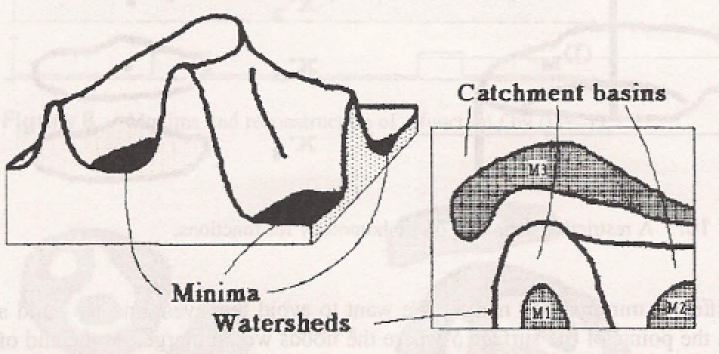
Graph and Watershed

Gianni FRANCHI

23/04/2020

Watershed for segmentation of images [1]:

Consider an image as a topographic surface composed of catchment basins and edges. Now consider that we flood the topographic surface from each minimum with constant vertical speed water. The watershed lines are the lines where the water is coming from two or more floods.



Problem: Oversegmentation

The construction of the watershed lines leads to severe over segmentations.
This may be corrected thanks to:

- preprocessing the images;
- using seeded watershed;
- using a graph based approach;
- using a stochastic watershed;

Graph applications

A graph is a data structure which is used for various applications:

- Social Networks (Facebook) use graphs to represent each user and their activities.
- Page rank is based on graph theory
- Recommendations on e-commerce websites: use graphs
- segmentation of images can be done using graphs

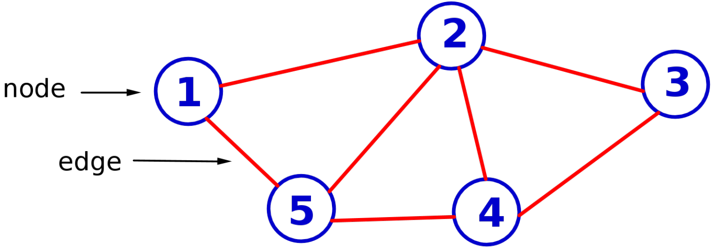
Why do we need graphs?

- Mathematically simple representation with multiple applications
- A lot of work has been done on graph theory

Graph Theory Basics

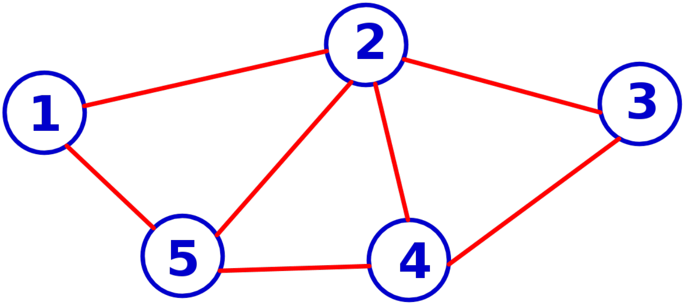
A graph is a data structure that is defined by two components :

- edges
- nodes (vertices)



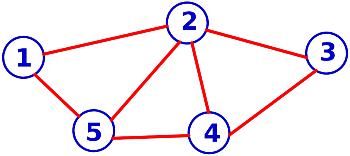
Graph Theory Basics

We write a graph $G = (V, E)$ where V is the set of nodes E is the set of edges. $E \subseteq \{(x, y)|(x, y) \in V^2\}$
On the following case $V = \{1, 2, 3, 4, 5\}$ and
$E = \{(1, 2), (1, 5), (2, 5), (2, 4), (2, 3)\}$

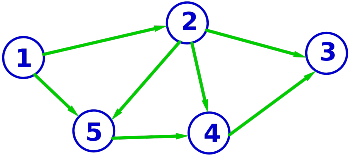


Directed/Undirected graphs

First, let us make a distinction between a directed graph and an undirected graph. A directed graph or digraph is a graph in which edges have orientations, while it doesn't for an undirected graph.



Undirected

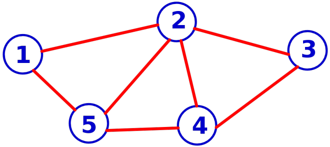


Directed

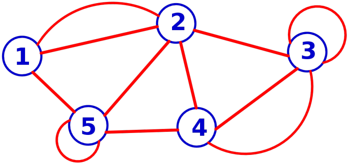
We will focus on undirected graphs.

Simple graphs / Multigraphs

A simple graph is a graph without any loop in which two nodes are connected by at most one edge.
A multigraph is a graph that can have multiple edges that have the same nodes. Thus two nodes may be connected by more than one edge. One node can also have a self-loop.



Simple graph

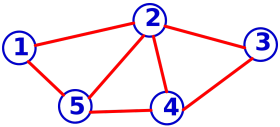


Multigraph

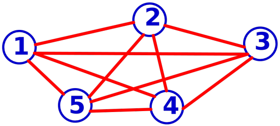
We will focus on simple graphs.

Connected, Complete, Bipartite graphs

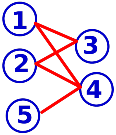
A connected graph is a graph composed of at least one vertex and there is a path (=finite or infinite sequence of edges which joins two nodes) between every pair of nodes.
A complete graph is a graph whose each nodes are connected to all other nodes .
A bipartite graph is a graph whose vertices can be divided into two disjoint and independent sets U and V such that all edges connect a node in U to one in V .



Connected graph



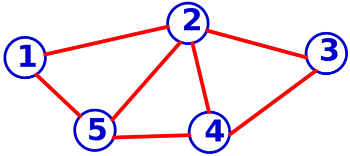
Complete graph



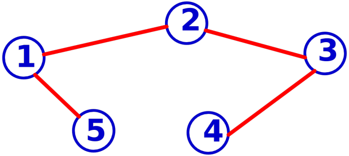
Bipartite graph

Cycle, tree graphs

For an undirected graph: A cycle graph is a graph that has at least one path (at least 3 nodes) that start and finish on the same node.
A tree is an undirected graph G is connected and acyclic (contains no cycles).
A spanning tree of an undirected graph G is a tree that includes all the nodes of G, with a minimum possible number of edges.



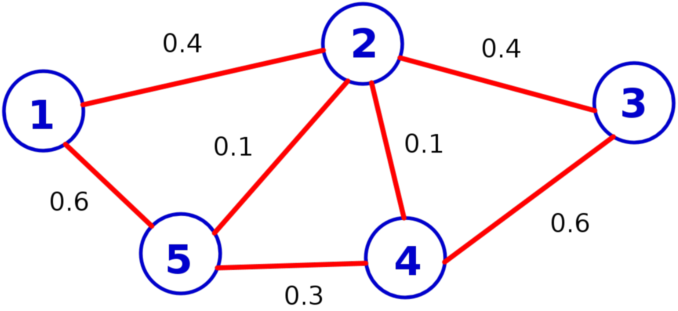
Cycle graph



Tree

A weighted graph

Let us consider a undirected simple graph $G = (V, E)$ where V is the set of vertices E is the set of edges. $E \subseteq \{(x, y)|(x, y) \in V^2\}$
Now let us consider a weighted graph : a graph whose edges have weights that we denote (G, w), with $w : E \longrightarrow \mathbb{R}$



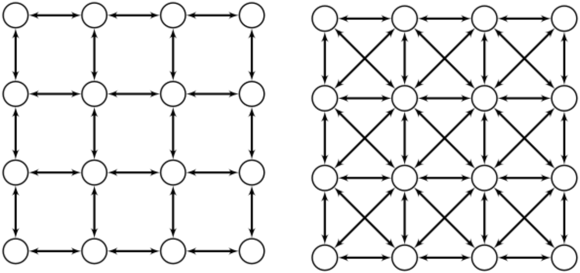
Images as graphs

- Pixel adjacency Graphs (PAG)
- Region Adjacency Graphs (RAG)

Pixel adjacency Graphs (RAG)

We consider as a weighted undirected graph the PAG.

- V is the set of pixel of the image
- E corresponds to an adjacency relation on V
- w is a measure of dissimilarity between the pixel



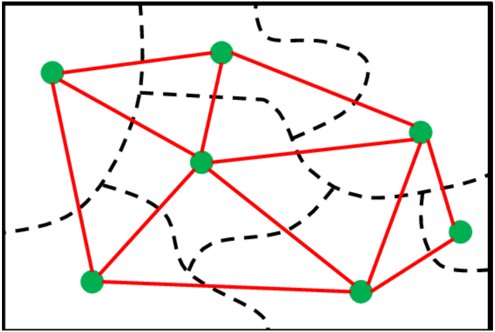
(a) (b)

Figure: (a)A 4-connected pixel adjacency graph, (b) a 8-connected pixel adjacency graph.

Region Adjacency Graphs (RAG)

We consider as a weighted undirected graph the RAG.

- V is the set of clusters of the image
- E corresponds to an adjacency relation on V
- w is a measure of dissimilarity between the clusters



Region Adjacency Graphs (RAG): the nodes

Figure: superpixels [8]

Region Adjacency Graphs (RAG): the nodes

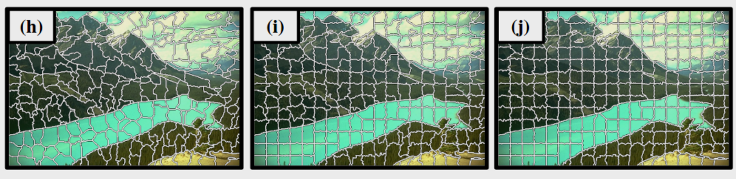


Figure: Watershed vs superpixels [8]

Region Adjacency Graphs (RAG): the edges

- We associate each edge $e \in E$ with a real valued, non-negative weight, $w(e)$;
- The weight of an edge represents the similarity ,or the dissimilarity between the nodes connected by edges;
- For example, we may define the edge weights as $w(e_{ij}) = |I(v_i) - I(v_j)|$ with v_i and v_j two nodes of V and e_{ij} the edge between these nodes, and $I(v_i)$ the mean value of the cluster i.
- For example, we may define the edge weights as $w(e_{ij}) = e^{-\frac{\|I(v_i)-I(v_j)\|^2}{\sigma}}$ with $\sigma \in \mathbb{R}^{+*}$

From RAG to watershed [2]

Given that the flooding of the watershed transform always follows the path of minimum height we can transform the watershed partition into a RAG by using as the weight between two regions the minimum pass point of the gradient along the frontier of these regions.

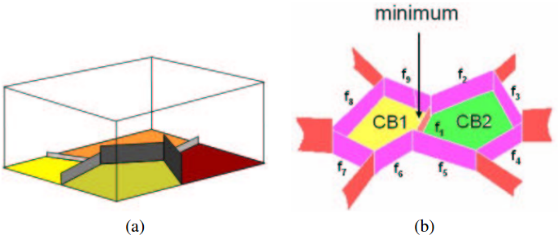
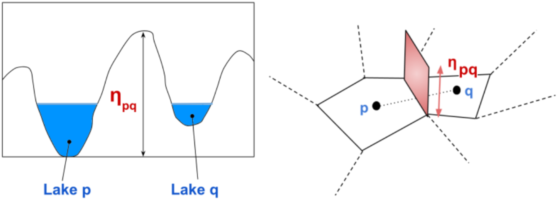


Figure: (a)Partition with valuated frontiers (b) example of frontiers

Then [2] proposed to link the watershed and the Minimum Spanning Tree (MST).

Minimum Spanning Tree and watershed



(a) Topographic relief flooding

(b) Graph modelization

Figure: Construction of the minimum spanning tree by flooding a topographic surface [9].

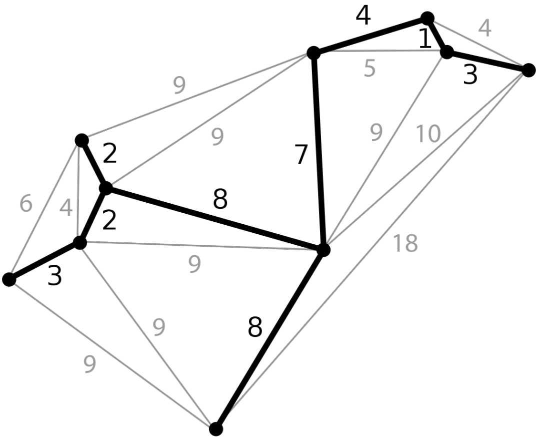
Shortest path - Minimum Spanning Tree

Definition shortest path : Given an edge-weighted graph (G, w), one of the main problems is the computation of $\text{dist}_G(x, y)$ and finding a shortest(x, y)-path among all possible paths.
Definition Minimum Spanning Tree : Given an edge-weighted graph (G, w), one of the main problems is to find: a spanning tree T of G with minimum weight, i.e. for which $\sum_{e \in T} w(e)$ is minimum.
Search of the minimum spanning tree can be solved with:

- Kruskal algorithm $O(n^2 + mlog_2(m))$ (n is the number of vertices and m of edges)
- Prim algorithm $O(n^2)$

Important: If each edge has a distinct weight then there will be only one, unique minimum spanning tree.

Minimum Spanning Tree



Minimum Spanning Tree and watershed

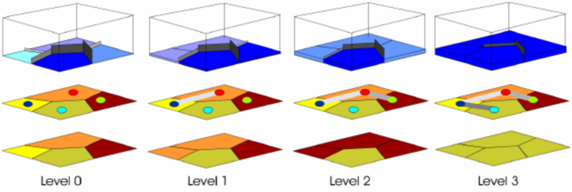
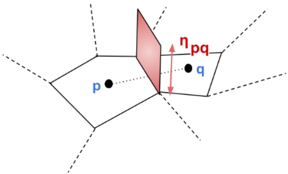
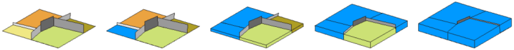


Figure: Construction of the minimum spanning tree by flooding a topographic surface [3].

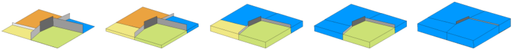
Minimum Spanning Tree and watershed



(a) Graph model



(b)



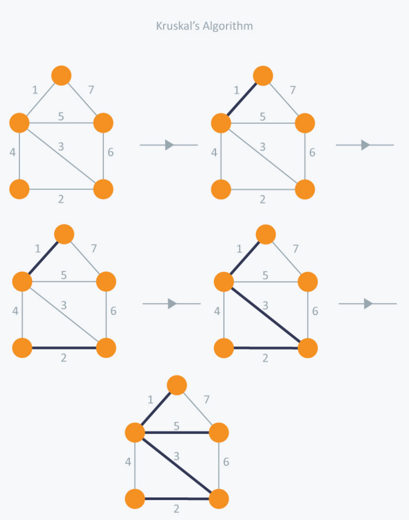
(c)

Figure: Construction of the minimum spanning tree by flooding a topographic surface [9].

Kruskal algorithm:

1. Sort the graph edges with respect to their weights.
2. Start adding edges to the MST from the edge with the smallest weight until the edge of the largest weight.
3. Only add edges which doesn't form a cycle , edges which connect only disconnected components.
4. repeat the previous step up to when all nodes are on the tree.

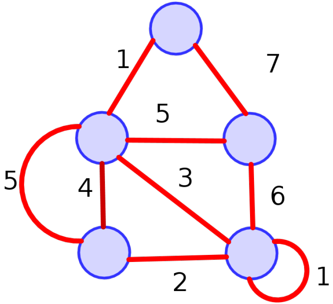
Kruskal algorithm:

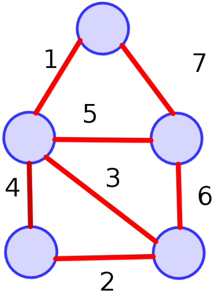


Prim algorithm:

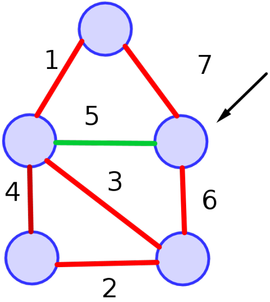
1. remove all loops and parallel edges (keep the one with minimum weight)
2. while adding new edge select the one with minimum weight out of the edges from already visited nodes. No cycle are allowed.
3. repeat the previous step up to when all nodes are on the tree.

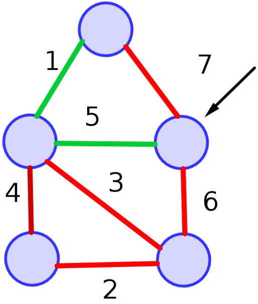
Prim algorithm:



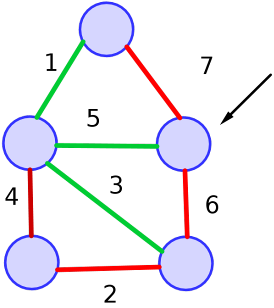


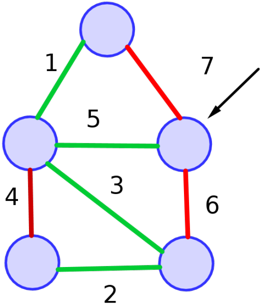
Prim algorithm:





Prim algorithm:





Minimum Spanning Tree and watershed

Then in order to obtain k partitions with $k < m$, one just need to remove the $k - 1$ biggest edges of the MST. These operations leads to a Minimum Spanning Forest (MSF).

Minimum Spanning Tree and watershed

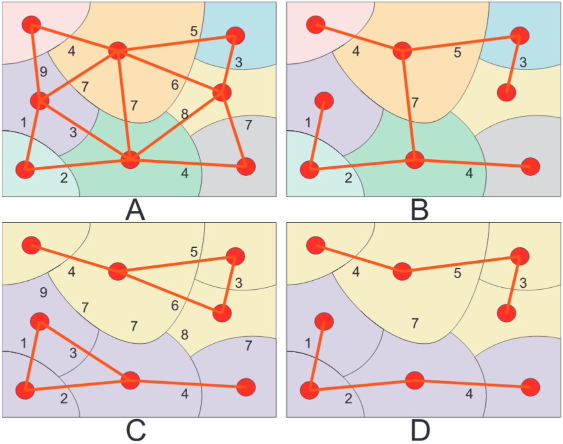


Figure: (A): a RAG, (B) : the MST of the graph, (C) two connected subgraphs, obtained by cutting all edges with a weightabove 6, (D) watershed cut if we want two clusters. [6]

Minimum Spanning Tree and watershed

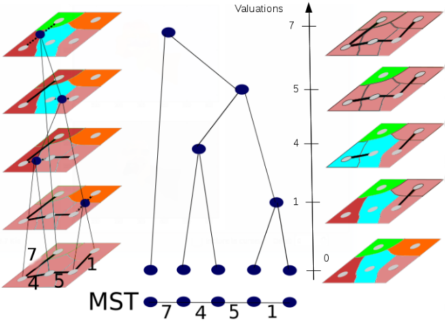


Figure 3.3 – Progressive fusion of regions as the cut level λ increases.

Figure: Construction of the partition [9].

Minimum Spanning Tree and watershed

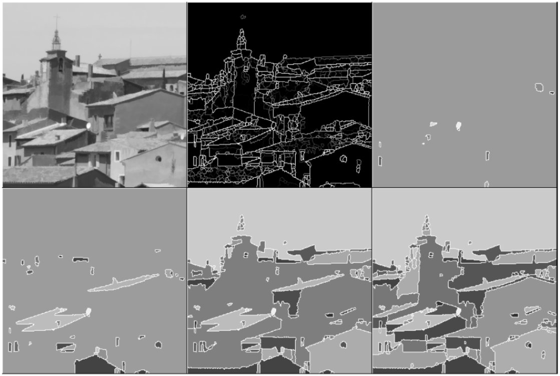


Figure: An image, its watershed partition, followed by 4 levels of hierarchy of MST cut, going from coarse to fine. [6]

Stochastic watershed

The basic idea comes from the difficulties to find the appropriate markers to have a good partition. Hence Angulo and Jeulin proposed the following algorithm [5]:

Algorithm 1: Stochastic watershed algorithm

Result: F_{out} a grey scale image of the same size than I having the probability of the boundaries

initialization: I (input image), L (nb markers), K (nb realisations);

while *step* $< K$ **do**

  generate random L markers in the image;;

  performed a marked watershed with these L markers;;

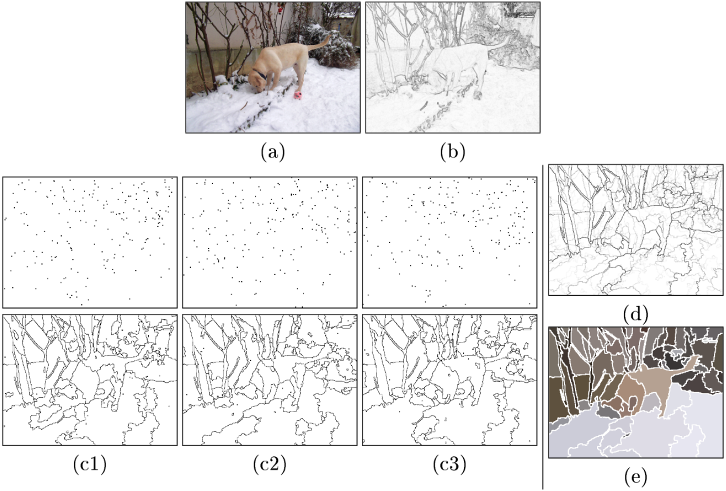
  save the frontiers of the watershed into F;

  $F_{\text{out}} = F_{\text{out}} + F$;

end

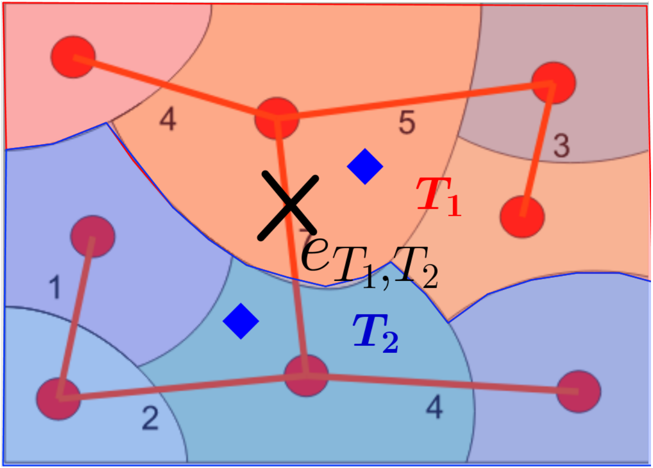
$F_{\text{out}} = F_{\text{out}}/K$

Stochastic watershed [7]



Stochastic watershed

Let us consider two markers in blue.



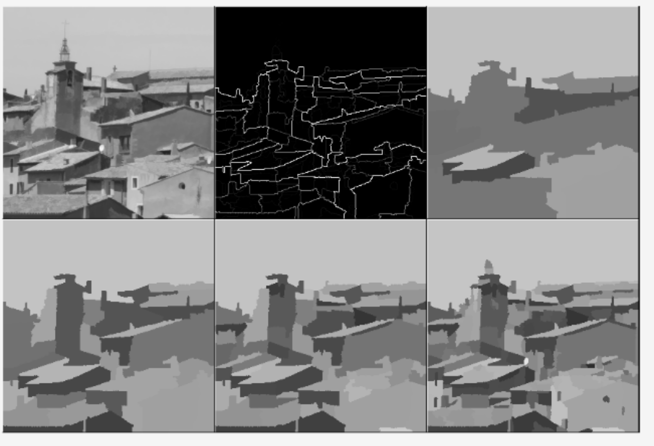
Stochastic watershed

If we consider just L markers. The edge e_{T_1,T_2} is an edge of the segmentation if and only if the tree T_1 and T_2 have each of them at least one node.
This means that at least one germ has fallen in each of the surfaces S_1 of T_1 and S_2 of T_2 spanned by both trees. Let us write S the total surface of the image.
We have to compute the probability of the event there is at least one germ in T_1 and there is at least one germ in T_2.
Its probability is the probability that there is no germ in T_1 + the probability that there is no germ in T_2 - the probability that there is no germ in $T_1 \cup T_2$.
Hence the probability of e_{T_1,T_2} to be an edge is :

$$P_{e_{T_1,T_2}} = 1 - (1 - \frac{S_1}{S})^L - (1 - \frac{S_2}{S})^L + (1 - \frac{S_1 + S_2}{S})^L \qquad (1)$$

Stochastic watershed

Let us consider two markers in blue.



bibliography:

1 S. Beucher,, & F. Meyer, (1993). The morphological approach to segmentation: the watershed transformation. Mathematical morphology in image processing, 34, 433-481.
2 B. Marcotegui, & S. Beucher, (2005). Fast implementation of waterfall based on graphs. In Mathematical Morphology: 40 Years On (pp. 177-186). Springer, Dordrecht.
3 F. Meyer, & J. Stawiaski, (2010, September). A stochastic evaluation of the contour strength. In Joint Pattern Recognition Symposium (pp. 513-522). Springer, Berlin, Heidelberg.
4 https://www.hackerearth.com/fr/practice/algorithms/graphs/minimum-spanning-tree/tutorial/
5 J. Angulo, & D. Jeulin, (2007, October). Stochastic watershed segmentation. In PROC. of the 8th International Symposium on Mathematical Morphology (pp. 265-276).
6 F. Meyer, (2015, January). Stochastic watershed hierarchies. In 2015 Eighth International Conference on Advances in Pattern Recognition (ICAPR) (pp. 1-8). IEEE.

bibliography:

7 J. Angulo, (2015, October). Statistical Gaussian model of image regions in stochastic watershed segmentation. In International Conference on Geometric Science of Information (pp. 396-405). Springer, Cham.

8 V. Machairas, E. Decenci¨¨re, & T. Walter, (2014, October). Waterpixels: Superpixels based on the watershed transformation. In 2014 IEEE International Conference on Image Processing (ICIP) (pp. 4343-4347). IEEE.

9 A. Fehri, (2018). Image Characterization by Morphological Hierarchical Representations (Doctoral dissertation).