Hands-on Morphological Processing using the Max-tree Data Structure

BY

ROBERTO SOUZA, LUIS TAVARES,
LETÍCIA RITTNER AND ROBERTO LOTUFO

OCTOBER 2016
Outline

1. Introduction
2. Basic concepts
3. Max-tree and component Tree
4. Max-tree array-based representation
5. Max-tree signature analysis
6. Interactive max-tree tool
7. Applications
8. Conclusions
1.1. Context

- The **max-tree** is a data structure that represents all connected components resulting from all upper thresholds of an image;
- It is a powerful structure that has been used in many applications such as segmentation, feature extraction, filtering, remote sensing...;
- It has also been used in interactive applications;
- The target audience of this tutorial are students and researchers interested in mathematical morphology and its application to image processing and machine learning problems.
1.2. Tutorial Goals

- Introduce the tutorial participants to the max-tree data structure;
- Present the *iamxt* toolbox. A max-tree toolbox that contains efficient implementations of many max-tree processing and visualization routines;
- Present the max-tree interactive tool;
- Introduce a methodology for developing automatic applications using the max-tree.
1.3. Tutorial Environment

- The tutorial will be developed at *Adessowiki* ([http://adessowiki.fee.unicamp.br/adesso](http://adessowiki.fee.unicamp.br/adesso)) an online collaborative environment.

- The hands-on portion of the tutorial will be build upon the *iamxt* toolbox available at GitHub ([http://github.com/rmsouza01/iamxt](http://github.com/rmsouza01/iamxt)).

- The environment will be available for the participants until **December 4th, 2016**.
Outline

1. Introduction
2. Basic concepts
3. Max-tree and component Tree
4. Max-tree array-based representation
5. Max-tree signature analysis
6. Interactive max-tree tool
7. Applications
8. Conclusions
2.1. Grayscale Image

**Definition:** It is a function:

\[ I(z) : E \rightarrow \{0, \ldots, k\}, \ E \subset \mathbb{R}^2 \text{ and } k \in \mathbb{Z} \]

- Low values are black, dark gray;
- High values are bright gray, white;

Figure. Grayscale image.
2.2. Binary Image

**Definition:** It is a function:

\[ I(z) : E \rightarrow \{0, 1\}, E \subset \mathbb{R}^2 \]

- 0 → black;
- 1 → White;

Figure. Binary image.
2.3. Threshold

- Upper threshold: \( \chi_h^\geq = \begin{cases} 1 & \text{if } I(z) \geq h \\ 0 & \text{otherwise} \end{cases} \)

- Lower threshold: \( \chi_h^< = \begin{cases} 1 & \text{if } I(z) < h \\ 0 & \text{otherwise} \end{cases} \)

Figure. Left to right: original image, upper threshold, lower threshold.
2.3. Connectivity and Connected components

Figure. Connectivity 4 (right) and connectivity 8 (left).

Figure. Connected component illustration.

How many CCs are in the figure above?
2.4. Definitions

**Definition – flat zone** - are connected components defined by pixels of the same gray-level.

**Definition - threshold set** - The threshold set is the set of connected components \{C_{h,1}, C_{h,2}, \ldots, C_{h,ncc}\}.

**Definition – Partition** – A partition P of a set X is a set of nonempty subsets of X, such that every element x in X is in exactly one of these subsets.

\[
P(X) = \{X_0, X_1, X_{n-1}\}, \quad X_i \cap X_j = \emptyset, \quad i \neq j
\]

\[
X = X_0 \cup X_1 \cup \ldots \cup X_{n-1}
\]
2.4. Definitions

Definition - Image ordering - \( f[z] \leq I[z], \ \forall z \)

Definition – anti-extensive filter – \( \psi(I) \leq I, \forall I \)

Definition – Connected filter – is a filter in which the partition of the input image flat zones is always finer than the partition of the filtered image.

\[ P_I \subseteq P_{\psi(I)}, \forall I \]
2.4. Definitions

*Definition – extinction value*: consider $M$ a regional maximum of an image $I$, and $\Psi(\psi_\lambda)_\lambda$ is a family of decreasing connected anti-extensive transformations. The extinction value $\epsilon_{\Psi}(M)$ is given by:

$$
\epsilon_{\Psi}(M) = \sup\{\lambda \geq 0 | \forall \mu \leq \lambda, M \subset \text{Max}(\psi_\mu(i))\}
$$

- Extinction values are a measure of persistence of extrema.
Outline

1. Introduction
2. Basic concepts
3. Max-tree and component Tree
4. Max-tree array-based representation
5. Max-tree signature analysis
6. Interactive max-tree tool
7. Applications
8. Conclusions
3.1. Component Tree

- The component tree is a structure for image representation that represents every connected component of every possible threshold of the image.
- It is an efficient structure for implementing connected anti-extensive filters and by duality extensive filters.
- It provides an attribute signature as means of discriminating features in the image.
3.1. Component Tree

Figure. Image \( I = [0, 5, 2, 4, 1, 1, 4, 4, 1, 0] \) (bottom) and component tree construction (top).
3.2. Max-tree

- It is a compact structure for the component tree representation.

Figure. Image I = [0,5,2,4,1,1,4,4,1,0] (bottom) and max-tree construction (top).
3.3. Component Tree vs Max-tree

Figure. Component tree (left) and max-tree (right) of the image $I = [0, 5, 2, 4, 1, 1, 4, 4, 1, 0]$. 

3. Max-tree and component tree
3.4. Filtering the max-tree

Figure. Max-tree filtering pipeline.

- Pruning strategies
- Non-pruning strategies
3.5. Area open filter

\[ \tau_1 = \psi_A(\tau_0) = \{ i \in \tau_0 \mid \text{Area}(i) \geq A \} \]

Figure. Illustration of the area open filter.
3.5. Area open filter

\[ \tau_1 = \psi_A(\tau_0) = \{ i \in \tau_0 | \text{Area}(i) \geq A \} \]

Figure. Illustration of the area open filter.
3.6. Extinction filter

$$\tau_1 = \psi_{NL,EX}(\tau_0)$$

Preserves relevant maxima.

- **NL**: number of leaves to be preserved.
- **EX**: extinction values of the volume attribute.

Figure. Illustration of the extinction filter.
3.6. Extinction filter

\[ \tau_1 = \psi_{NL,EX}(\tau_0) \]

Preserves relevant maxima.

- **NL**: 3 leaves.
- **EX**: extinction values of the volume attribute.

Figure. Illustration of the extinction filter.
Outline

1. Introduction
2. Basic concepts
3. Max-tree and component Tree
4. Max-tree array-based representation
5. Max-tree signature analysis
6. Applications
7. Conclusions
4.1. Array-based representation

- The max-tree representation we are going to use is array-based and node-oriented;
- It consists of two arrays: node array (NA) and node index (NI);
- NA is ordered in a way it is easier to perform tree traversals;
- NA minimally stores the parent relationship of the max-tree nodes and the gray level of each node. It can also store other attributes like area, bounding-box coordinates,...;
4.1. Array-based representation

Figure. Left to right: sample image. NA/NI max-tree representation of the sample image, and the node oriented max-tree illustration (node ID: h [area]).
4.2. iamxt Toolbox

- Developed by our research group with state-of-the-art algorithms;

- Accepts 2D and 3D images of integer type;

- Developed in Python/NumPy, loops optimized in C++ and wrapped with SWIG.
4.2. iamxt Toolbox

Figure. Class diagram of the *iamxt* toolbox.
4.2. iamxt Toolbox

Table. Attributes stored in node array of the *iamxt* toolbox.

<table>
<thead>
<tr>
<th>Line index</th>
<th>Attribute</th>
<th>Line index</th>
<th>Attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>par</td>
<td>7</td>
<td>xmax</td>
</tr>
<tr>
<td>1</td>
<td>nchild</td>
<td>8</td>
<td>sumy</td>
</tr>
<tr>
<td>2</td>
<td>Level</td>
<td>9</td>
<td>ymin</td>
</tr>
<tr>
<td>3</td>
<td>area</td>
<td>10</td>
<td>ymax</td>
</tr>
<tr>
<td>4</td>
<td>seed</td>
<td>11</td>
<td>sumz</td>
</tr>
<tr>
<td>5</td>
<td>sumx</td>
<td>12</td>
<td>zmin</td>
</tr>
<tr>
<td>6</td>
<td>xmin</td>
<td>13</td>
<td>zmax</td>
</tr>
</tbody>
</table>
4.2. iamxt Toolbox

Code fragment. Max-tree construction example.

```python
import iamxt
import numpy as np

# Adessowiki image reading function
img = adreadgray(<image path>)

# Structuring element connectivity C8
Bc = np.ones((3,3), dtype = bool)

# Max-tree construction
mxt = iamxt.MaxTreeAlpha(img, Bc)
```
4.2. iamxt Toolbox


```python
#number of leaves to preserve
n = 8
# Height extinction values computation
ext = mxt.computeExtinctionValues(mxt.computeHeight(), "height")
#Extinction filter
mxt.extinctionFilter(ext, n)
#MMS filter
mxt.mmsT()
#Displaying the resulting max-tree graph
mmgraphviz(mxt.generateCCGraph(parent_scale = True))
```
4.2. iamxt Toolbox

Figure. (a) Original image. (b) Result of the area extinction filter set to preserve 8 leaves. (c) Result of the maximal max-tree simplification filter.
4.2. iamxt Toolbox

Figure. Max-tree after applying the maximal max-tree simplification methodology.
4.2. iamxt Toolbox

Figure. Sample images.

<table>
<thead>
<tr>
<th>Dimensions</th>
<th>Construction</th>
<th>Filtering</th>
<th>Restitution</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>256x256</td>
<td>17.7</td>
<td>1.7</td>
<td>0.08</td>
<td>19.5</td>
</tr>
<tr>
<td>512x512</td>
<td>72.7</td>
<td>2.6</td>
<td>0.3</td>
<td>75.6</td>
</tr>
<tr>
<td>1024x1024</td>
<td>216.7</td>
<td>4.3</td>
<td>1.3</td>
<td>222.3</td>
</tr>
</tbody>
</table>

Table. Average iamxt processing times in milliseconds.
Outline

1. Introduction
2. Basic concepts
3. Max-tree and component Tree
4. Max-tree array-based representation
5. Max-tree signature analysis
6. Interactive max-tree tool
7. Applications
8. Conclusions
5.1. Signature Analysis

- The max-tree signature consists in analyzing an attribute variation starting at a leaf node and going towards the root.

Figure. Brain MR image (left) and its area signature (right).
5. Max-tree signature analysis

5.1. Signature Analysis

Figure. Node reconstruction before the sudden drop in the area signature value (right) and node reconstruction after (left).
Outline

1. Introduction
2. Basic concepts
3. Max-tree and component Tree
4. Max-tree array-based representation
5. Max-tree signature analysis
6. Interactive max-tree tool
7. Applications
8. Conclusions
6.1. Context

Simple synthetic image

Resulting max-tree with 9 nodes.

Figure. Max-tree of a simple synthetic image.
6.1. Context

Simple synthetic image

Resulting max-tree in dendrogram.

Max-tree of a simple synthetic image.
6.1. Context

Figure. Max-tree of a simple synthetic image.
6.2. Motivation

Figure. Max-tree of a natural image.
6.2. Motivation

- The development of a method to view a significant graphic representation of the max-tree;

- The development of interactive features to enable manipulation of nodes and attributes of the max-tree;

- It was the first proposal for an interactive graphical representation of the max-tree.
6.3. Contributions of the tool

- A max-tree simplification method;
- A proposal of a new simplifier filter called area difference filter;
- A max-tree interactive graphical representation methodology;
- Interactive features that allow to manipulate max-tree nodes;
- Many applications, such as segmentation, filtering and analysis;
- Educative tool for the learning of the max-tree data structure;
- Tool that helps the planning of automatic methods.
6.4. Methodology

- Interactive max-tree process flow

Figure. Interactive max-tree methodology.
6.5. Max-tree simplification

• The simplification process aims to reduce the number of max-tree nodes while maintaining a significant representation, it preserves the most relevant nodes.

• It can be seen as a sequence of three filters, which its parameters are interactively defined by the user.
6.5. Max-tree simplification

- **Extinction filter**: most important filter of simplification process, it drastically reduces the number of nodes, preserving the most relevant maxima and removing noise;

- **Area open filter**: is a filter more intuitive for the user, it removes nodes less than a area criterion;

- **Area difference filter**: preserves great changes of area in the image, which is typically relevant information.
6.5. Max-tree simplification

\[
\begin{align*}
\tau_b &= \psi_{NL,EX}(\tau_a) \\
\tau_c &= \psi_A(\tau_b) \\
\tau_d &= \psi_D(\tau_c)
\end{align*}
\]

- **NL**: number of leaves to be preserved;
- **EX**: extinction values for the volume attribute;
- **A**: minimum area;
- **D**: area difference criterion.
6.6. Area difference filter

\[ \tau_1 = \psi_D(\tau_0) = \{ i \in \tau_0 | \Delta_A(i) > D \} \cup \{ i \in \tau_0 | \Delta_A(k) > D \text{ where } k \in d_i \} \]

where \( d_i \) are descendants of \( i \),
\[ \Delta_A(i) = \text{Area}(p_i) - \text{Area}(i), \]
\( p_i \) is the parent of \( i \)

Figure. Illustration of the area difference filter.
6.6. Area difference filter

- Preserves great changes of the area attribute.

Figure. Illustration of the area difference filter.
6.7. Simplification method

- Sequence of filters:
  \[ \tau_b = \psi_{NL,EX}(\tau_a) \]
  \[ \tau_c = \psi_A(\tau_b) \]
  \[ \tau_d = \psi_D(\tau_c) \]

- Extinction filter, area open filter and area difference filter;

- \( \tau_a \) is the input max-tree and \( \tau_d \) is simplified tree.
6.8. Dendrogram

- The dendrogram is an appropriate representation to display hierarchical structures.
- Although we show a simplified version of the max-tree, the operations are processed in the original max-tree.

Figure. Illustration of the simplified tree.
6.9. First simplification approach

Simplification based on extinction values preserving 362 nodes. 1% of max-tree nodes.

Figure. Simplification based on extinction. values.
6.10. Current simplification approach

Simplification based on the extinction filter (NL=6), area open filter (A=50) and area difference filter (D=34), preserving 357 nodes. 

~1% of the max-tree nodes.
6.11. Reconstruction of the max-tree

Figure. Reconstruction of the dendrogram nodes.
6.12. User interaction

Nodes selection and visualization.

Figure. Interactive dendrogram and some node samples.
6.13. Indirect node selection

- Select a node by clicking on the input.

Figure. Illustration of the indirect node selection.

- Zoom descendants operation
- Max-tree navigation
- Recursive feature

Figure. Zoom operation.

- Branch+ operation
- Max-tree navigation
- Recursive feature

Figure. Branch+ operation.
6.15. Creating sub-sets of nodes

- Creating sub-sets of nodes for segmentation, filtering and analysis tasks.

Figure. The reconstruction of a sub-set of max-tree. nodes.
6.16. Segmentation

- Segmenting the carotid artery wall from an MR image.

Figure. Segmentation of the carotid artery wall.
6.17. Association with other technique

- Nodes subsets as markers for the watershed.

Figure. Selecting markers for the watershed transform.
6.18. Filtering

- Local filtering.

Figure. Illustration of a filtering task.

6. Interactive max-tree tool
6.19. Visualization of attributes

- Gray-level;
- Area;
- Volume;
- **Rectangularity ratio**: area of the connected component divided by the area of its bounding box;
- **Eccentricity**: how elongated it is;
- **MSER**: area stability measure.
6.19. Visualization of attributes

- Input image and the color scale used to demonstrate the attribute values.

Figure. Color scale

Figure. Input image.
6.19. Visualization of attributes

- Analysing eccentricity.

Figure. Samples of nodes with high eccentricity values.
6.19. Visualization of attributes

- Analysing eccentricity.

Figure. Samples of nodes with low eccentricity values.
6.19. Visualization of attributes

- Evolution of the eccentricity through the gray levels.

Figure. Evolution of eccentricity values in a sub-branch.
6.19. Visualization of attributes

- 1st depth image and its dendrogram with eccentricity values.

Figure. Eccentricity values of the depth image 1.
6.19. Visualization of attributes

- 2nd depth image and its dendrogram with eccentricity values.

Figure. Eccentricity values of the depth image 2.
6.19. Visualization of attributes

- 3rd depth image and its dendrogram with eccentricity values.

Figure. Eccentricity values of the depth image 3.
6.19. Visualization of attributes

- 4th depth image and its dendrogram with eccentricity values.
6.19. Visualization of attributes

- 5th depth image and its dendrogram with eccentricity values.

Figure. Eccentricity values of the depth image 5.
6.19. Visualization of attributes

- 6th depth image and its dendrogram with eccentricity values.

Figure. Eccentricity values of the depth image 6.
6.19. Visualization of attributes

- First sub-branch of the 6th depth image.

Figure. Analysing the first sub-branch of the depth image 6.
6.20. Max-tree of 3D images

- 3 views for the user navigate through the slices of the image.

Figure. CT of the lungs
6.20. Max-tree of 3D images

- Max-tree of the CT image of the lungs.

Figure. Samples of max-tree nodes of the CT image.
6.21. Processing time

- The processing time was computed by averaging five user interactions. The max-tree was built from an image with 256 x 256 pixels.

<table>
<thead>
<tr>
<th>Operation</th>
<th>Timing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Building the max-tree</td>
<td>0,021 s</td>
</tr>
<tr>
<td>Max-tree simplification</td>
<td>0,035 s</td>
</tr>
<tr>
<td>Generation of the initial dendrogram</td>
<td>1,508 s</td>
</tr>
<tr>
<td>Updating the dendrogram interface</td>
<td>0,143 s</td>
</tr>
<tr>
<td>Reconstruction of a single node (binary image)</td>
<td>0,003 s</td>
</tr>
<tr>
<td>Reconstruction of a node and its descendants (gray-scale)</td>
<td>0,008 s</td>
</tr>
</tbody>
</table>

Table. Timing samples.
6.22. Conclusions

- Educative tool for the learning of max-tree data structure;

- Tool to analyze the nature of an image through the visualization of many attributes;

- It allows to manipulate the max-tree nodes by browsing, selecting and visualizing the max-tree nodes.
6.22. Conclusions

- Potential to be used in segmentation and filtering tasks.
- It can be used in association with other techniques;
- Contribution of the area difference filter and the simplification process, which can also be a pre-processing step for operations on the max-tree;
6.22. Conclusions

• The 3D extension of the interactive max-tree allows to work with medical images such as CT and MRI;

• The visualization of attributes provides insight to develop automatic methods.

• The tool is a web-based application which has many features, is intuitive and has good usability.
6.23. Demonstration

Outline

1. Introduction
2. Basic concepts
3. Max-tree and component Tree
4. Max-tree array-based representation
5. Max-tree signature analysis
6. Applications
7. Conclusions
6.1. Applications List

- Size filtering and shape filtering
- Extraction of size and shape information of the max-tree nodes
- Max-tree signature
- Attribute profile
- Marker selection
- Maximal max-tree simplification methodology
- Suggestions from the audience?
Outline

1. Introduction
2. Basic concepts
3. Max-tree and component Tree
4. Max-tree array-based representation
5. Max-tree signature analysis
6. Applications
7. Conclusions
7.1. Conclusions

- We illustrated the power of the max-tree and introduced iamxt an efficient toolbox to process it;
- Many possible applications of the max-tree were illustrated;
- Current trends:
  - There are other tree structures similar to the max-tree gaining popularity;
  - Algorithms for color images;
  - Construction of a max-tree of a max-tree.
References


References


That's all Folks!