Graphs for image processing, analysis and pattern recognition

Florence Tupin
Florence.Tupin@telecom-paristech.fr
http://perso.telecom-paristech.fr/~tupin

Télécom ParisTech - LTCI
Paris - France
Overview

1. Definitions and representation models

2. Single graph methods
   - Segmentation or labeling and graph-cuts
   - Graphs for pattern recognition

3. Graph matching
   - Graph or subgraph isomorphisms
   - Error tolerant graph-matching
   - Approximate algorithms (*inexact matching*)
Overview

1. Definitions and representation models

2. Single graph methods
   • Segmentation or labeling and graph-cuts
   • Graphs for pattern recognition

3. Graph matching
   • Graph or subgraph isomorphisms
   • Error tolerant graph-matching
   • Approximate algorithms (inexact matching)
Why using graphs?

- Interest: they give a compact, structured and complete representation, easy to handle
- Applications:
  - Image processing: segmentation, boundary detection
  - Pattern recognition: printed characters, objects (buildings 2D ou 3D, brain structures, ...), faces, ...
  - Image registration
  - Understanding of structured scenes
  - ...

F. Tupin - Graphes – p.4/91
Definitions

Graph : \( G = (X, E) \)

- \( X \) set of nodes (\(|X|\) order of the graph)
- \( E \) set of edges (\(|E|\) size of the graph)

- complete graph (size \( \frac{n(n-1)}{2} \))
- partial graph \( G = (X, E') \) with \( E' \) part of \( E \)
- subgraph \( F = (Y, E') \), \( Y \subseteq X \) et \( E' \subseteq E \)
- degree of a node \( x \) : \( d(x) = \) number of edges
- connected graph: for each pair of nodes you find a path linking them
- tree: connected graph without cycle
- clique: complete subgraph
- dual graph (face \( \rightarrow \) node)
- segment graph (edge \( \rightarrow \) node)
- hypergraph (n-ary relations)
- weighted graphs: weights on the edges
Notations

Graph: \( G = (X, E) \)

- weight of an edge linking \( i \) et \( j \) : \( w_{ij} \)
- adjacency matrix \( W \) of size \(|X| \times |X|\) defined by

\[
W_{ij} = \begin{cases} 
  w_{ij} & \text{if } e_{ij} \in E \\
  0 & \text{else}
\end{cases}
\]

for undirected edges \( W \) is symmetric
- Laplacian matrix of an undirected graph

\[
d_i = \sum_{e_{ij} \in E} w_{ij}
\]

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

\[
L = D - W
\]

with \( D_{ii} = d_i \)
Representation

Adjacency matrix, adjacency lists
Representation

Adjacency matrix, adjacency lists

FIGURE 1.4
From top-left to bottom-right: a weighted directed graph, its adjacency list, its adjacency matrix, and its (transposed) incidence matrix representations.

(figure from “Image processing and analysis with graphs”, Lézoray - Grady)
Examples of graphs

- **Attributed graph** : \( G = (X, E, \mu, \nu) \)
  - \( \mu : X \to L_X \) nodes interpreter \((L_X = \text{attributes of nodes})\)
  - \( \nu : E \to L_E \) edges interpreter \((L_E = \text{attributes of edges})\)

Exemples :
- graph of pixels
- region adjacency graph (RAG)
- Voronoï regions / Delaunay triangulation
- graph of primitives with complex relationships
- **Random graph** : edges and nodes = random variables
- **Fuzzy graph** : \( G = (X, E = X \times X, \mu_f, \nu_f) \)
  - \( \mu_f : X \to [0, 1] \)
  - \( \nu_f : E \to [0, 1] \)
  - avec \( \forall (u, v) \in X \times X \) \( \nu_f(u, v) \leq \mu_f(u)\mu_f(v) \) or \( \nu_f(u, v) \leq \min[\mu_f(u)\mu_f(v)] \)
Examples of image graphs

FIGURE 1.11
The rectangular (left) and hexagonal (right) lattices and their associated Voronoi cells.

FIGURE 1.12
Different adjacency structures in a 3D lattice.

(figure from “Image processing and analysis with graphs”, Lézoray - Grady)
Examples of image graphs

RAG (Region Adjacency Graph)
Examples of image graphs

Voronoï diagram (in blue) and Delaunay triangulation (pink)
Examples of image graphs

Figure 1.14
Examples of proximity graphs from a set of 100 points in $\mathbb{Z}^2$.

(figure from “Image processing and analysis with graphs”, Lézoray - Grady)
**Examples of graphs**

- **Graph of fuzzy attributes**: attributed graph with fuzzy value for each attribute
- **Hierarchical graph**:
  multi-level graph and bi-partite graph between 2 levels
  (multi-level approaches, object grouping, ...)

  Exemples :
  - quadtrees, octrees
  - hierarchical representation of the brain

- **Graph for reasoning**
  decision tree, matching graph
Graph examples

(a) An image with the quadtree tessellation, (b) the associated partition tree, (c) a real image with the quadtree tessellation, (d) the region adjacency graph associated to the quadtree partition, (e) and (f) two different irregular tessellations of an image using image-dependent superpixel segmentation methods: Watershed [23] and SLIC superpixels [24].

(figure from “Image processing and analysis with graphs”, Lézoray - Grady)
Graph examples

Figure 2 – Représentation de variété des points clés de $S_{\omega}^{\text{max}}(I)$ (en rouge) et $S_{\omega}^{\text{min}}(I)$ (en bleu) sur une image Pléiades ayant des textures locales différentes.

(figure from M.T. Pham PhD, 2016)
Graph examples

(a) Image initiale 512 × 512
(b) Extrema locaux
(c) Détecteur de Harris
(d) Détecteur SIFT

Figure 2: Evolution de la distribution de points et d'indications d'extrema initiaux.
Graph examples

**Figure 5** – Vecteur de description proposé pour l’analyse ponctuelle de la texture.
Graph examples - BPT Binary Partition Tree
Some classical algorithms

Search of the minimum spanning tree
- Kruskal algorithm $O(n^2 + m \log_2(m))$
- Prim algorithm $O(n^2)$

Shortest path problems
- positive weights: Dijkstra algorithm $O(n^2)$
- arbitrary weights but without cycle: Bellman algorithm $O(n^2)$

Max flow and Min cut
- $G = (X, E)$
- partitioning in two sets $A$ et $B$ ($A \cup B = X$, $A \cap B = \emptyset$)
- $\text{cut}(A, B) = \sum_{x \in A, y \in B} w(x, y)$
- Ford and Fulkerson algorithm

Search of maximal clique in a graph
- decision tree
- cut of already explored branches
Overview

1. Definitions and representation models

2. Single graph methods
   - Segmentation or labeling and graph-cuts
   - Graphs for pattern recognition

3. Graph matching
   - Graph or subgraph isomorphisms
   - Error tolerant graph-matching
   - Approximate algorithms (*inexact matching*)
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 ⇒ partitioning by suppressing the most costly edges

![Image]

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. sorting of 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
     - else, add the edge in the graph

- Complexity: \(O(n^2 + m \log_2(m))\)

Prim algorithm

- Extension from near to near of the current tree

- Complexity: \(O(n^2)\)
Constantinidès (1986)
Segmentation by graph-cut

Graph-cut definition:

- graph $G = (X, E)$
- partitioning in 2 parts $A$ et $B$ ($A \cup B = X$, $A \cap B = \emptyset$)
- $cut(A, B) = \sum_{i \in A, j \in B} w_{ij}$
Segmentation by graph clustering

Clustering: partitioning of the graph in groups of nodes based on their similarities
Each cluster (group): a closely connected component

The clustering corresponds to:
- edges between different groups have low weights (weak similarities)
- edges inside a group have high weights (high similarities)

Possible cost functions for the cut:
- minimum cut \( \text{Cut}(A_1, \ldots, A_k) = \sum_{i=1}^{k} \text{Cut}(A_i, \overline{A_i}) \)
- minimum cut normalized by the size of each part (RatioCut)
  \( \text{RatioCut}(A_1, \ldots, A_k) = \sum_{i=1}^{k} \frac{1}{|A_i|} \text{Cut}(A_i, \overline{A_i}) \)
  (\( |A_i| \) number of vertices in \( A_i \))
- minimum cut normalized by the connectivity of each part (NCut)
  \( \text{NCut}(A_1, \ldots, A_k) = \sum_{i=1}^{k} \frac{1}{\text{vol}(A_i)} \text{Cut}(A_i, \overline{A_i}) \)
  (\( \text{vol}(A_i) = \sum_{k \in A_i} d_k \) sum of the weight of all edges of vertices in \( A_i \))
Toy example

Wu and Leavy (93): search for the MinCut

Influence of the number of edges: $\text{Cut}(A, B) = 4b$, $\text{Cut}(A', B') = 3b$

$\Rightarrow$ normalized cut (NCut)
Normalized cut

- Principle: graph clustering
- + suppression of the influence of the number of edges: normalized cut

\[ N_{\text{cut}}(A, B) = \frac{\text{cut}(A, B)}{\text{assoc}(A, X)} + \frac{\text{cut}(A, B)}{\text{assoc}(B, X)} \]

\[ \text{assoc}(A, X) = \sum_{a \in A, x \in X} w(a, x) \]

- Measuring the connectivity of a cluster:

\[ N_{\text{assoc}}(A, B) = \frac{\text{assoc}(A, A)}{\text{assoc}(A, X)} + \frac{\text{assoc}(B, B)}{\text{assoc}(B, X)} \]

\[ N_{\text{cut}}(A, B) = 2 - N_{\text{assoc}}(A, B) \]

minimizing the cut ⇔ maximizing group connectivity
Graph theory and cuts

MinCut by combinatorial optimization

• Stoer-Wagner algorithm
• Principle: iterative reducing of the graph by fusion of the nodes linked by the maximal weights

Min K-cut by combinatorial optimization

• Partitioning the (un-oriented graph) graph in many components
• Gomory-Hu algorithm

minCut in oriented graph by combinatorial optimization

• Ford-Fulkerson algorithm (oriented graph with two terminal nodes (sink / tank)
• Principle: MaxFlow search (MinCut equivalence) by search for an augmenting chain to increase the flow
Graph theory and cuts

Laplacian matrices
\[ D = \text{diag}(d_i) \text{ with } d_i = \sum_j w_{ij} \]
\[ W = (w_{ij}) \]
- Graph Laplacian matrix
  \[ L = D - W \]
- Normalized graph Laplacian matrix
  \[ L_n = D^{-\frac{1}{2}} LD^{-\frac{1}{2}} = I - D^{-\frac{1}{2}} WD^{-\frac{1}{2}} \]

Spectral clustering algorithms and cuts
- Computation of the eigen-values and eigen-vectors of some matrix (\( L, L_n \), or generalized eigen problems \( Lu = \lambda Du \))
- selection of the \( k \) smallest eigen-values and associated \( k \) eigen-vectors \( u_k \)
- \( U = (u_1, \ldots, u_k) \in R^{n \times k} \)
- let \( y_i \in R^k \) be the \( i \)th row of \( U \) (\( i = 1, \ldots, n \))
- cluster the points \( (y_i)_{1 \leq i \leq n} \) with the k-means algorithm into clusters \( C_1, \ldots, C_k \)
- clusters \( A_1, \ldots, A_k \) with \( A_j = \{ j | y_j \in C_i \} \)
Examples (univ. Berkeley)

http://www.cs.berkeley.edu/projects/vision/Grouping/
Examples (univ. Berkeley)

http://www.cs.berkeley.edu/projects/vision/Grouping/
Examples (univ. Alberta) with linear constraints
Examples (Mean Shift et Normalized Cut)
Examples (texture classification with point-wise graph)
Graph-cuts

Bibliography

• An optimal graph theoretic approach to data clustering: theory and its application to image segmentation, Z. Wu et R. Leahy, IEEE PAMI, vol.15, num.11, nov. 93
• Normalized cuts and image segmentation, J. Shi et J. Malik, IEEE PAMI, vol. 22, num. 8, 2000
• Image segmentation with minimum mean cut, Wang, Siskind, 8th ICCV, 2001
• Efficient graph-based image segmentation, Felzenszwalb, Huttenlocher, IJCV, 2004
• Color Image segmentation Based on Mean-Shift and Normalized Cuts, Tao, Zhang, IEEE Trans. on Systems, Man and Cybernetics, 2007
• Pointwise approach for texture analysis and characterization from VHR remote sensing images, M.-T. Pham, PhD, 2016
Overview

1. Definitions and representation models

2. Single graph methods
   - Segmentation or labeling and graph-cuts
   - Graphs for pattern recognition

3. Graph matching
   - Graph or subgraph isomorphisms
   - Error tolerant graph-matching
   - Approximate algorithms (*inexact matching*)
Full scene labeling (scene parsing)

Figure from Farabet et al., PAMI 13
Tenenbaum and Barrow (1977)

- Segmentation in regions
- Building of the 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 with fuzzy attributed graphs
Markovian labeling (random graphs)

\[ E(l) = \sum_i \Phi(d_i, l_i) + \beta \sum_{ij} \Psi(l_i, l_j) \]

- Low-level applications:
  - pixel graphs
  - segmentation, classification, restoration

- High-level applications:
  - graph of super-pixels (SLIC, watershed, ...)
  - graph of primitives (edges, key-points, lines, ...)

⇒ pattern recognition, full scene labeling

F. Tupin - Graphes – p.38/91
Example on a region adjacency graph (T. Géraud)

nuclei segmentation

3D morphological closing

3D watershed

3D over-segmentation

result of graph labeling

Markovian relaxation

\[ p(y|x) = \exp \left\{ -\sum \frac{\text{vol}_s(y_s - \mu_s)^2}{2\sigma^2_s} \right\} \]
\[ p(x) = \frac{1}{Z} \exp \left\{ -\sum \sum \text{surf}_{s,n} P(x_s,x_n) \right\} \]
Example on a line graph
Example on a region adjacency graph
Example on a region adjacency graph
Markov random fields and graph-cut optimization

Binary labeling (Greig et al. 89):

\[ \mathcal{E}(l) = \sum_{i} \Phi(d_i | l_i) + \sum_{(i,j)} \beta (l_i - l_j)^2 \]

- source \( S \) (label 1), sink \( P \) (label 0)
- edges connected to terminal nodes with likelihood weights \( \Phi(d_i | l_i) \)
- edges between neighbor nodes with weights \( \beta \)

Minimizing \( \mathcal{E}(l) \) \( \iff \) Min Cut search

\[ \text{cut}(E_S, E_P) = \sum_{i \in E_S} \Phi(d_i | 1) + \sum_{i \in E_P} \Phi(d_i | 0) + \sum_{(i \in E_S, j \in E_P)} \beta \]

\((l_i = 1 \text{ for } i \in E_S, l_i = 0 \text{ for } i \in E_P)\)
(l_i = 1 for \( i \in E_S \), \( l_i = 0 \) for \( i \in E_P \))
MRF and graph-cut optimization

FIGURE 2.5
(a) The graph for binary MRF minimization. The edges in the cut are depicted as thick arrows. Each node other than $v_s$ and $v_t$ corresponds to a site. If a cut $(S, T)$ places a node in $S$, the corresponding site is labeled 0; if it is in $T$, the site is labeled 1. The 0’s and 1’s at the bottom indicate the label each site is assigned. Here, the sites are arranged in 1D; but according to the neighborhood structure this can be any dimension as shown in (b) and (c).

(figure from “Image processing and analysis with graphs”, Lézoray - Grady)
**MRF/CRF and graph-cut optimization**

Multi-level labeling (Boykov, Veksler 99):

⇒ generalization of the previous binary labeling

Definition of two space moves (to go back to the binary labeling)

- $\alpha$-expansion: source $S$ and sink $P$ correspond to label $\alpha$ and the current label $\overline{\alpha}$ ($\Psi$ should be a metric)

- $\alpha - \beta$ swap: source $S$ for $\alpha$ and sink $P$ for $\beta$ ($\Psi$ should be a semi-metric)

Optimization by iterative mincut search:

- graph: nodes for super-pixels
- weights: depending on the current labeling
- good trade off time / efficiency compared to simulated annealing or ICM

But for multi-labeling no guarantee on optimality of the solution
MRF/CRF and graph-cut optimization

Image restoration:

⇒ exact optimization for quantized levels when $\Psi$ is convex

- Ishikawa (2003): building of a multi-layer graph (one layer for each label) and mincut search
- Darbon (2005): decomposition of the solution on level-sets and binary mincut search on each level-set

⇒ exact solution for convex functions!
⇒ but need of (potentially) huge memory size!....
Examples - multi-labeling optimization

(a) (b)

(c) (d)

(e) (f)
Interactive segmentation: “hard” constraints

**Principle** Background and object manually defined

⇒ finding of a binary labeling minimizing an energy including “hard” constraints

**Method** Mincut search and edges with high weights (should not be cut)

**Advantages**

- easy introduction of “hard” constraints
- the manually defined areas permit to do a fast learning
- iterative algorithm
Graph construction (Boykov et Jolly, 2001)

(a) Image with seeds.  

(d) Segmentation results.
Graph weights *(Boykov et Jolly, 2001)*

<table>
<thead>
<tr>
<th>edge</th>
<th>weight (cost)</th>
<th>for</th>
</tr>
</thead>
<tbody>
<tr>
<td>{p, q}</td>
<td>(B_{{p, q}})</td>
<td>({p, q} \in \mathcal{N})</td>
</tr>
<tr>
<td>{p, S}</td>
<td>(\lambda \cdot R_p(\text{“bkg”}))</td>
<td>(p \in \mathcal{P}, p \not\in \mathcal{O} \cup \mathcal{B})</td>
</tr>
<tr>
<td></td>
<td>(K)</td>
<td>(p \in \mathcal{O})</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>(p \in \mathcal{B})</td>
</tr>
<tr>
<td>{p, T}</td>
<td>(\lambda \cdot R_p(\text{“obj”}))</td>
<td>(p \in \mathcal{P}, p \not\in \mathcal{O} \cup \mathcal{B})</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>(p \in \mathcal{O})</td>
</tr>
<tr>
<td></td>
<td>(K)</td>
<td>(p \in \mathcal{B})</td>
</tr>
</tbody>
</table>
Illustrations (Boykov et Jolly, 2001)

(a) Original B&W photo

(b) Segmentation results
Interactive methods with mincut

Grab-cut (Rother et al. 2004)

- take into account color
- two labels (background and object but with a Gaussian Mixture Model)
- CRF (conditional random field): regularization term weighted by the image gradient
- iterative semi-supervised learning of the GMM parameters (after manual initialization and after each cut)
Illustrations - GrabCut- (Rother, Kolmogorov et Blake, 2004)
Deep learning and graph labeling for full scene labeling

Farabet et al., PAMI, 2013
Deep learning and graph labeling for full scene labeling

\[ \Phi(d_i, l_i) = \exp(-\alpha d_i,a)1(l_i \neq a) \]

\[ \Psi(l_i, l_j) = \exp(-\beta \|\nabla I\|_i)1(l_i \neq l_j) \]

Farabet et al., PAMI, 2013
Pattern recognition

• Object: defined by a set of primitives (nodes of the graph)
• Binary relationship of compatibility between nodes (edges of the graph)
• Clique: sub-set of primitives all compatible between each other
  = possible object configuration
• recognition by maximal clique detection

Search of maximal cliques:

• NP-hard problem
• Building of a decision tree: a node of the tree = 1 clique of the graph
• pruning of the tree to suppress already found cliques
• Theorem: let $S$ be a node 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 sons $S$ not adjacent to $x$ have to be explored.
Example: building reconstruction by the maximal clique search (IGN)
Example: building reconstruction by the maximal clique search (IGN)
Example: building reconstruction by the maximal clique search (IGN)
Example: building reconstruction by the maximal clique search (IGN)
Example: building reconstruction by the maximal clique search (IGN)
Example: building reconstruction by the maximal clique search (IGN)
Example: building reconstruction by the maximal clique search (IGN)
Overview

1. Definitions and representation models

2. Single graph methods
   • Segmentation or labeling and graph-cuts
   • Graphs for pattern recognition

3. Graph matching
   • Graph or subgraph isomorphisms
   • Error tolerant graph-matching
   • Approximate algorithms (*inexact matching*)
Graph matching

Correspondance problem:
- Graph(s) of the model (atlas, map, model of object)
- Graph built from the data
- Graph matching:

\[ G = (X, E, \mu, \nu) \rightarrow? \quad G' = (X', E', \mu', \nu') \]

Graph isomorphism: bijective function \( f : X \rightarrow X' \)

- \( \mu(x) = \mu'(f(x)) \)
- \( \forall e = (x_1, x_2), \exists e' = (f(x_1), f(x_2)) / \nu(e) = \nu'(e') \) and conversely

Too strict \( \Rightarrow \) isomorphisms of sub-graphs
**Sub-graph isomorphisms**

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

- There exists a sub-graph $S$ of $G$ and a sub-graph $S'$ of $G'$ such that $f$ is an isomorphism from $S$ to $S'$
Graph isomorphisms: searching the maximal clique

search of the maximal clique of the association graph

- principle: building of the association graph
- maximal clique: sub-graph isomorphism
Sub-graph isomorphism: Ullman algorithm

- Principle: extension of the association set \((v_i, w_x_i)\) until the \(G\) graph has been fully explored. In case of failure, go back in the association graph ("backtrack"). Acceleration: “forward checking” before adding an association.

- Algorithm:
  - matrix of node associations
  - matrix of future possible associations for a given set of associations matrix
  - list of updated associations by “Backtrack” et “ForwardChecking”

- Complexity: worst case \(O(m^n n^2)\) (\(n\) ordre de \(X\), \(m\) de \(X'\), \(n < m\))
Overview

1. Definitions and representation models

2. Single graph methods
   • Segmentation or labeling and graph-cuts
   • Graphs for pattern recognition

3. Graph matching
   • Graph or subgraph isomorphisms
   • Error tolerant graph-matching
   • Approximate algorithms (*inexact matching*)
Error tolerant graph-matching

- Real world: noisy graphs, incomplete graphs, distortions
- Distance between graphs (editing, cost function,...)
- Sub-graph isomorphism with error tolerance: search of the sub-graph $G'$ with the minimum distance to $G$
- Optimal algorithms: A*
- 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
Decomposition in common sub-graphs

Messmer, Bunke

G1

G2

G3

F. Tupin - Graphes – p.72/91
Example

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

Model graph

\[ \mu(x) = \text{(portion de plan)} \]
\[ \mu(y) = \text{(segment de droite, horizontal=1)} \]
\[ \nu(g) = \text{(borde=1, plus haut que=1, moins haut que=0)} \]
Example - building reconstruction

Model graph and data graph matching
Example - building reconstruction

Model graph and data graph matching
Overview

1. Definitions and representation models

2. Single graph methods
   • Segmentation or labeling and graph-cuts
   • Graphs for pattern recognition

3. Graph matching
   • Graph or subgraph isomorphisms
   • Error tolerant graph-matching
   • Approximate algorithms (inexact matching)
Matching with geometric transformation

- Graph = representation of the spatial information
- Matching = computation of the geometric transformation
  - polynomial deformation
  - elastic transformation (morphing)
- Matching approaches:
  - translation: maximum of correlation
  - Hough transform (in the parameter space)
  - RANSAC method: select randomly a set of matching points, compute the transformation, compute the score (depends on the number of matched pairs for the transformation)
  - AC-RANSAC: RANSAC + a contrario framework reducing the number of parameters (NFA to be set)
Example - MAC-RANSAC (PhD Julien Rabin)

(a) Paire d’images analysée.

(b) Reconnaissance de chacun des objets superposés.
Example - MAC-RANSAC (PhD Julien Rabin)
Inexact matching

Optimization of a cost function

• Dissimilarity cost between nodes

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

• Dissimilarity cost between edges

\[ C_E((a_1^D, a_2^D), (a_1^M, a_2^M)) = \sum \beta_j d(a_j^A(a_1^D, a_2^D), a_j^A(a_1^M, a_2^M)) \quad \sum \beta_j = 1 \]

• Matching cost function \( h \):

\[ f(h) = \frac{\alpha}{|N_D|} \sum_{a_D \in N_D} c_N(a_D, h(a_D)) + \frac{1 - \alpha}{|E_D|} \sum_{(a_1^D, a_2^D) \in E_D} c_E((a_1^D, a_2^D), (h(a_1^D), h(a_2^D))) \]

Optimization methods:

• Tree search
• Expectation Maximization
• Genetic algorithms
• ...

F. Tupin - Graphes – p.81/91
Example: brain structures (A. Perchant)
Example: face structures (R. Cesar et al.)
Spectral method for graph matching (1)

Optimization of a cost function

- weighted adjacency matrix $M$
- nodes = potential assignments $a = (i, i')$ (can be selected by descriptor matching)
- edges = $M(a, b)$ agreement between the pairwise matchings $a$ and $b$ (geometric constraints)
- correspondance problem = finding a cluster $C$ of assignments maximizing the inter-cluster score $S = \sum_{a, b \in C} M(a, b)$ with additional constraints
- cluster $C = \text{vector } x$ (with $x(a) = 1$ if $a \in C$ and 0 else)

\[
S = \sum_{a, b \in C} M(a, b) = x^T M x
\]

\[
x^* = \text{argmax} (x^T M x)
\]

+ constraints (one to one mapping)
Spectral method for graph matching (2)

Search of the optimal cluster

• number of assignments
• inter-connection between the assignments
• weights of the assignment

Spectral method: relaxation of the constraints on $x$

$$x^* = \text{principal eigenvector}(x^T M x)$$

+ introduction of the one-to-one correspondance constraints (iterative selection of $a^* = \arg\max_{a \in L} (x^*(a))$

and suppression in $x^*$ of the incompatible assignments)
**Example: point matching (Leordeanu, Hebert)**

\[ d_{ab} = \frac{d_{ij} + q}{d_{i',j'} + q} \]

\[ \alpha_{ab} = \text{angle between the matchings} \]
(with centring and normalization)

\[ M(a, b) = (1 - \gamma)c_\alpha + \gamma c_d \]

F. Tupin - Graphes – p.86/91
Example: feature matching (Leordeanu, Hebert)
Example: factorized graph matching (Zhou, de la Torre)
Spatial reasoning in images

(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).
Spatial reasoning in images

(a)

(b)

(c)
References

Bibliography

- *A spectral technique for correspondence problems using pairwise constraints*, Leordeanu and Hebert, ICCV, 2005
- *A statistical approach to the matching of local features*, Rabin, Gousseau, Delon, SIAM Imaging science, 2009