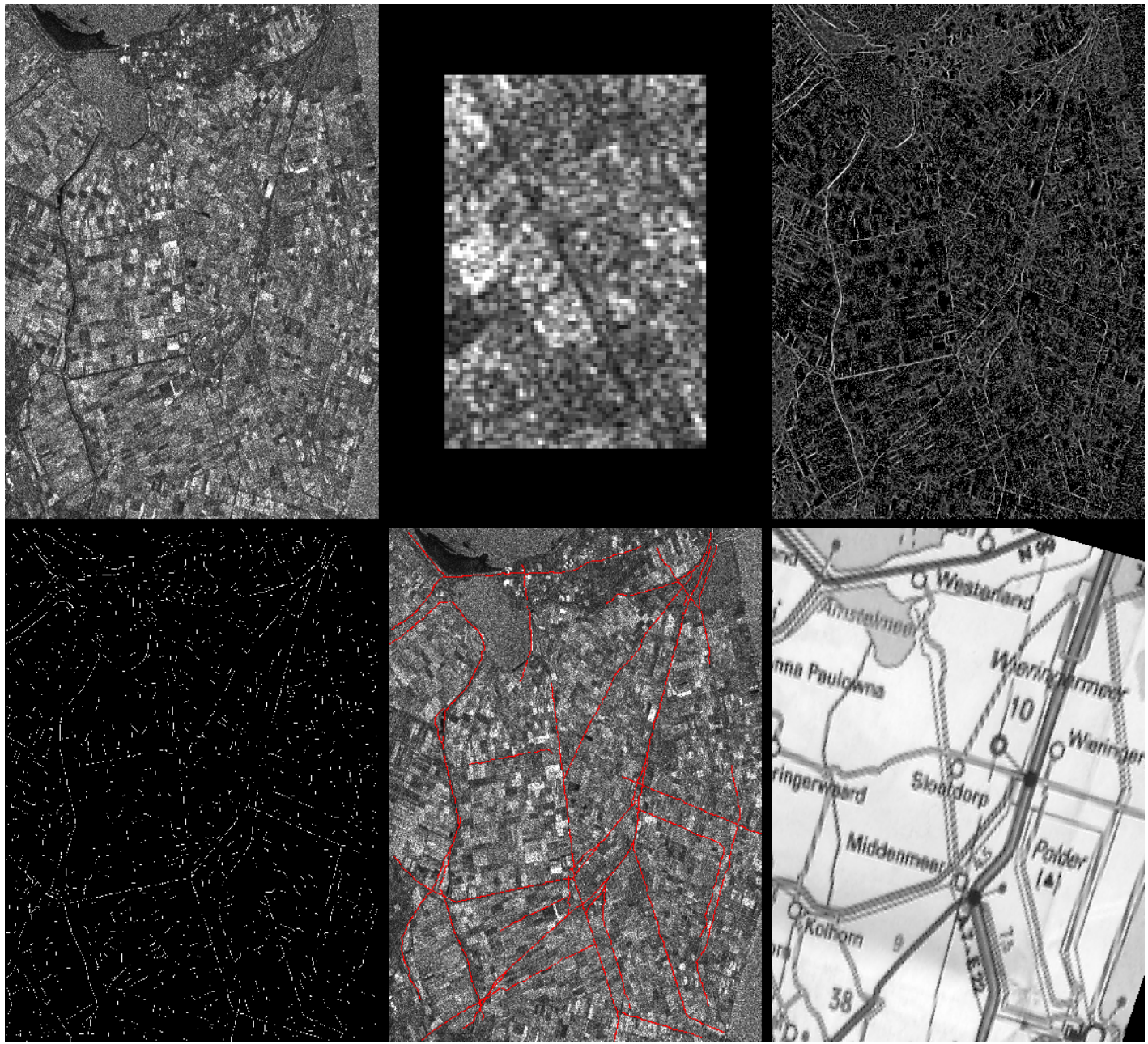
- Low-level applications:
  - pixel graphs
  - segmentation, classification, restoration
- High-level applications:
  - graph of super-pixels (SLIC, watershed...)
  - graph of primitives (edges, key-points, lines...)

$\Rightarrow$ pattern recognition, full scene labeling, scene understanding

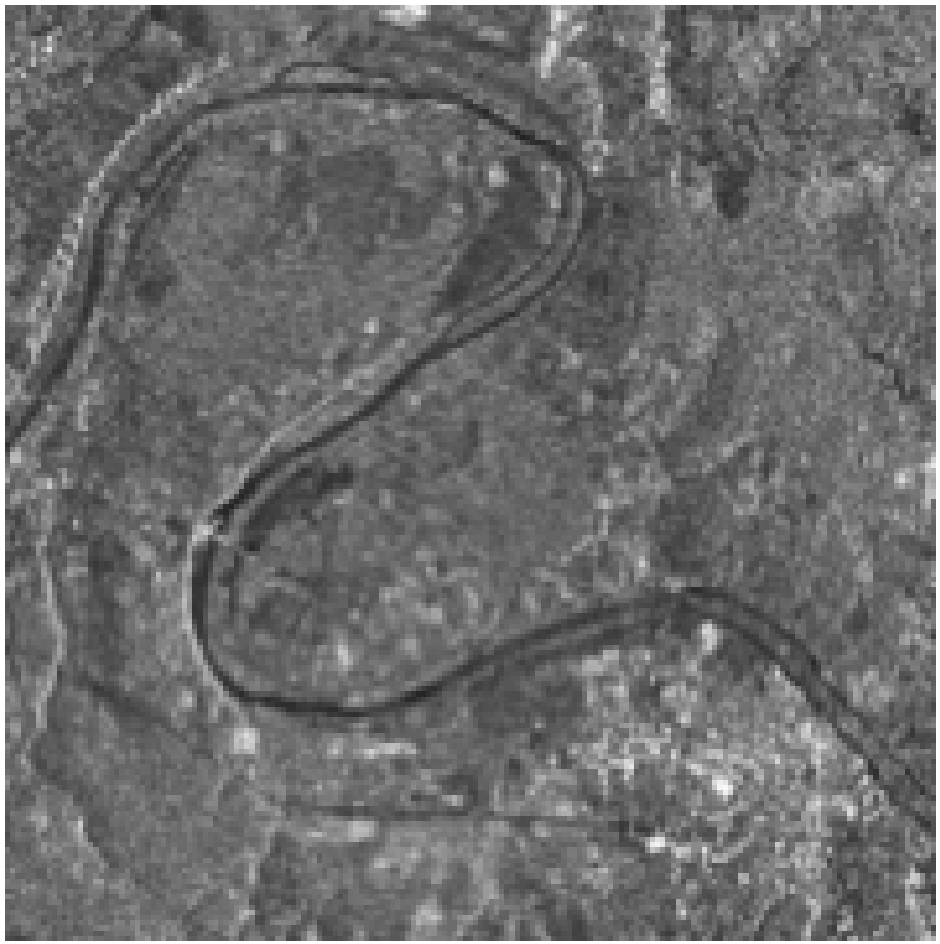
Example on a region adjacency graph (T. Géraud)

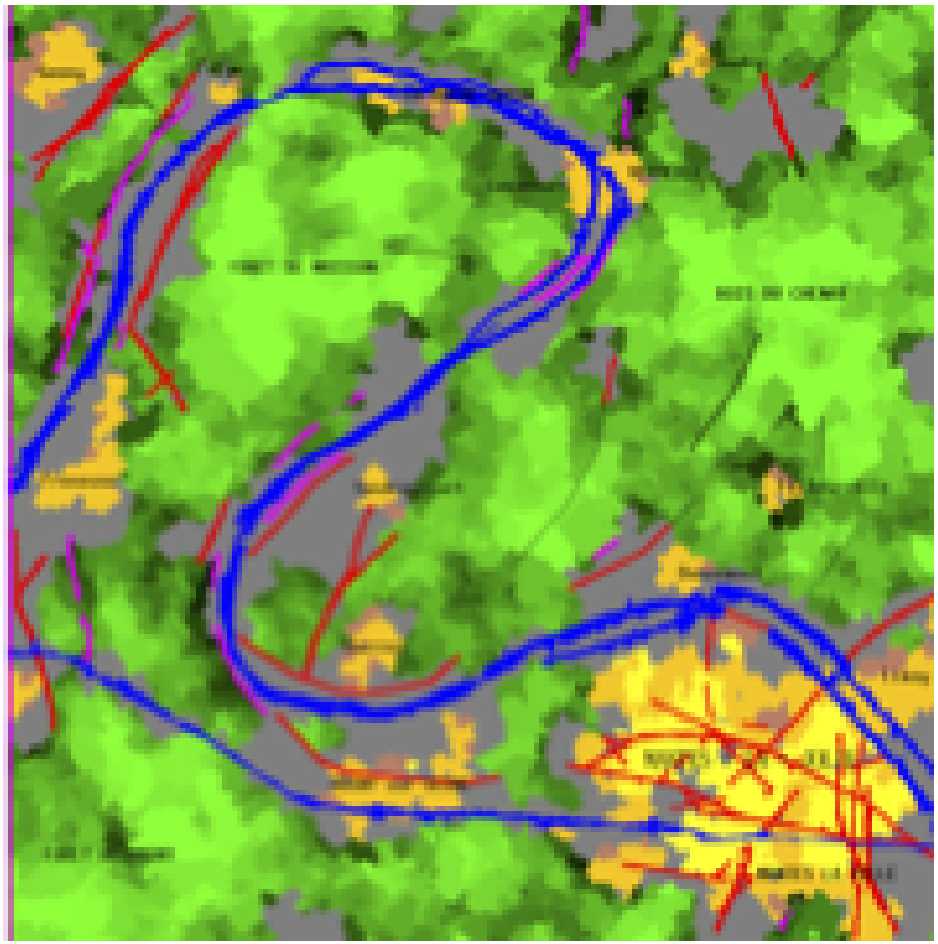


Example on a line graph (F. Tupin)



Example on a region adjacency graph (F. Tupin)

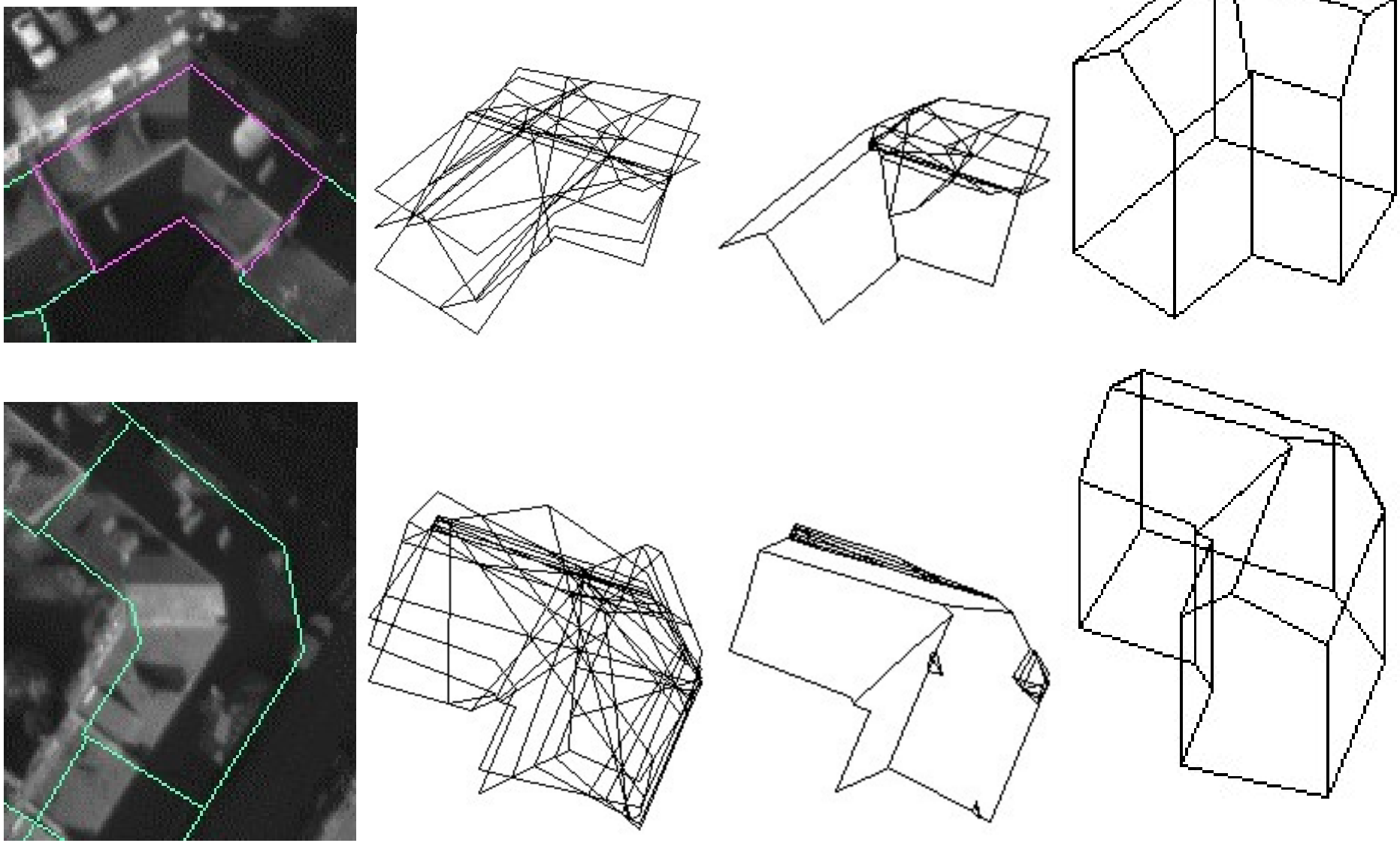




Object recognition

- Object: defined by a set of primitives (nodes of the graph)
- Binary relationship representing compatibility between primitives (edges of the graph)
- Clique: sub-set of primitives all mutually compatible
  = possible object configuration
- Recognition by maximal clique detection:
  - NP-hard problem
  - Building a decision tree: a node of the tree = a clique of the graph
  - Pruning the tree to suppress already found cliques
  - Theorem: let S be a vertex of the search tree T, and let x be the first unexplored child of S (to be explored). If all the sub-trees of $S \cup \{x\}$ have been generated, only the children S not adjacent to x have to be explored.

Example: building detection and reconstruction using maximal cliques (IGN)



3. Graph matching

Graph matching

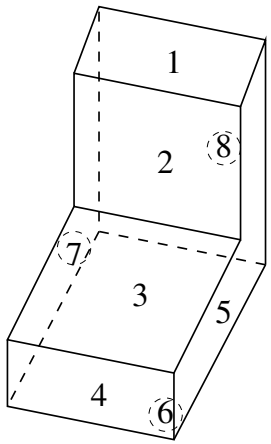
- Model graph (atlas, map, object model...) G
- Image graph built from data G'
- Matching the two graphs:

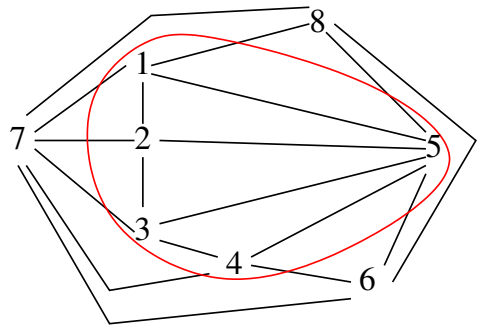
$$G = (X, E, \mu, \nu) \quad \rightarrow \quad G' = (X', E', \mu', \nu')$$

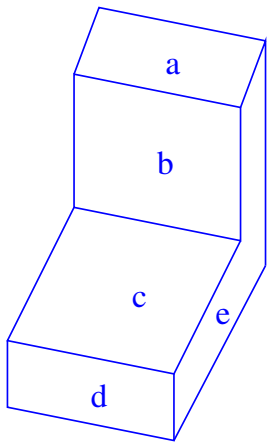
- Graph isomorphism: bijective (one-to-one) function $f : X \rightarrow X'$
  - $\mu(x) = \mu'(f(x))$
  - $\forall e = (x_1, x_2),\ \exists e' = (f(x_1), f(x_2)) \mid \nu(e) = \nu'(e')$ and conversely
- Often too strict! $\Rightarrow$ sub-graphs isomorphism

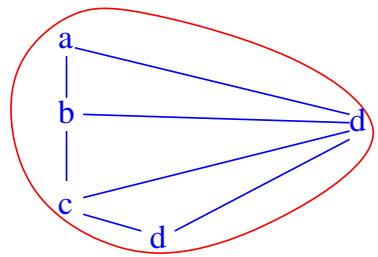
Sub-graphs isomorphism

- There exists a sub-graph S' of G' such that f is an isomorphism from G into S'





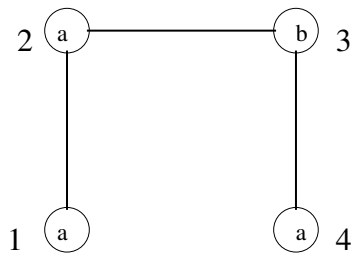


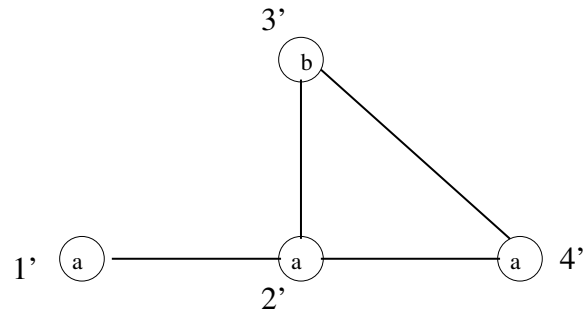


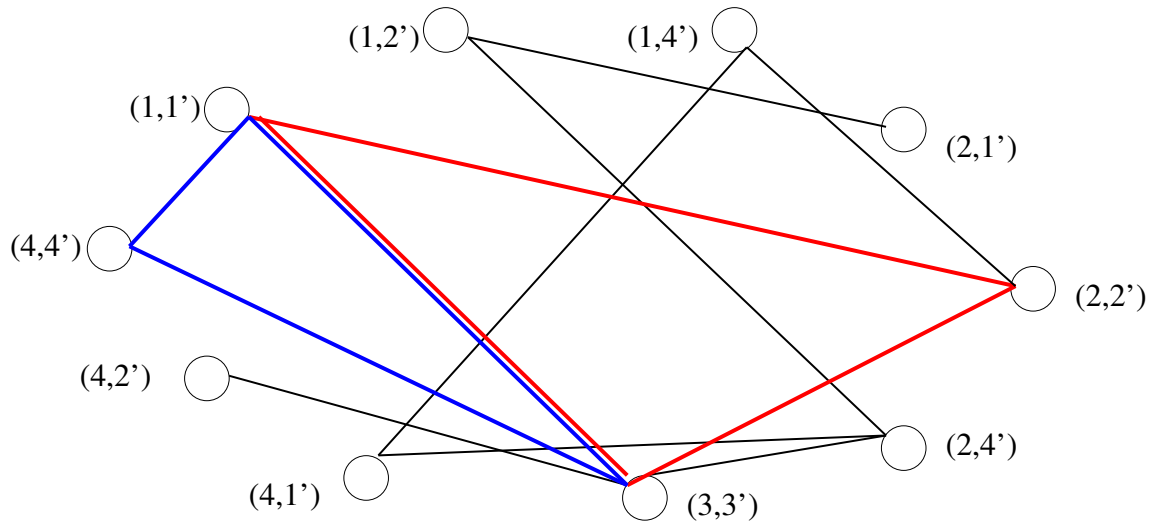
- There exist a sub-graph S of G and a sub-graph S' of G' such that f is an isomorphism from S into S'

Isomorphism as a maximal clique of the association graph

- principle: building the association graph
- maximal clique = sub-graph isomorphism







Sub-graph isomorphism: Ullman algorithm

- Principle: extension of the association set until the graph G has been fully explored.
  In case of failure, backtrack in the association graph.
- Acceleration: "forward checking" before adding an association.
- Algorithm:
  - matrix of vertex associations
  - matrix of future possible associations for a given set of associations
  - list of updated associations using "Backtrack" and "ForwardChecking" procedures
  - Complexity: worst case $O(m^n n^2)$ (n order of X, m order of X', $n < m$)

Error tolerant graph matching

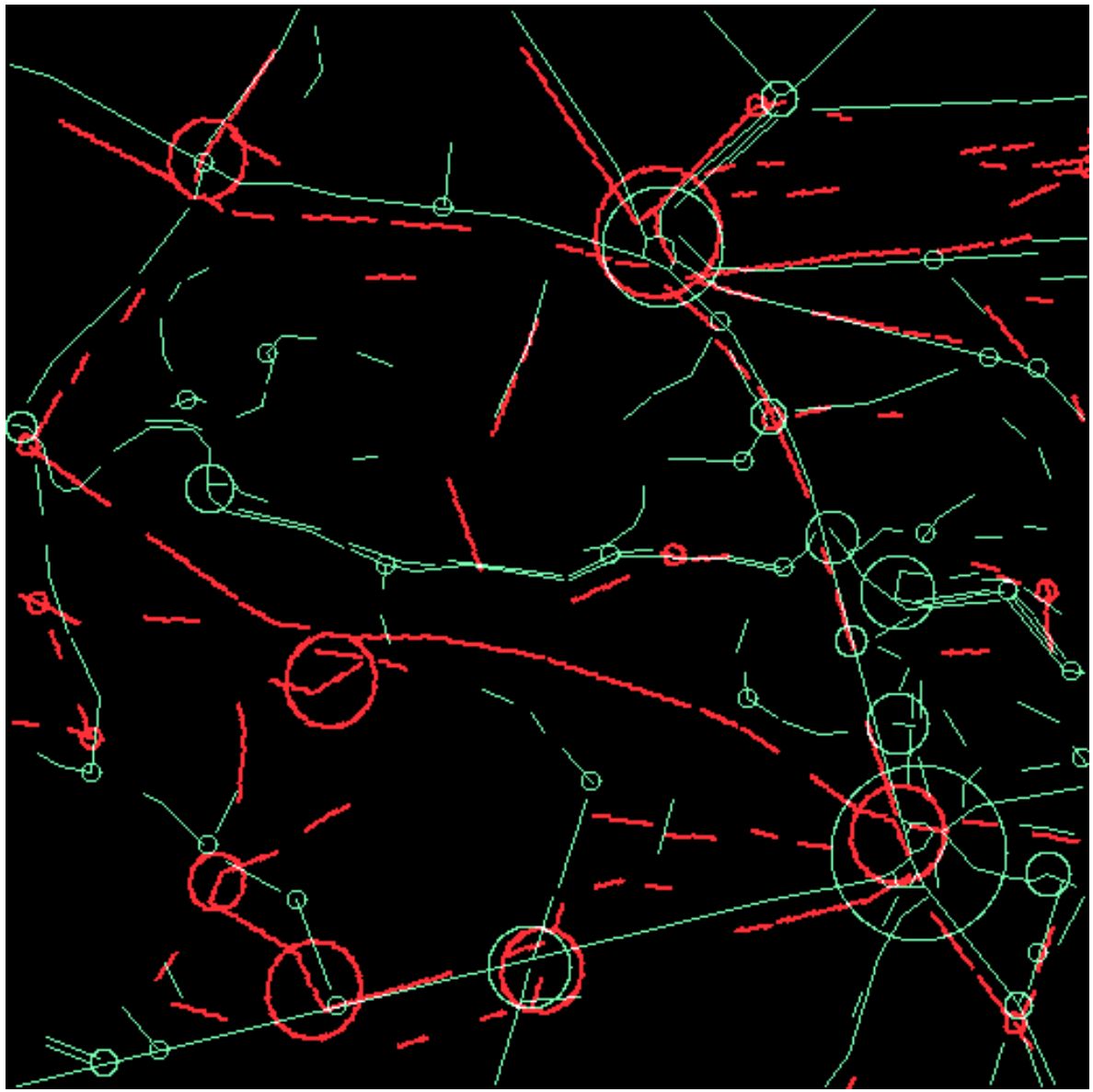
- Real world: noisy graphs, incomplete graphs, distorsions
- Distance between graphs (editing, cost function...)
- Sub-graph isomorphism with error tolerance: search the sub-graph G' with the minimum distance to G
- Optimal algorithm: A* (guarantee of correctness, exponential complexity)
- Approximate matching: genetic algorithms, simulated annealing, neural networks, probabilistic relaxation...
  - iterative minimization of an objective function
  - better adapted for big graphs
  - problem of convergence and local minima

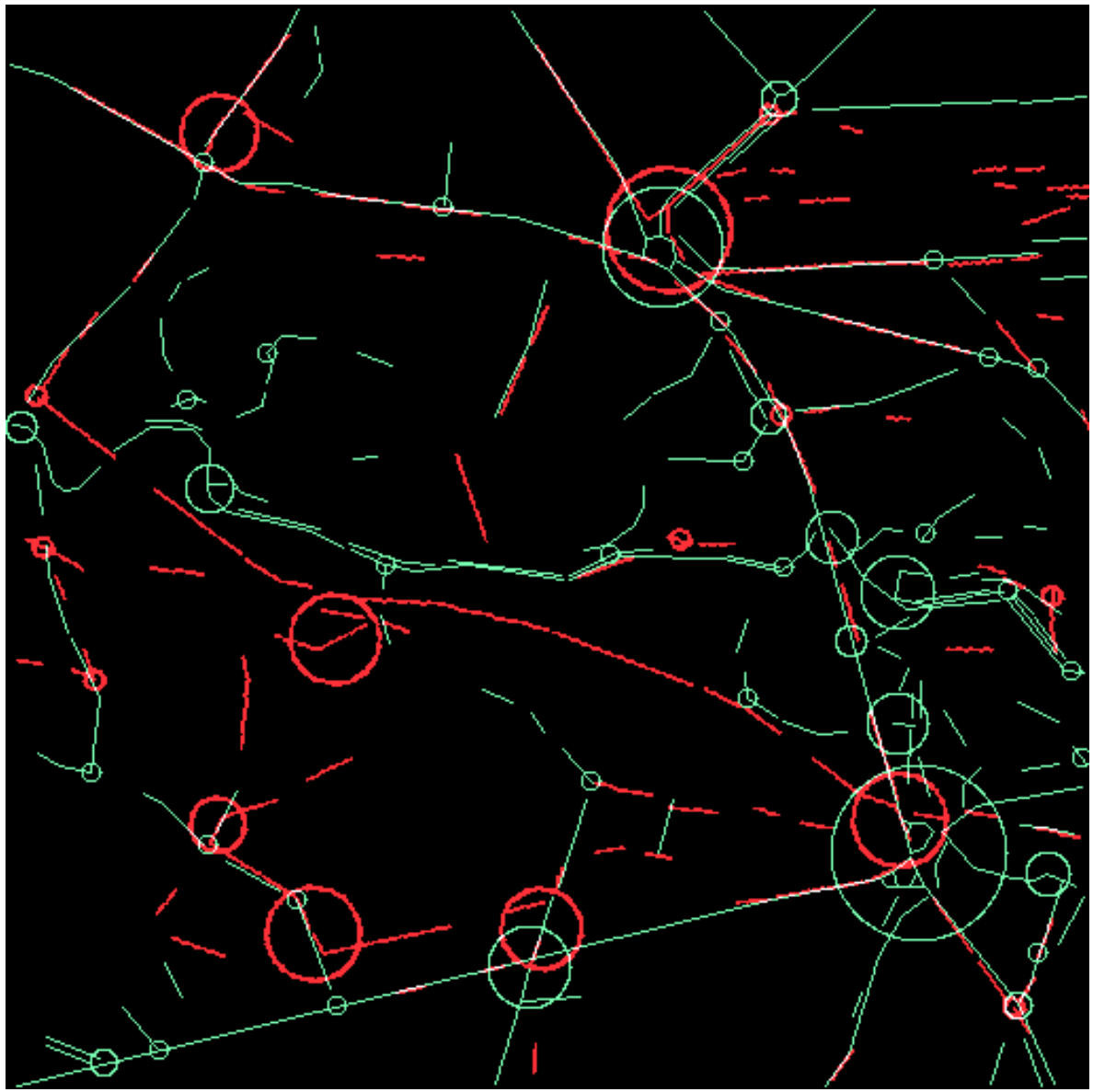
A* algorithm

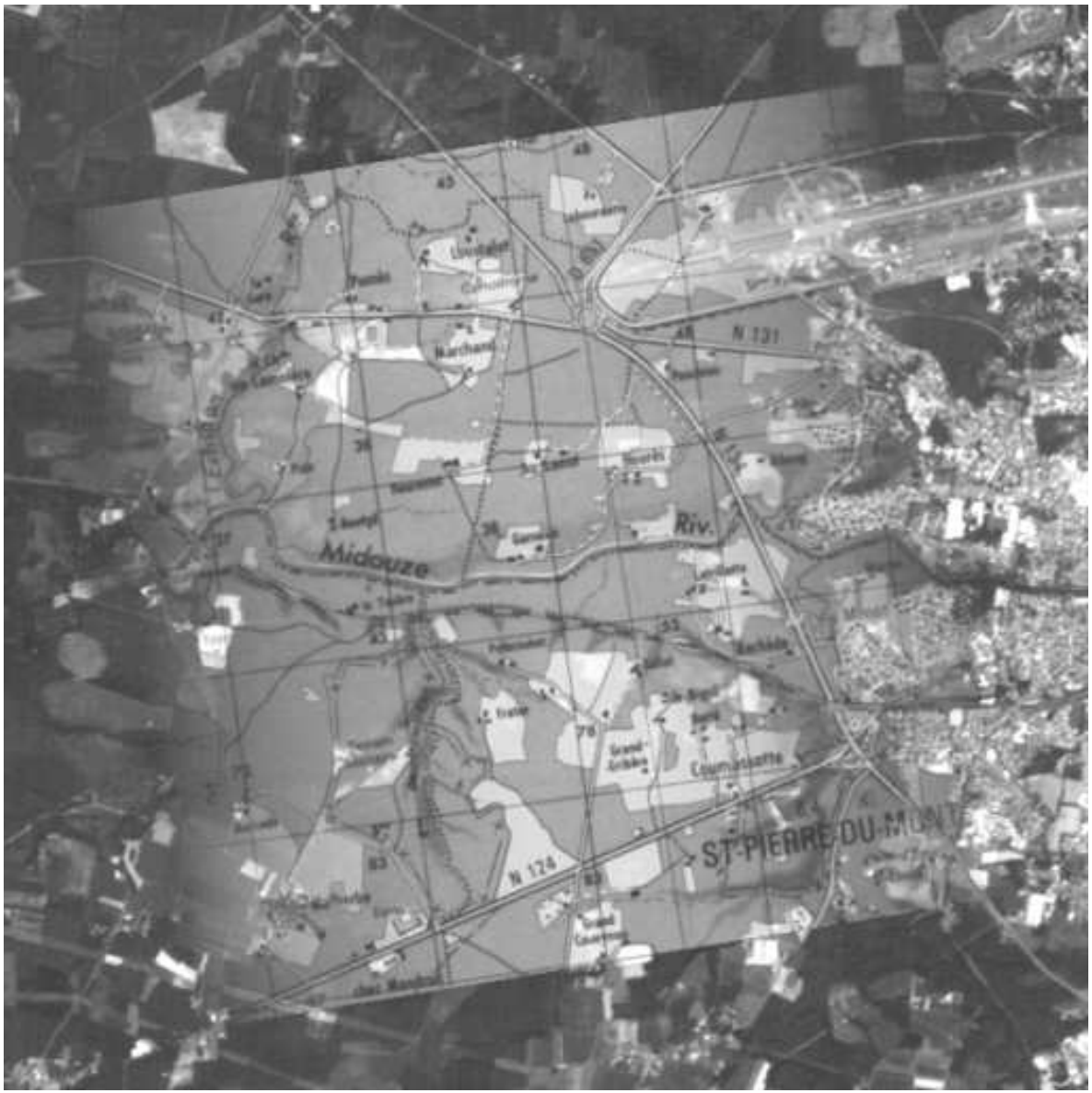
- Principle: building a tree search by successive vertex pairings, and evaluation of a cost function for each state
  (only low cost states are propagated)
- Complexity: $O(n^2 m^n)$ (worst case)
- Acceleration: estimation of futur costs to avoid unnecessary branch exploration

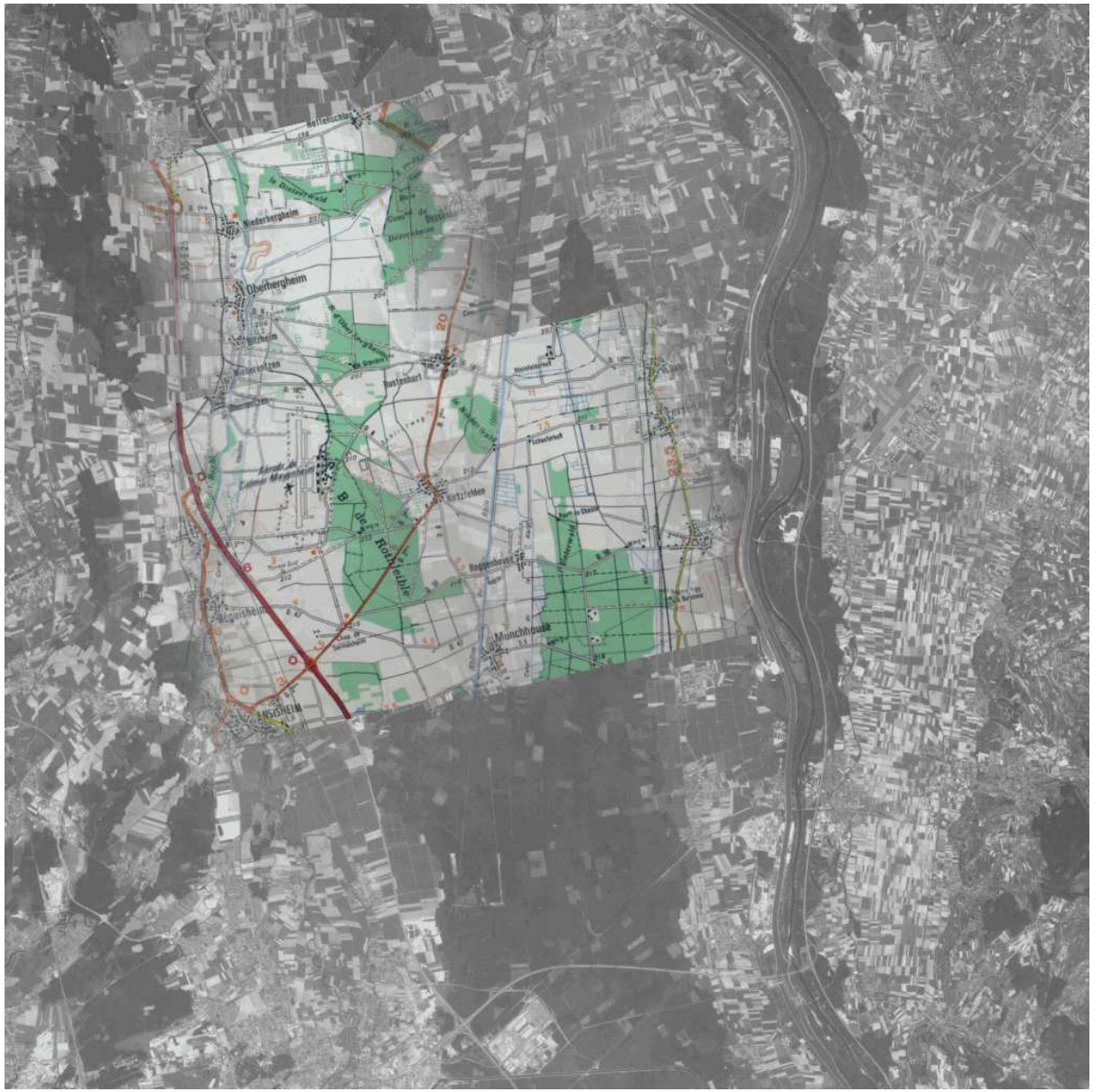
Example: matching a map and a SPOT image (M. Roux)

Pomès
Loustalot
Cynodrome
Labourdette
D 651
Tire-Dur
la Cure
Maque
la Ch.au de Lamolère
Marchand
l'Estrigon
Ste Pomp.
N 131
Pouchiou
St Pomp.
Plan
Bousquès
Hourestot
Au Lanot
Hourès
N 134
Routgé
Midouze
Riv.
Garreloit
Lespugnat
Yoye
la Tuilère
Pilon
Pins
Joyanas
St d'Epur
Latrillette
Pelecoubat
Saliot
Rechède
Manot
Frater
Terrain Militaire
le Bigné
Burté
Marcade
Grand-Gribère
Coumassotte
Malherbe
N 124
ST-PIERRE-DU-MONT
Ch. d'Eau
St Louis
Grand Couevnon
chez Mandron



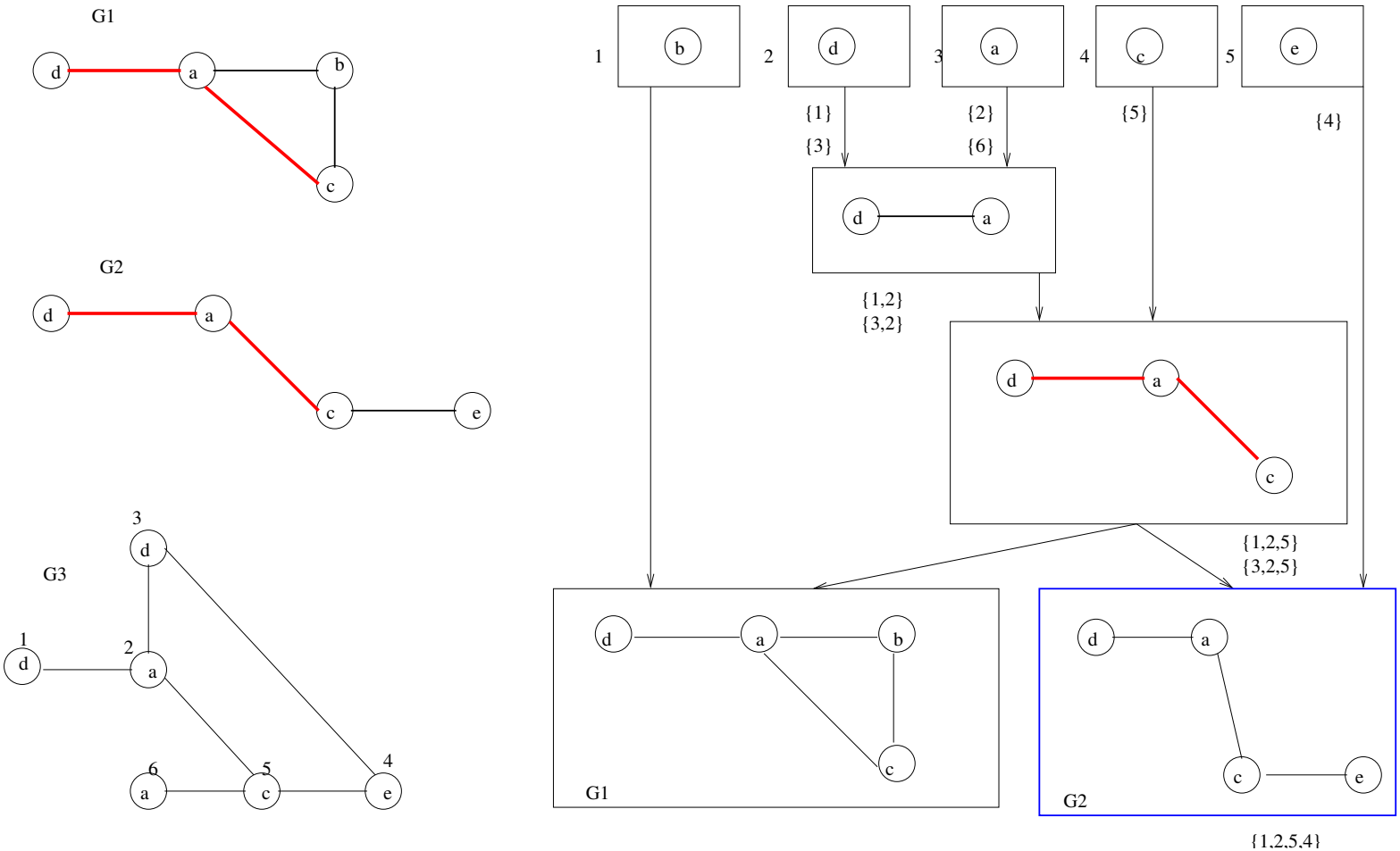



N 131
Riv.
Midouze
N 124
ST-PIERRE-DU-M

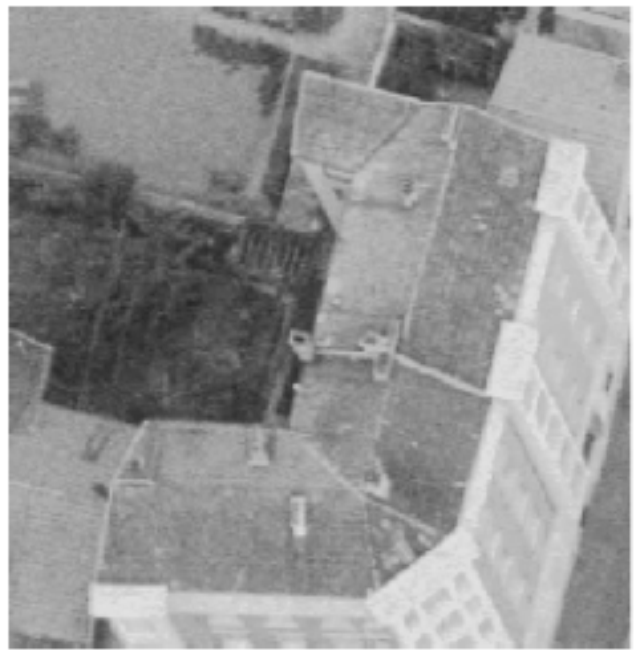

Niederhergheim
Obenhergheim
Munchhouse
EINSISHEIM

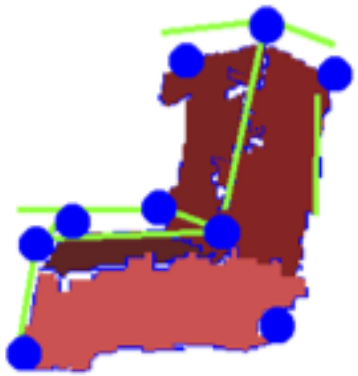
Decomposition into common sub-graphs

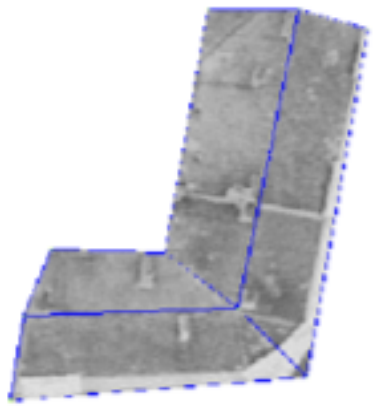
Messmer (1996)



Example: 3D reconstruction by graph matching between a graph (data) and a library of model graphs (IGN)







Inexact matching as an optimization problem

- Dissimilarity between vertices:

$$c_N(a_D, a_M) = \sum \alpha_i d(a_i^N(a_D), a_i^N(a_M)) \quad \sum \alpha_i = 1$$

- Dissimilarity between edges:

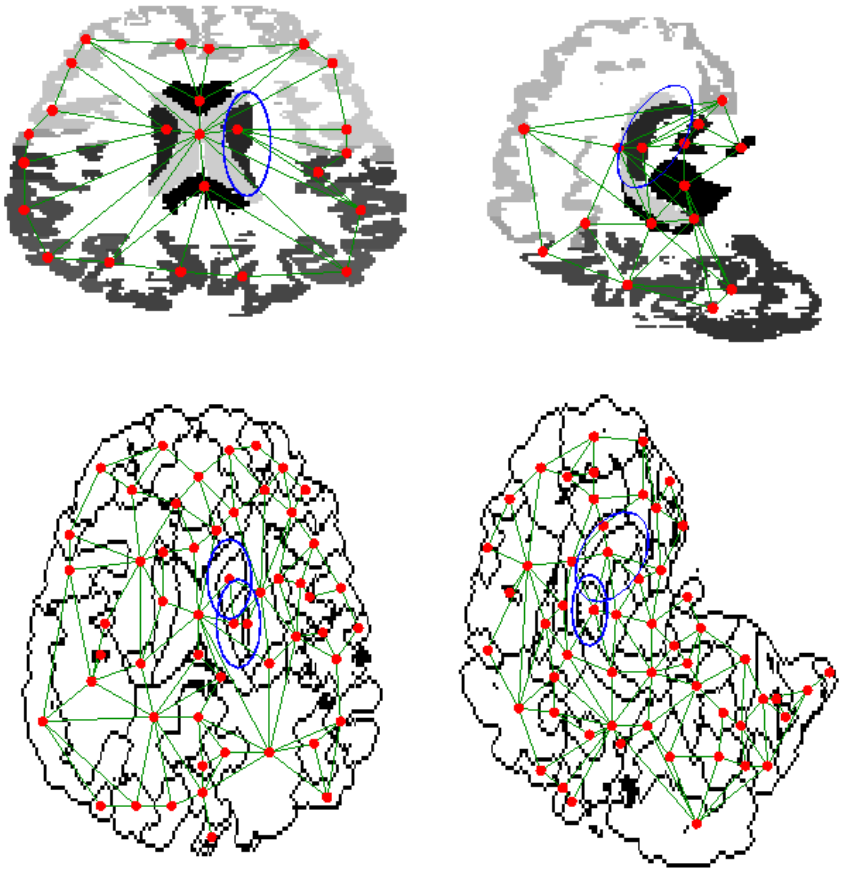
$$C_E((a_D^1, a_D^2), (a_M^1, a_M^2)) = \sum \beta_j d(a_j^A(a_D^1, a_D^2), a_j^A(a_M^1, a_M^2)) \quad \sum \beta_j = 1$$

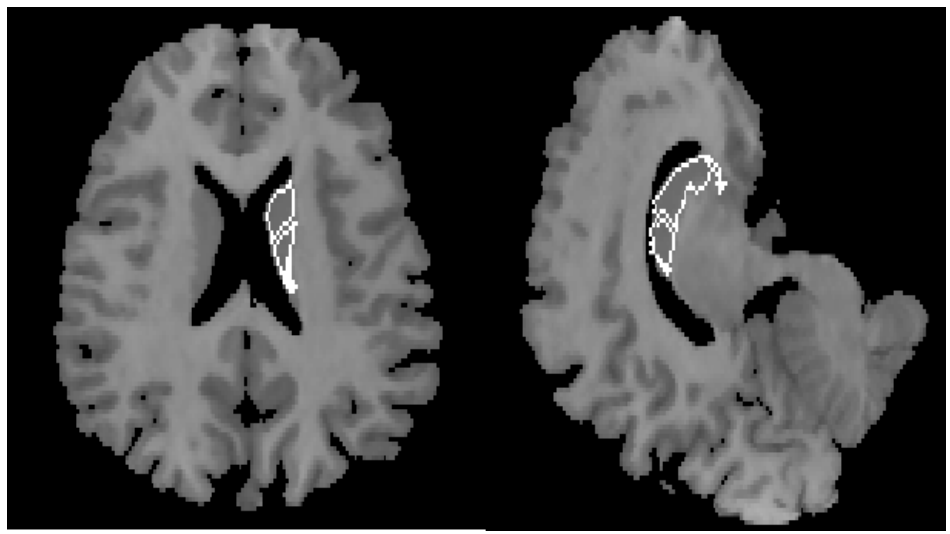
- Cost of matching h:

$$f(h) = \frac{\alpha}{|N_D|} \sum_{a_D \in N_D} c_N(a_D, h(a_D)) + \frac{1-\alpha}{|E_D|} \sum_{(a_D^1, a_D^2) \in E_D} c_E((a_D^1, a_D^2), (h(a_D^1), h(a_D^2)))$$

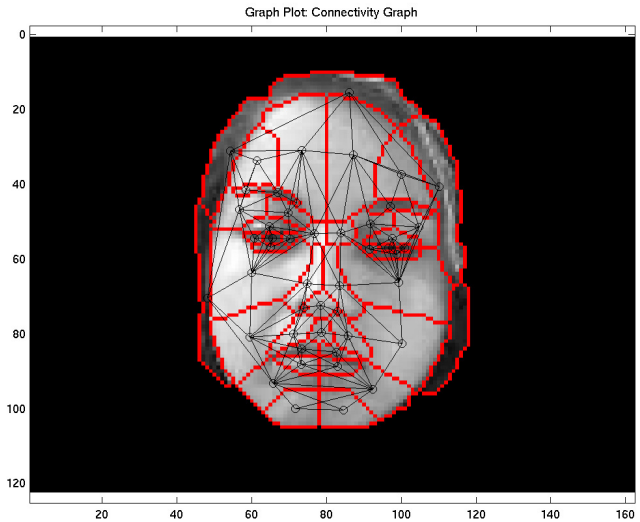
- Optimization of the cost function:
  - tree search
  - expectation maximization (EM)
  - genetic algorithms
  - Bayesian networks and estimation of distributions
  - ...

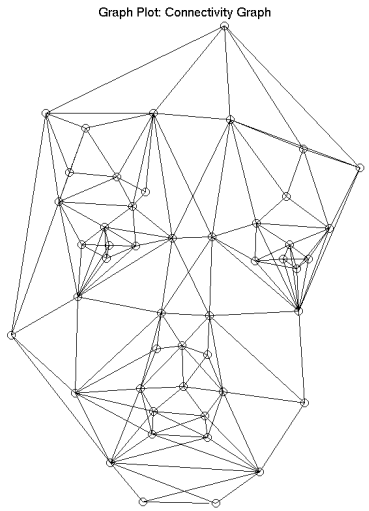
Example: brain structures (A. Perchant)

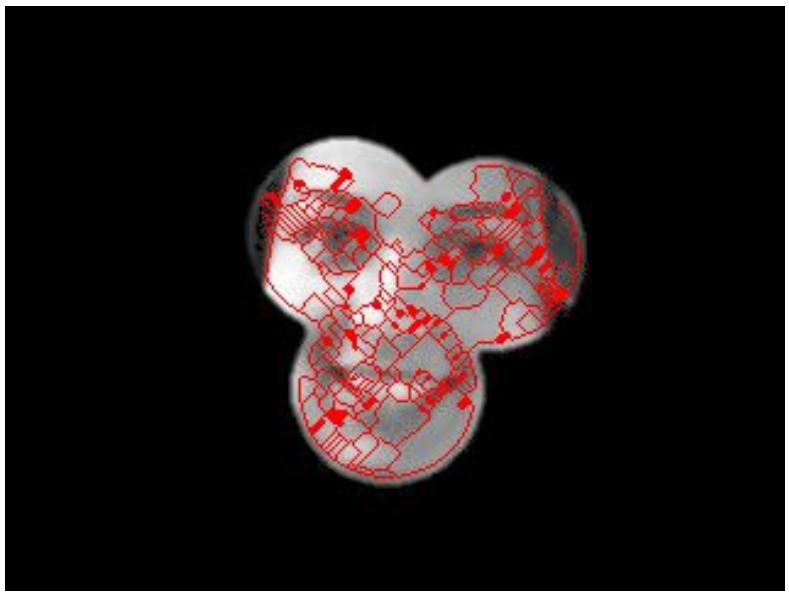


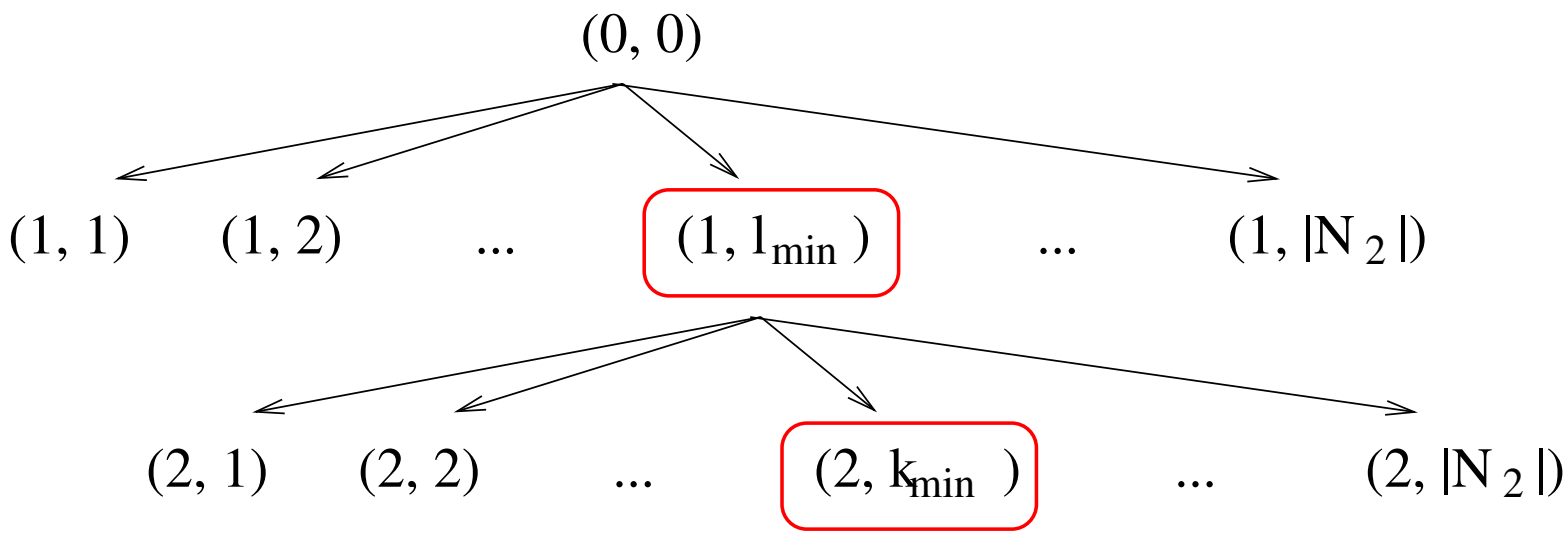


Example: faces (R. Cesar et al.)

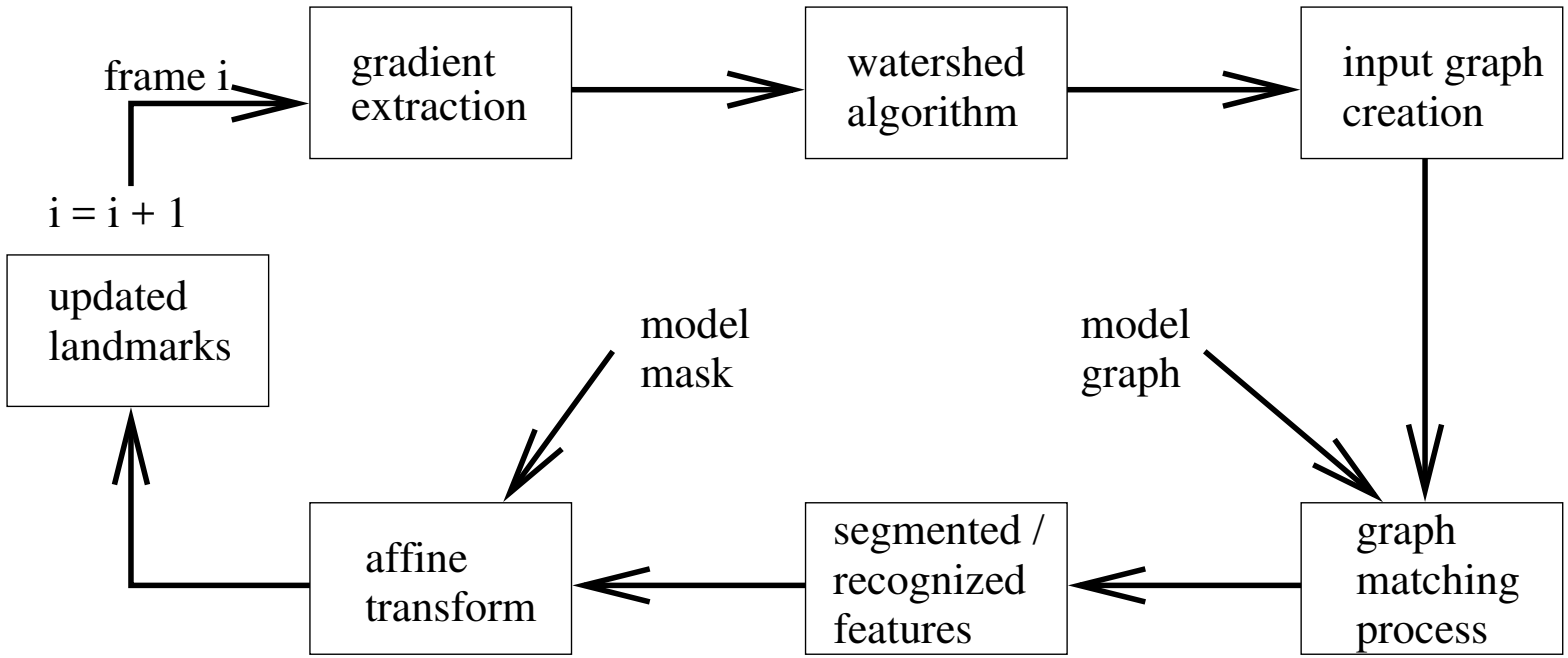






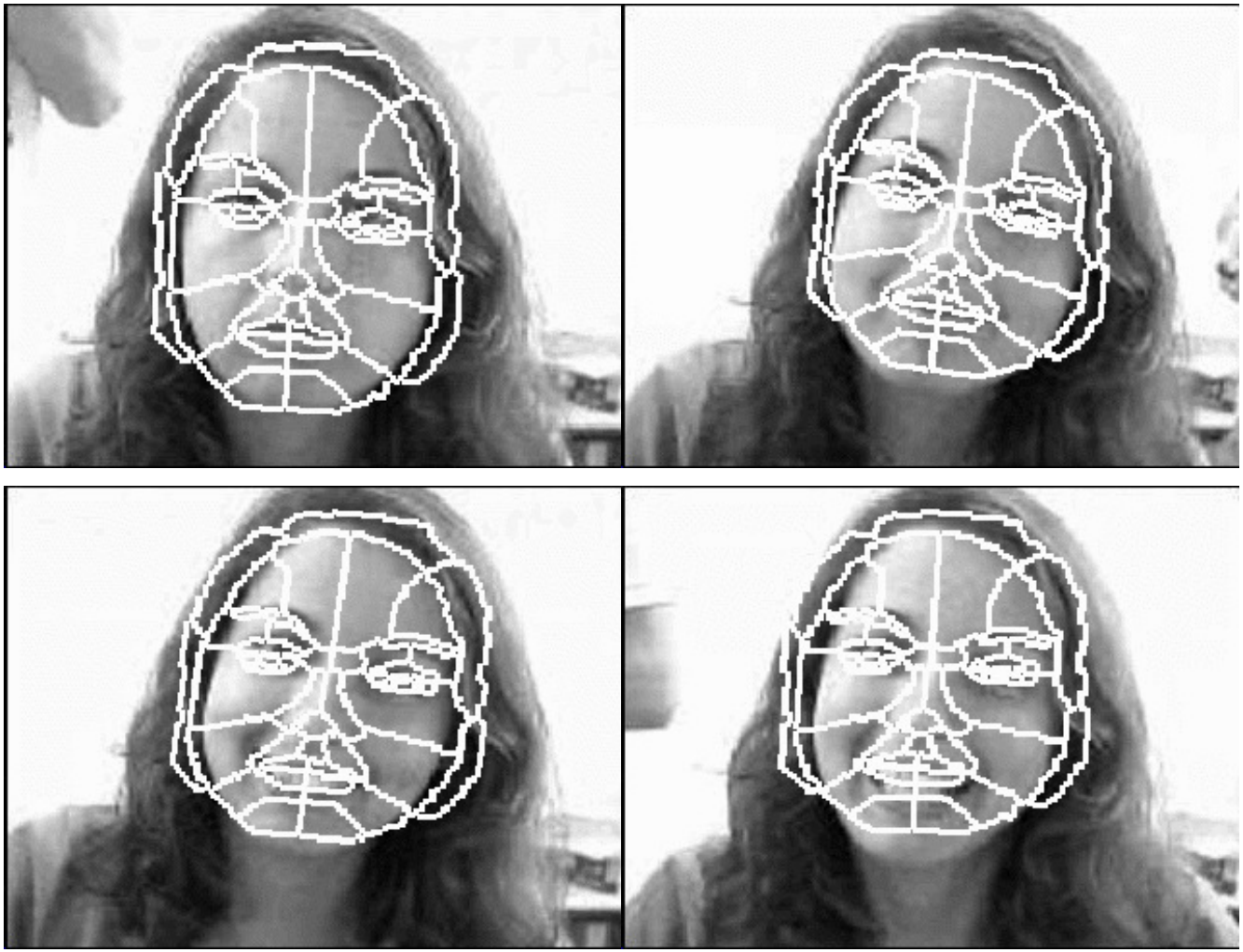


etc.

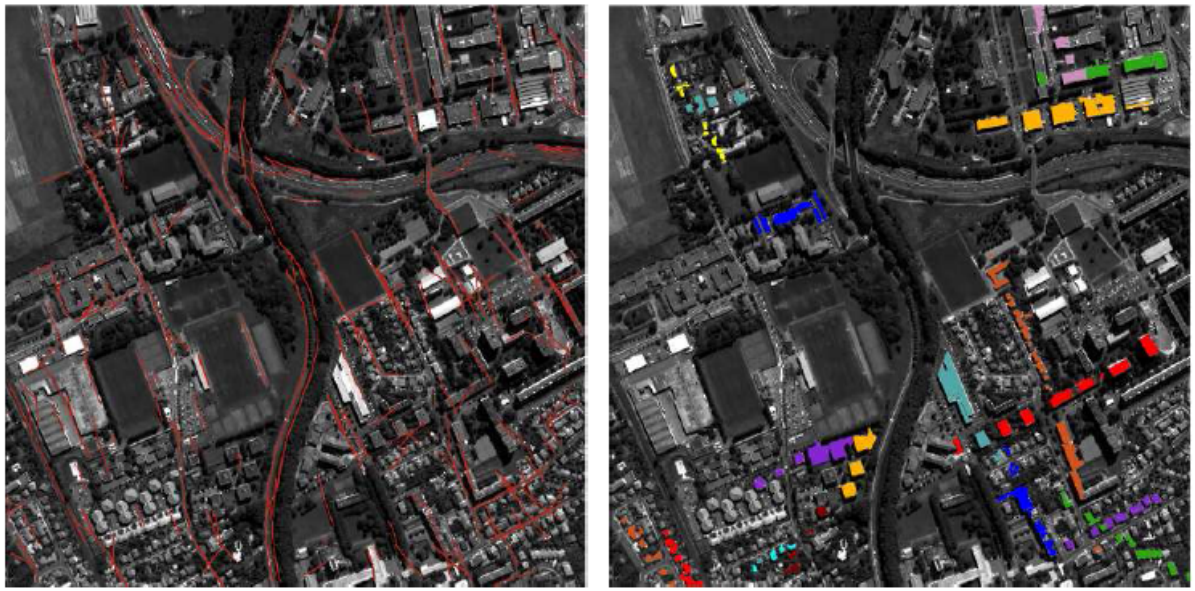

frame i
gradient extraction
watershed algorithm
input graph creation
i = i + 1
updated landmarks
model mask
model graph
affine transform
segmented / recognized features
graph matching process

Segmentation

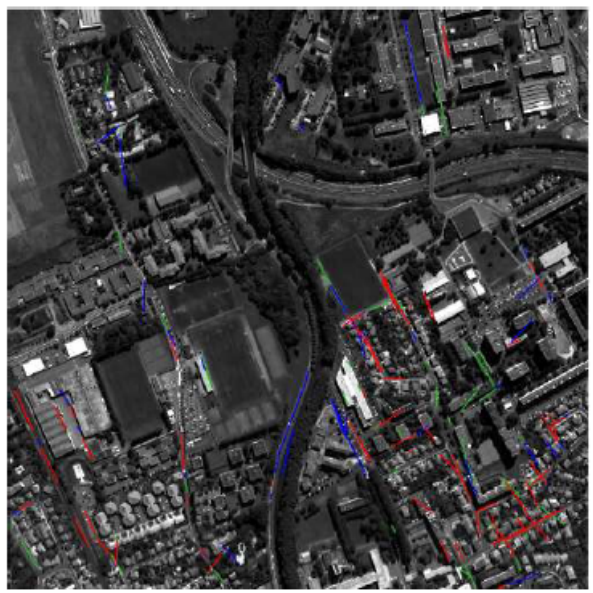
Deformation of the model during tracking



Spatial reasoning (C. Vanegas)

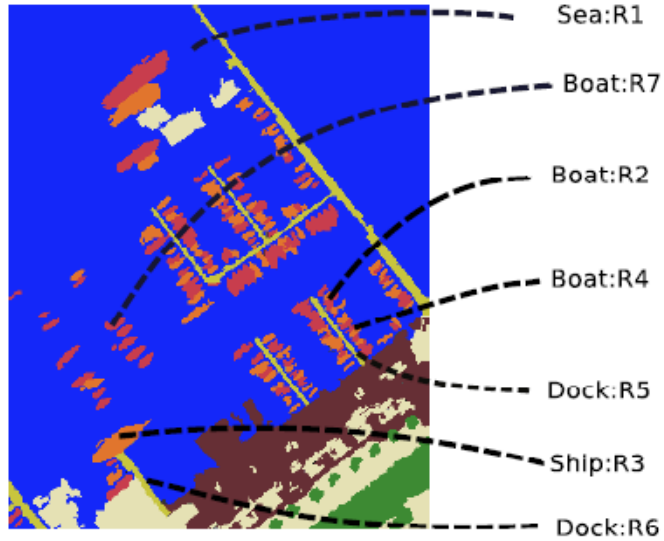


(a) (b)

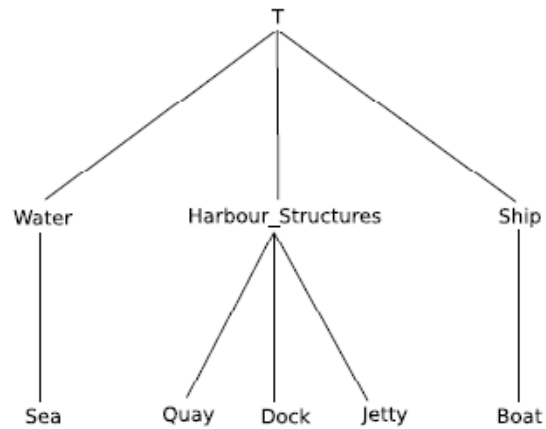


(c)

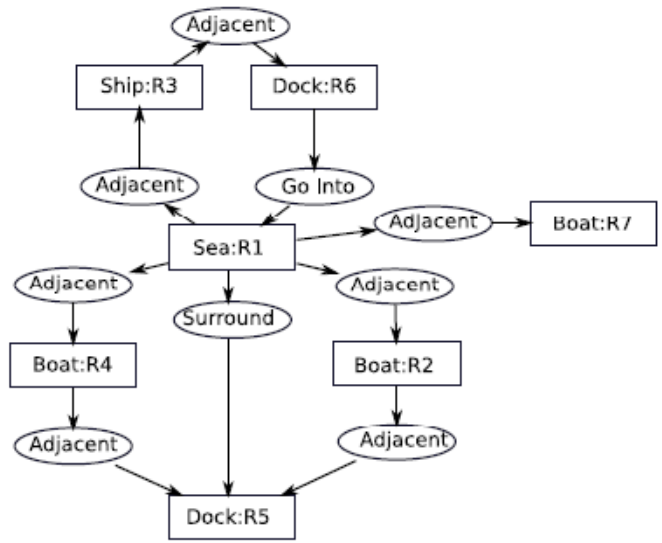
(a) Example image.



(b) Labeled image: The blue regions represent the sea, the red and orange represent ships or boats and the yellow regions represent the docks.



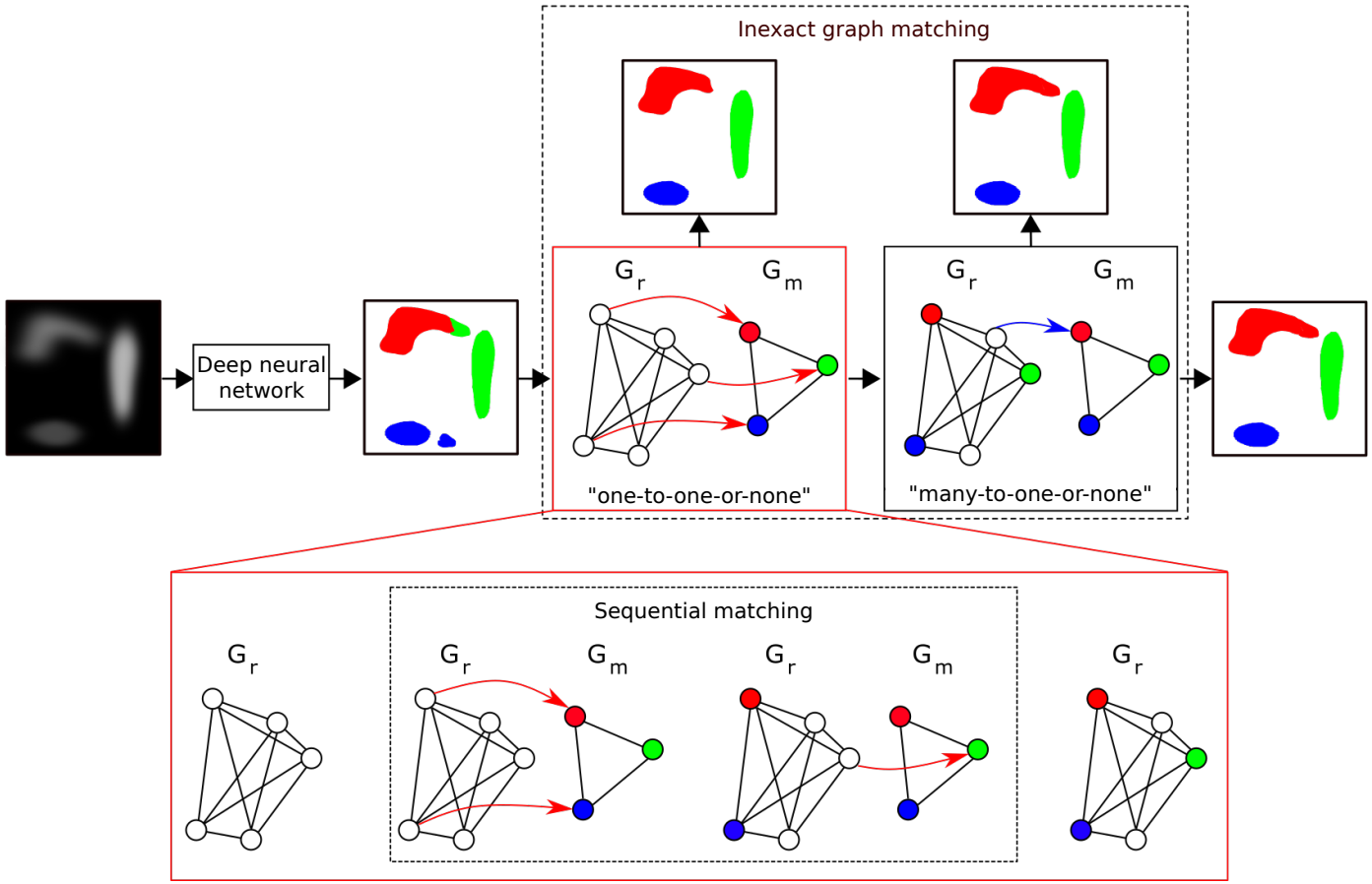
(c) Concept hierarchy T_C in the context of harbors.



(d) Conceptual graph representing the spatial organization of some elements of Figure 5.8(b).

Combining graphs and neural networks

- post-processing (ex: J. Chopin)



- graph embeddings
- message passing (a node receives a message from its neighbors)
- graph convolution
- ...