Hierarchical data analysis: definitions and optimization

Benjamin Perret

GT GDMM
Nancy 15-16/11/2021

Plan

1. **Introduction**
2. **Hierarchical watersheds**
3. **Ultrametric fitting**
4. **Component tree loss function**
5. **Conclusion and Perspectives**

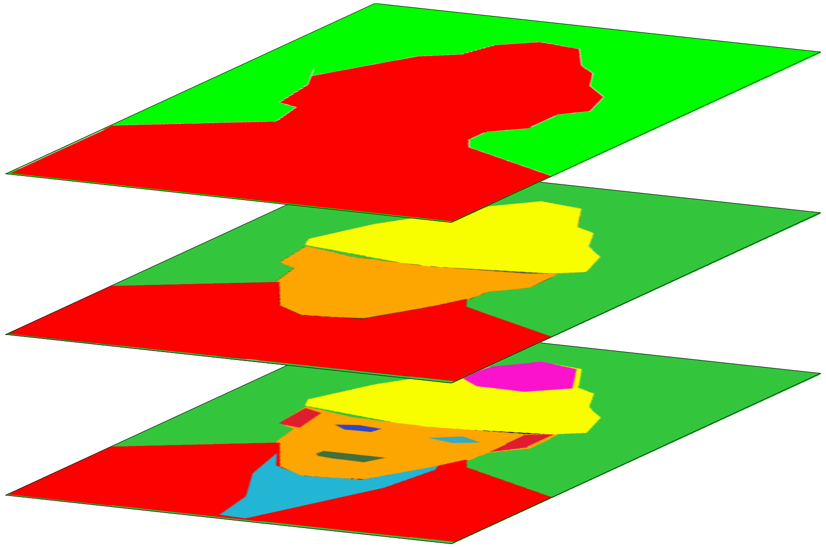
Plan

1. **Introduction**
2. **Hierarchical watersheds**
3. **Ultrametric fitting**
4. **Component tree loss function**
5. **Conclusion and Perspectives**

Hierarchical representations

- **Expectation:** iterative decomposition of the space

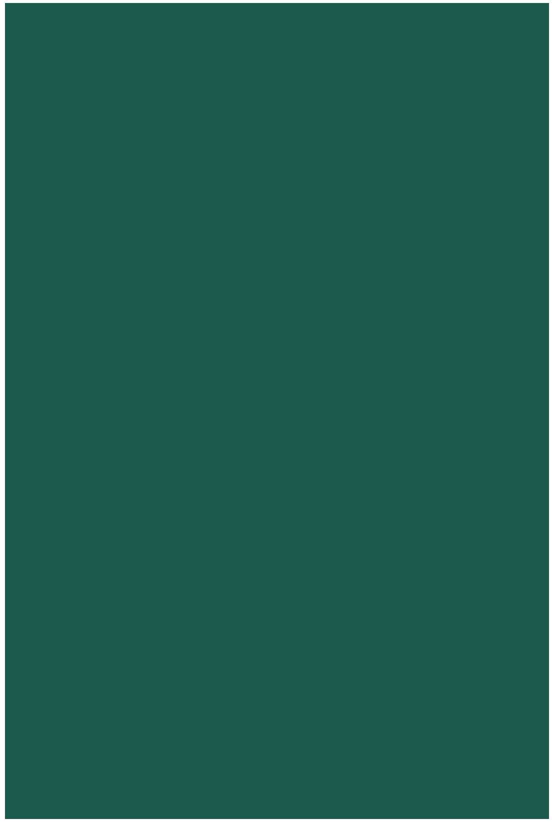
Image



Hierarchy of partitions

Hierarchical representation

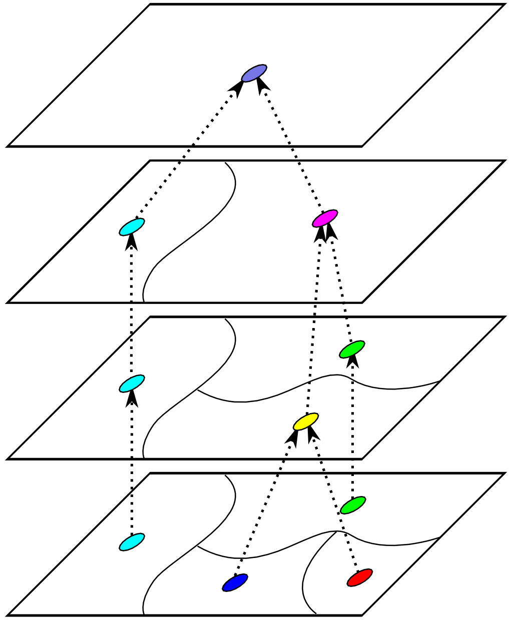
- **Reality:** imperfect representation
- **(Lower) Expectation:** interesting elements are somewhere there and easier to find than in raw pixel data



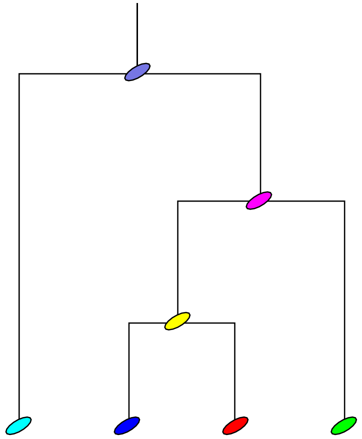
Hierarchy from a state-of-the-art method in computer vision

Classic processing scheme

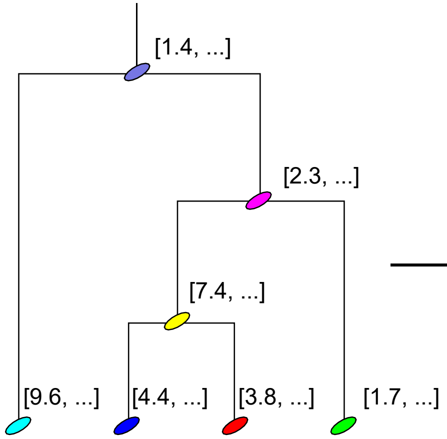
1. **Compute relevant features in tree space**
2. **Process this new structured space**



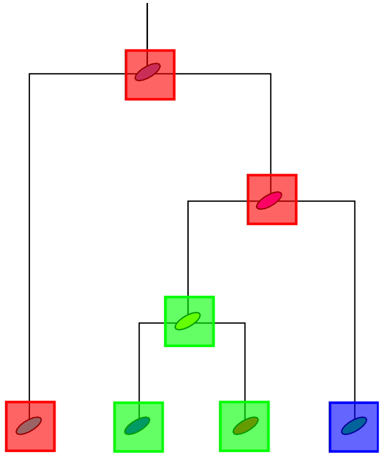
Tree: relation of inclusion between regions



Dendrogram



Node features: hand-crafted, learned



Filtering, optimisation, classification

In the tree space :
- ➢ Higher level features
- ➢ Efficient algorithms

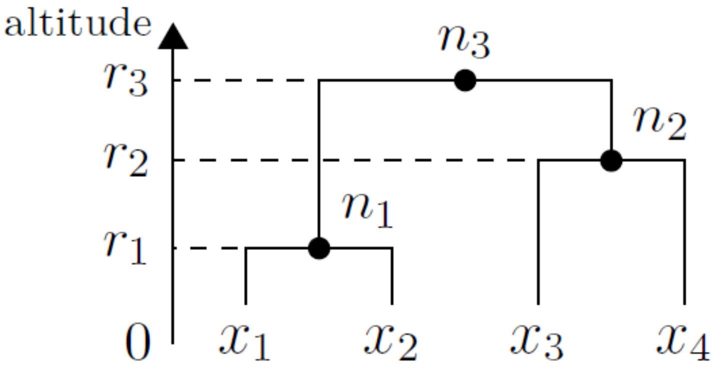
Hierarchies of segmentations

Dendrogram

- Vertex weighted tree
- Fast access to scale relation between regions
- "Combinatorial" algorithms

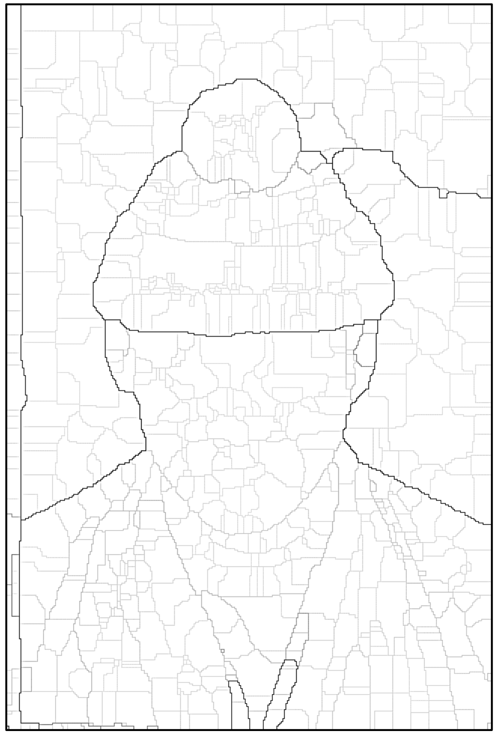
Saliency Map / Ultrametric distance

- Edge weighted graph
- Visualization
- "Numeric" algorithms



Quasi-flat zone hierarchy
(≈Single linkage clustering)

Lowest common ancestor altitude

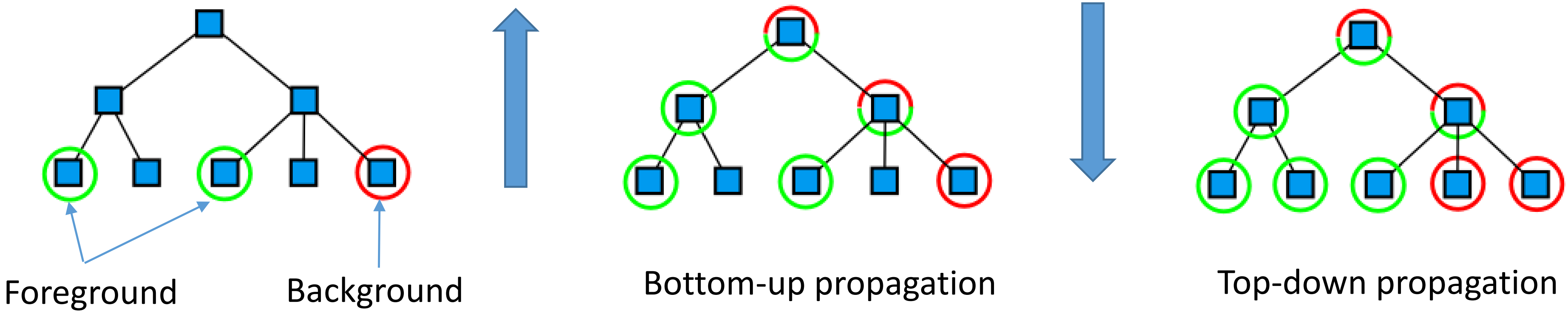


Example

Interactive segmentation

https://perso.esiee.fr/~perretb/ISeg/

- **Two markers**
  - **B: background marker**
  - **F: foreground marker**
- **Object**
  - **The largest regions of the hierarchy that intersect B but do not touch F**



Motivations

Challenges

- **Optimal hierarchies:**
  Definitions and algorithms
- **Machine learning and hierarchies:**
  Continuous optimization scheme
- **Flexible topological regularization**
  Link with topological persistence

Plan

1. Introduction
2. **Hierarchical watersheds**
3. Ultrametric fitting
4. Topological regularization
5. Conclusion and Perspectives

From clustering to hierarchical clustering?

- **Clustering/cut algorithm**
  - Implicit or explicit scale parameter
- **Do we get a hierarchy when the scale parameter varies?**
  - In most cases: no

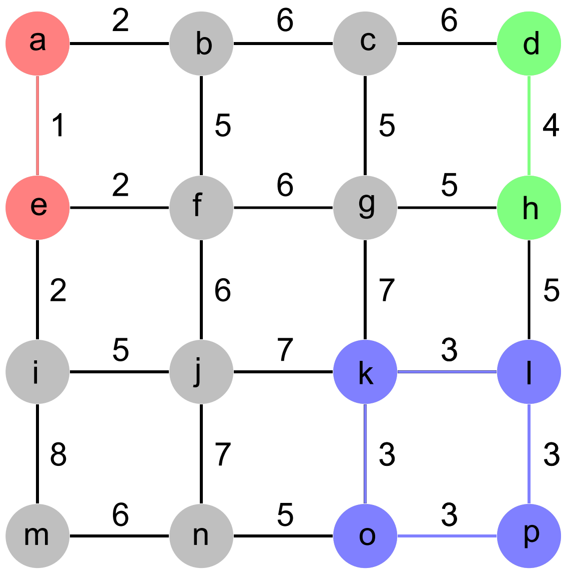
For watershed cuts: YES

- **Scale-wise optimal hierarchies**
  - Watershed hierarchies

Minima of an edge weighted graphs

A minima of the graph G is a subgraph $G' = (V', E')$ such that:

- G' is connected;
- G' has a constant weight: $\exists k, \forall e \in E', w(e) = k$; and
- all edges *surrounding* G' have a weight greater than k: $\forall \{x, y\} \in E, x \in E', y \notin E', w(e) > k$.



3 minima in red, green and blue

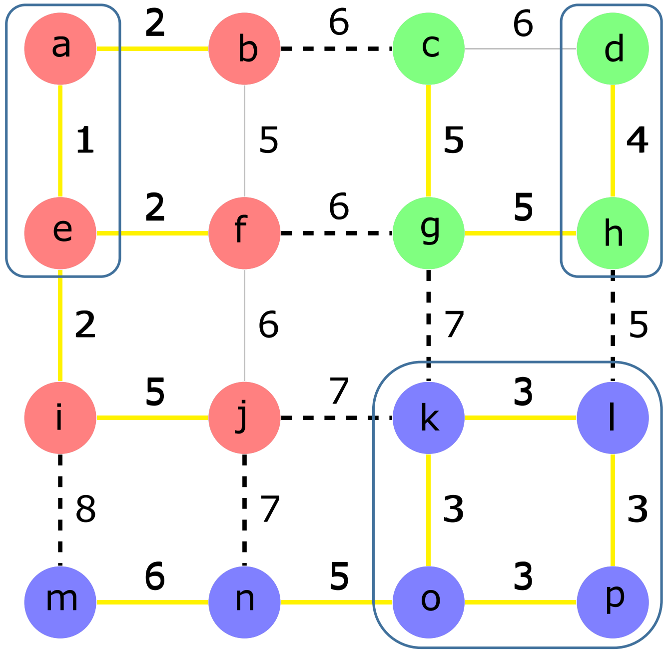
Watershed and Minimum spanning forest

A subgraph $G^* = (V, E^*)$ is a minimum spanning forest rooted in the minima of G if:

$$E^* = \arg\min_{E' \subseteq E} \sum_{e \in E'} w(e),$$

s.t. each connected component of (V, E') contains one minimum of G

Cousty et al. IEEE TPAMI 2009



Minimum spanning forest: yellow edges

Induced watershed cut: dashed edges

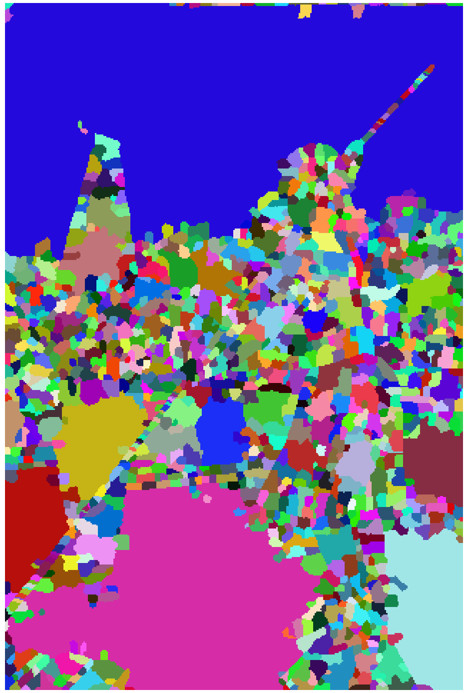
Watershed – example

Watersheds usually over-segment images

Filter minima whose measure is below a given threshold

Image

Gradient
(edge weights)



Watershed

Watershed after applying an area filter of size 1000 to the gradient

Hierarchical Watershed - idea

Watersheds usually over-segment images

Filter minima whose measure is below a given threshold

Connected filter: do not create or move contours

Give the importance of a minimum: size, depth, volume… of the associated catchement bassin

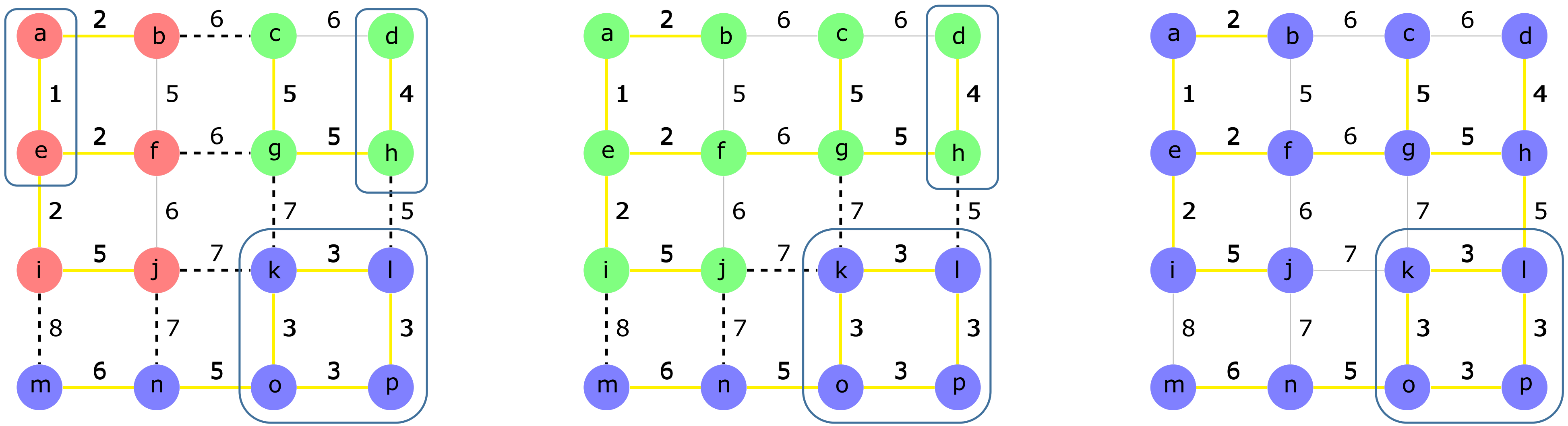
Ranks the minima according to the measure

Varying the threshold produces a sequence of fine to coarse partitions

We call it a Watershed Hierarchy

Hierarchical Watershed - formalization

Given a sequence of minima $S = (M_1, M_2, \ldots, M_n)$ of G, a hierarchical watershed of G for S is a sequence of nested watersheds WS_i, $i = 1 \ldots n$ such that WS_i is induced by a minimum spanning forest rooted in $\{M_1, \ldots, M_i\}$

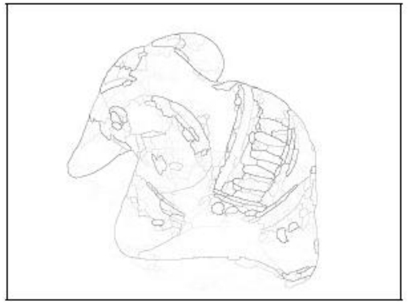


Watershed hierarchy for the sequence (●, ●, ●)

Hierarchical Watershed – examples

(a) Original

In natural image analysis:
result close to the state of the art methods
but 10x faster



(c) WS-Dynamics

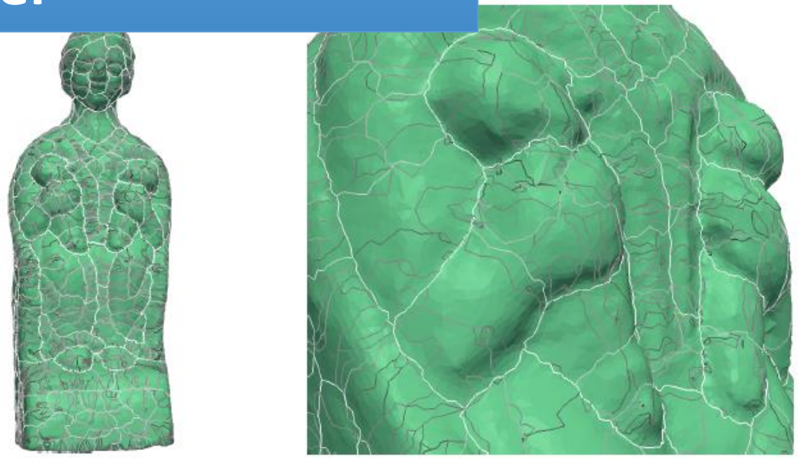
(d) WS-Area

Images

Perret et al. TIP 2018

S cities dataset

ousty et al. JMIV 2017



3D models

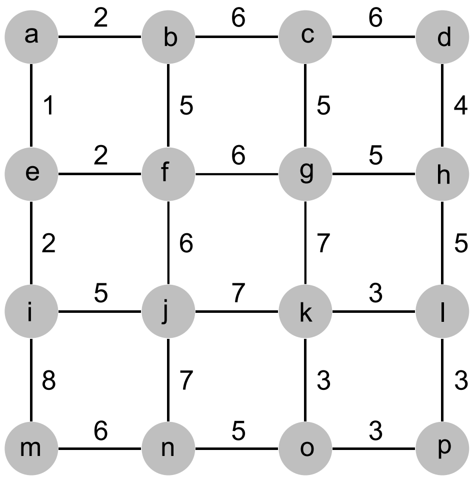
Philipp-Foliguet et al. PR 2011

Plan

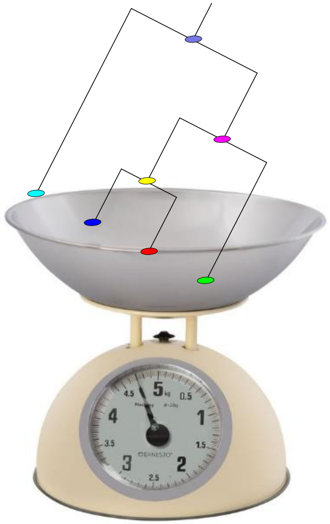
1. Introduction
2. Hierarchical watersheds
3. **Ultrametric fitting**
4. Component tree loss function
5. Conclusion and Perspectives

Ultrametric fitting - motivation

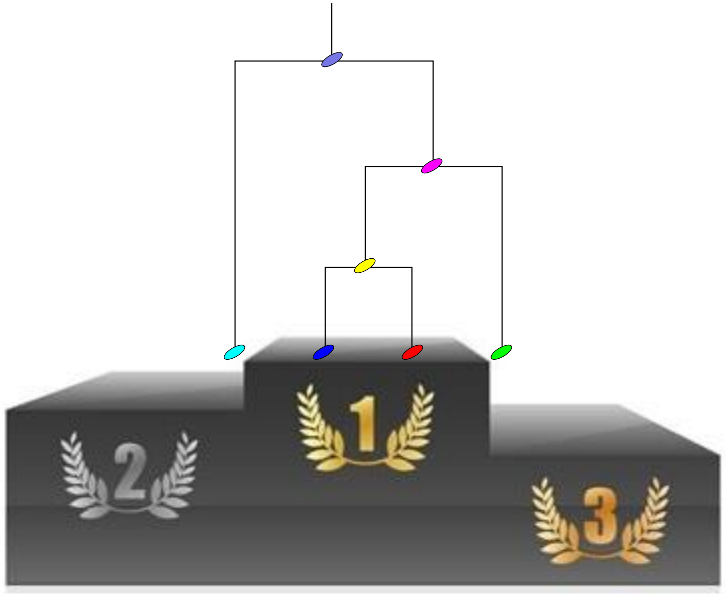
- Continuous optimization of hierarchical cost functions
- Flexible cost functions
- Integration with other machine learning methods



Input: Undirected graph with dissimilarity edge weights



Input: Hierarchical cost function



Output: Best hierarchy given as a saliency map

Associated publication: Chierchia and Perret NeurIPS 2019, Chierchia and Perret JSMTE 2020

Ultrametric fitting – Optimization problem

- **Constrained optimization problem over a continuous domain**

$$\underset{u \in \mathcal{W}}{\text{minimize}} \quad J(u; w) \quad \text{s.t.} \quad u \text{ is a saliency map}$$

Set of edge weights

Hierarchical cost function

Input dissimilarity (edge weights)

- u **is a saliency map**

$$(\forall C \in \mathcal{C}, \forall e \in C) \qquad u(e) \leq \max_{e' \in C \setminus \{e\}} u(e')$$

Set of cycles of the graph

Ultrametric fitting – Optimization problem

- **Reformulation with an implicit constraint**

$$\underset{\tilde{w} \in \mathcal{W}}{\text{minimize}} \; J\big(\Phi_{\mathcal{G}}(\tilde{w});\, w\big)$$

- **Where $\Phi_{\mathcal{G}}$ is the subdominant ultrametric operator**

$$(\forall \tilde{w} \in \mathcal{W}, \forall e_{xy} \in E) \qquad \Phi_G(\tilde{w})(e_{xy}) = \min_{P \in \mathcal{P}_{xy}} \max_{e' \in P} \tilde{w}(e')$$

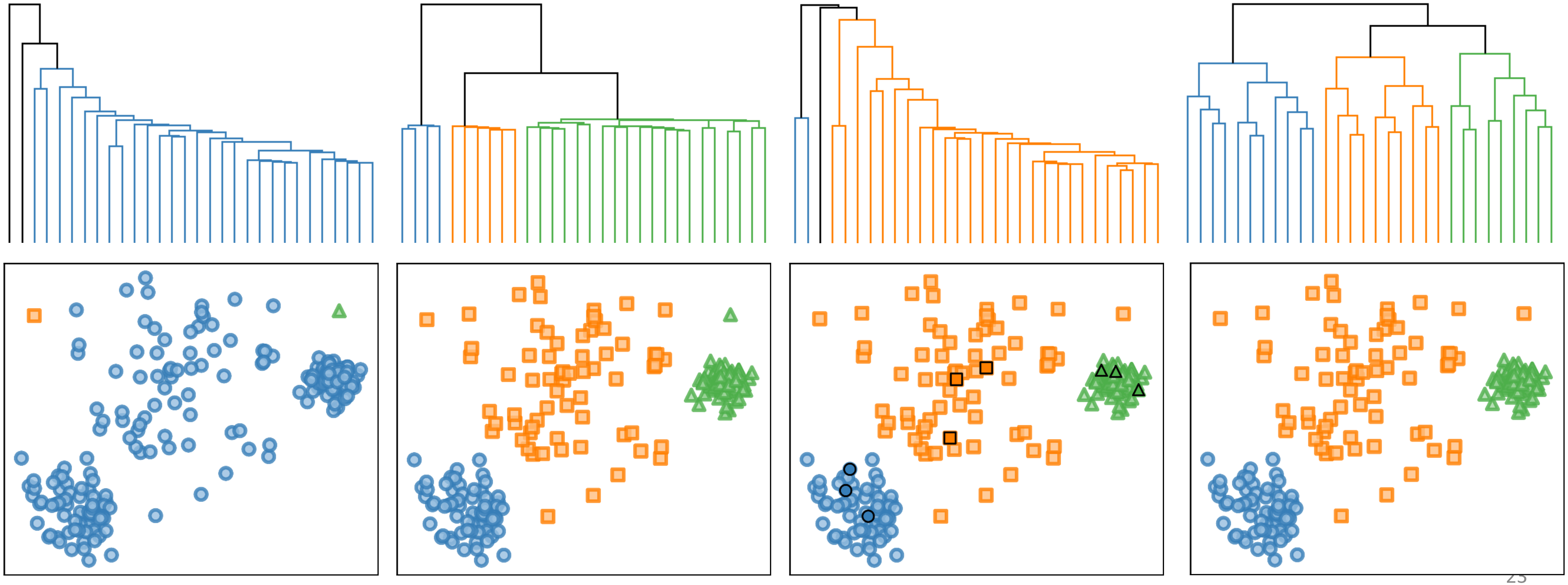
- **Optimization with a gradient descent algorithm**

Ultrametric fitting – Cost functions

- **Closest ultrametric: data fidelity term** $J_{closest}$
  L2 loss between the input edge weights and the saliency map
- **Cluster size: regularization** J_{size}
  Penalize small clusters close to the root
- **Triplet loss: semi-supervision** $J_{triplet}$
  Decrease intra-cluster distance, increase inter-cluster distance
- **Dasgupta's loss** $J_{Dasgupta}$
  Relaxation of a famous hierarchical loss (NP-hard)

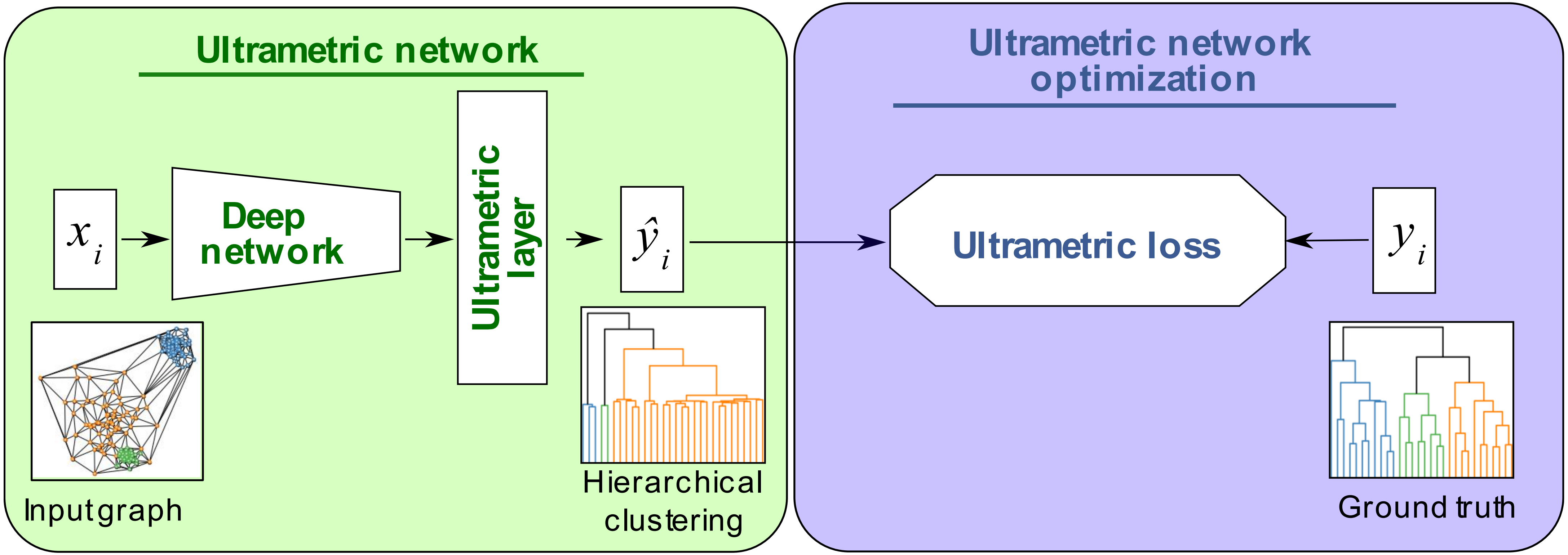
Ultrametric fitting – Examples

$J_{closest}$ | $J_{closest} + J_{size}$ | $J_{closest} + J_{triplet}$ | $J_{Dasgupta}$



Ultrametric fitting – Supervised hierarchical segmentation

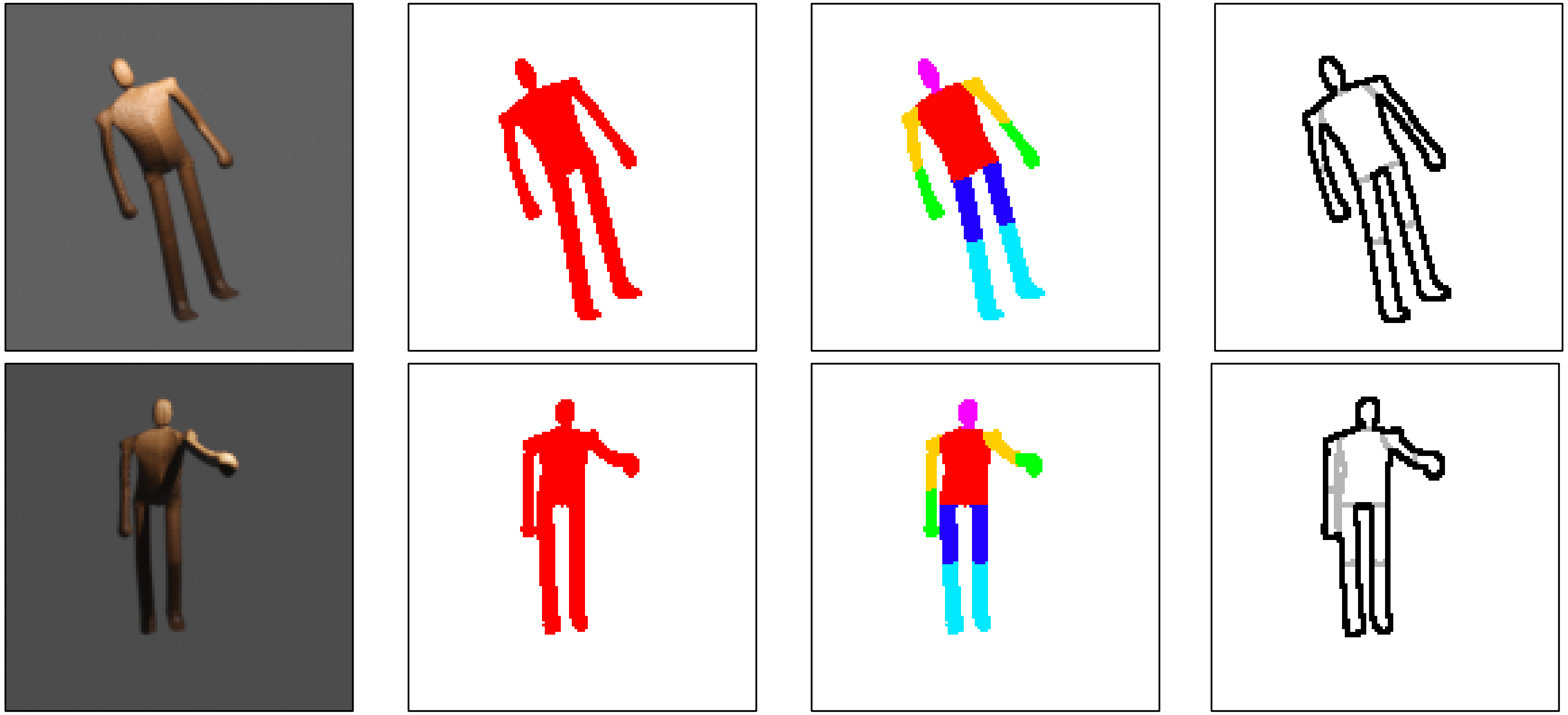
- **Structured end-to-end deep learning of hierarchical segmentation**
  - $\Phi_{\mathcal{G}}$ can be a seen as an "Ultrametric" layer



Ultrametric fitting – Supervised hierarchical clustering

- **Simulated dataset**
  - 3D human model

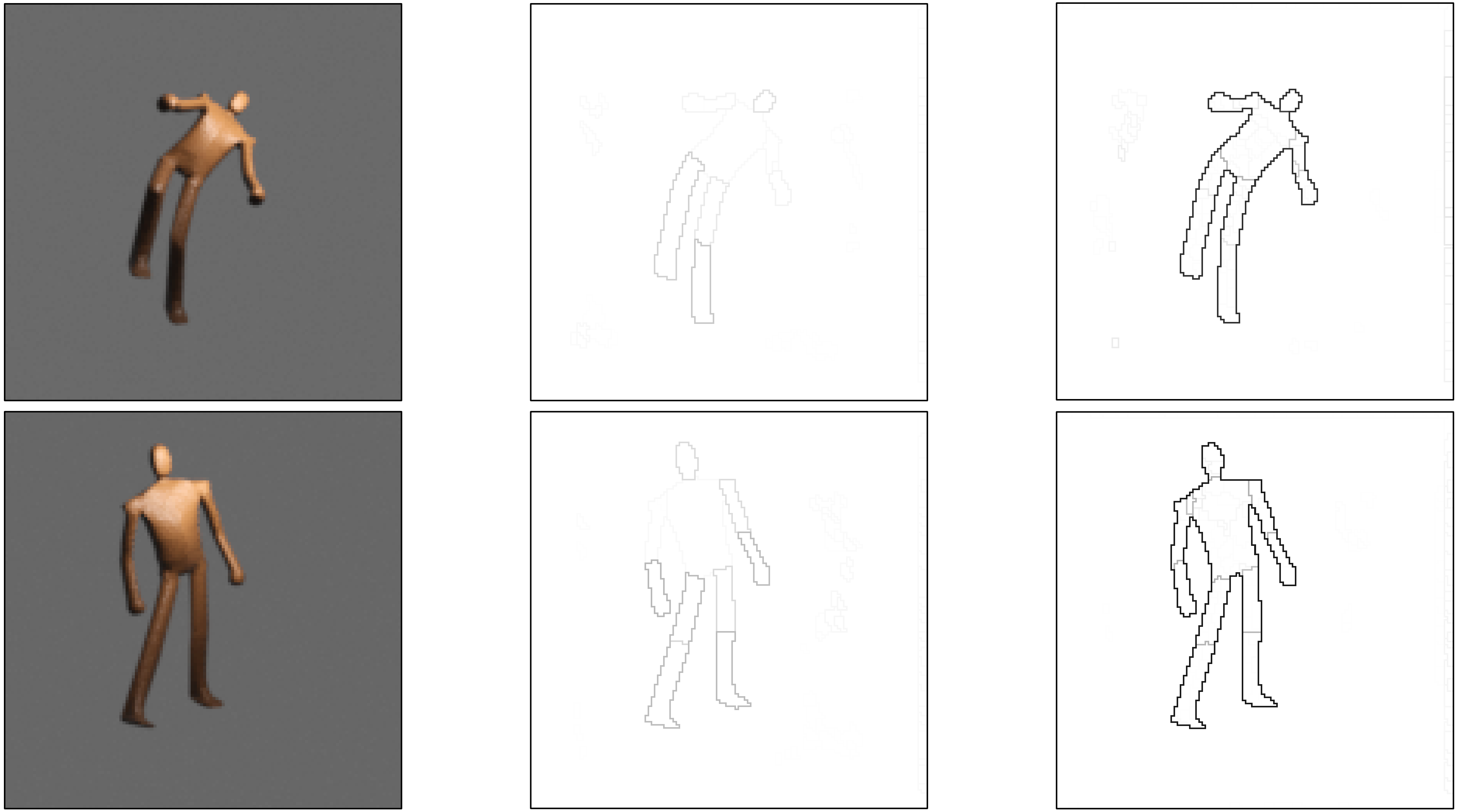
Image	Silhouette	Members	Hierarchy



Ultrametric fitting – Supervised hierarchical clustering

Results after training

Image | Without Ultrametric Layer | With Ultrametric Layer



Plan

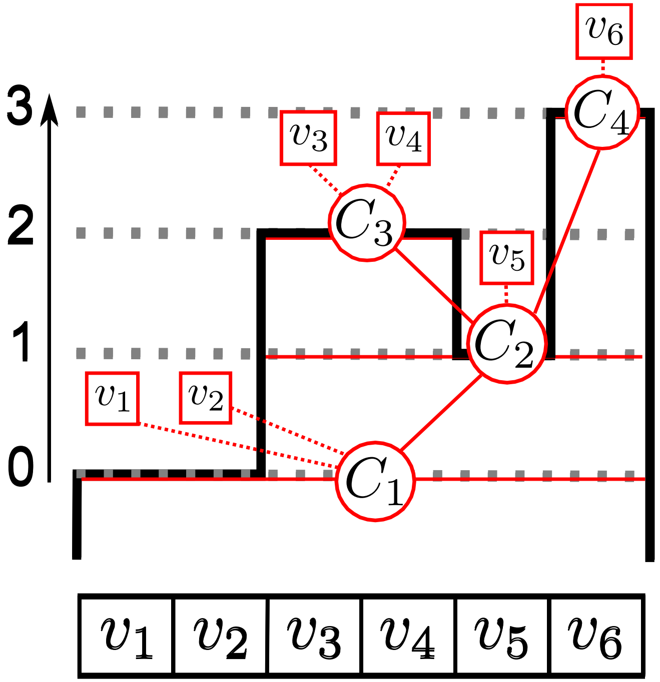
Component tree loss function - motivation

- **Component trees: max/min-trees**
- **Classical representation in mathematical morphology**
- **Differentiable loss function based on component tree**

Perret & Cousty. Component Tree Loss Function: Definition and Optimization 2021

Component tree loss function – Max-tree

- **Hierarchy of the level sets' connected components**



$$\mathbf{f} = [0, 0, 2, 2, 1, 3]$$

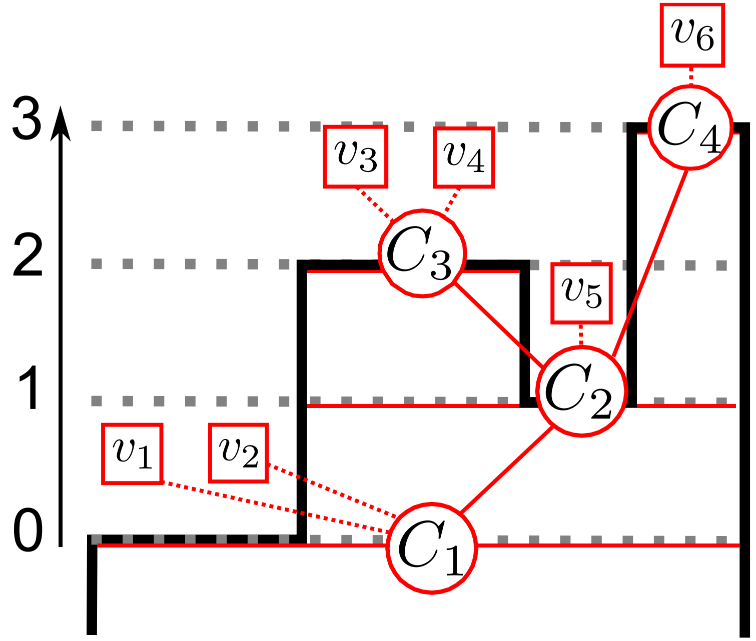
The nodes $C_1, \ldots, C_4$ **are associated to the altitude vector** $\mathbf{a} = [0, 1, 2, 3]$

Max-tree – Altitudes

Jacobian of the node altitudes

$$\frac{\partial \mathbf{a}}{\partial \mathbf{f}} = \left[\mathbb{1}_{\mathrm{par}(v_1)}, \ldots, \mathbb{1}_{\mathrm{par}(v_n)}\right]$$

- $\mathrm{par}(v)$ is the parent of the node v
- $\mathbb{1}_{C_k}$ is the column vector equals to 1 in position k and 0 elsewhere



$$\mathbf{f} = [0, 0, 2, 2, 1, 3]$$

$$\mathbf{a} = [0, 1, 2, 3]$$

Jacobian

$$\begin{array}{c} \\ \mathbf{a}_1 \\ \mathbf{a}_2 \\ \mathbf{a}_3 \\ \mathbf{a}_4 \end{array} \begin{array}{c} \begin{array}{cccccc} \mathbf{f}_1 & \mathbf{f}_2 & \mathbf{f}_3 & \mathbf{f}_4 & \mathbf{f}_5 & \mathbf{f}_6 \end{array} \\ \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \end{array}$$

Component tree loss function – Maxima Loss

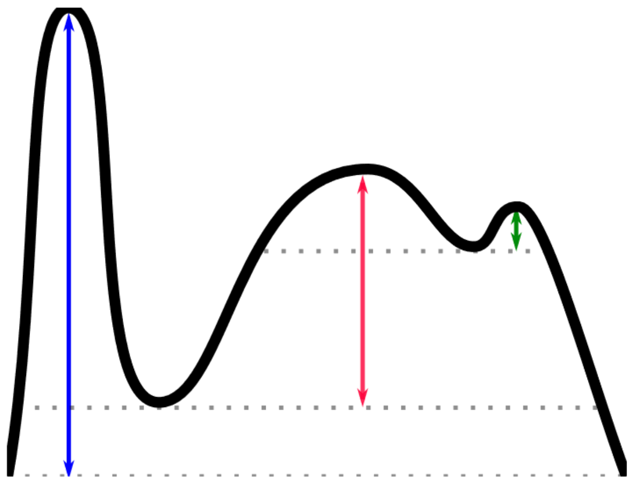
- **Select the k most interesting maxima**
  - Measure of importance: **im**
- **Reinforce selected maxima, remove others**
  - Measure of saliency: **sm**
  - Must be a function of the max-tree node altitudes

$$J_r(\mathbf{sm}, \mathbf{im}; k) = \sum_{i=1}^{i \leq k} \max(m - \mathbf{sm}_{\mathbf{r}_i}, 0) + \sum_{i=k+1} \mathbf{sm}_{\mathbf{r}_i}$$

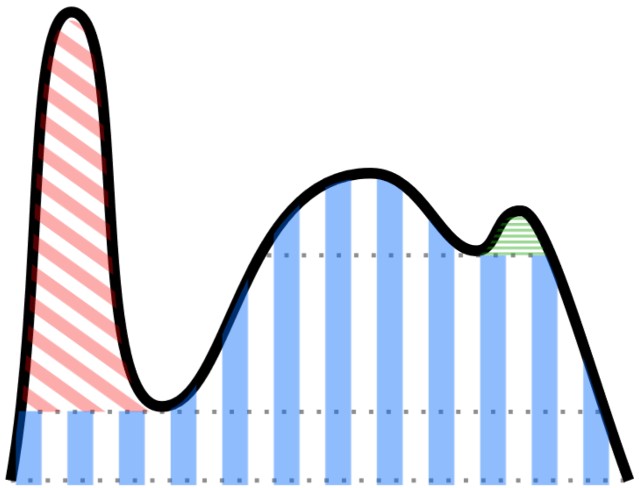
$$\text{with } \mathbf{r} = \operatorname{argsort}(\mathbf{im}),$$

Max tree – Maxima importance

- **Extinction values**
  - How long does a maxima survive during a filtering process?
  - The filtering can be based on depth, area, volume
  - The higher the value the more important it is



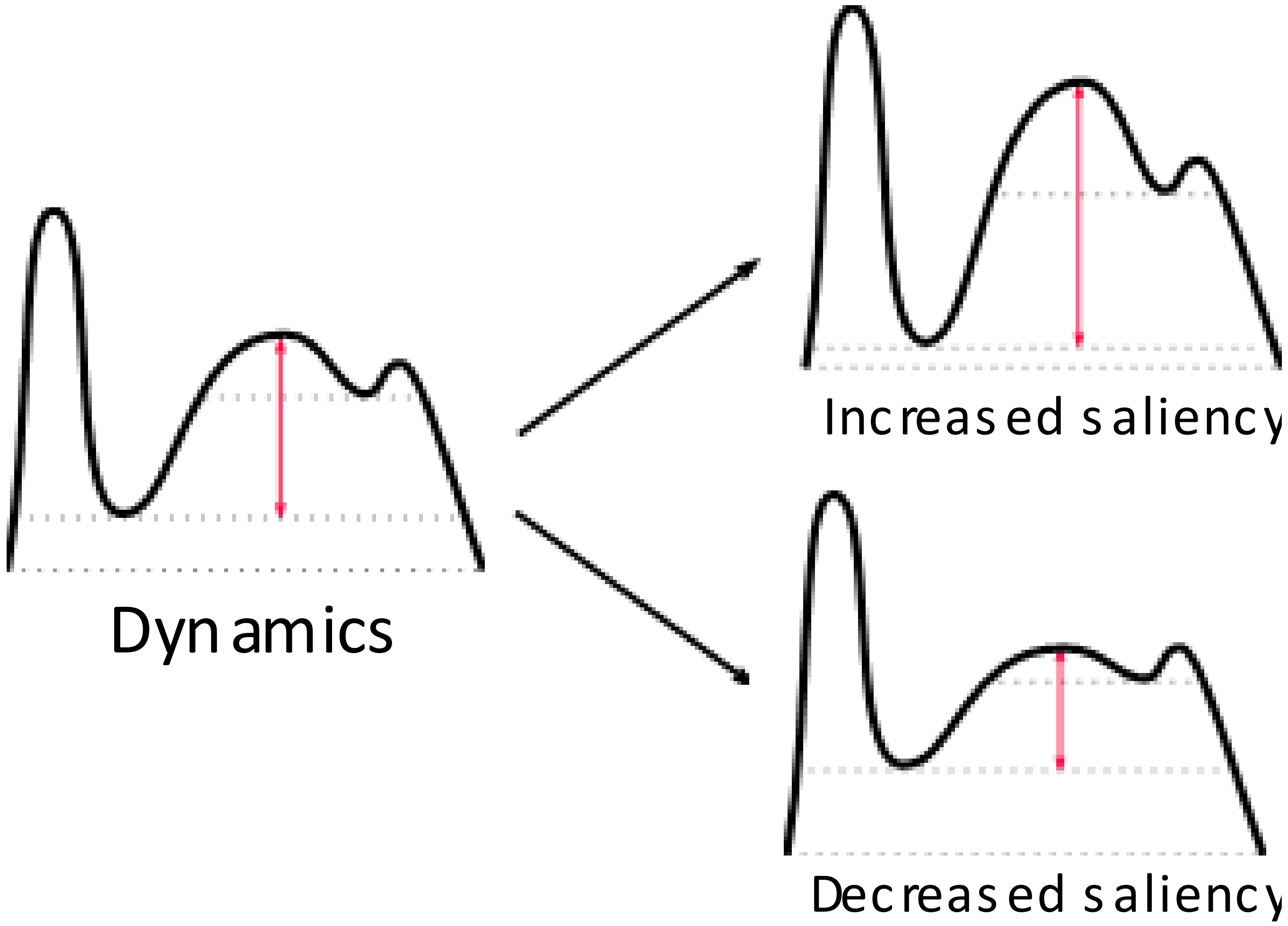
Dynamics



Volume

Max tree – Maxima saliency

- **Saliency Measure : Dynamics**



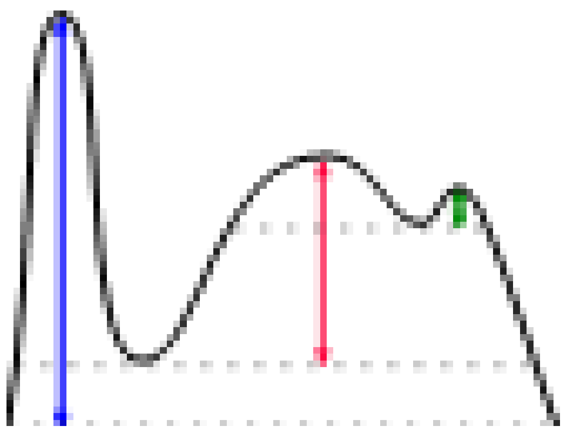
Component tree loss function – Topological Persistence

- **Topological loss function based on barcode diagrams**
- **Dynamics ⇔ Topological persistence**

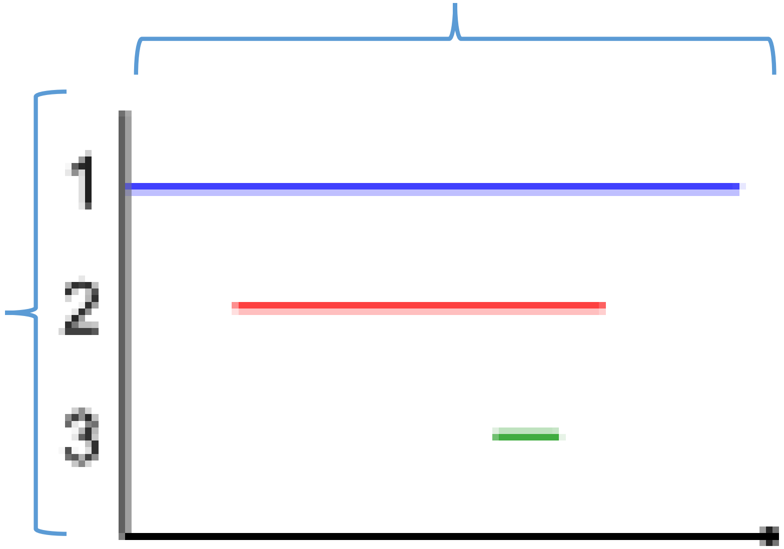
$$J_r(\mathbf{dyn}(\mathbf{f}), \mathbf{dyn}(\mathbf{f}), k)$$

Ranking by dynamics

Saliency by dynamics



Dynamics

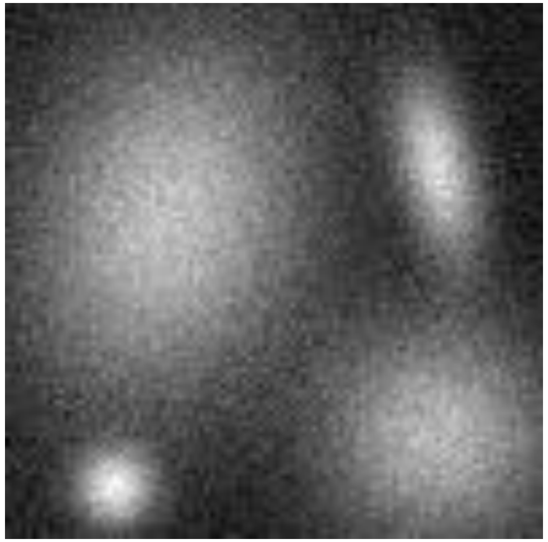


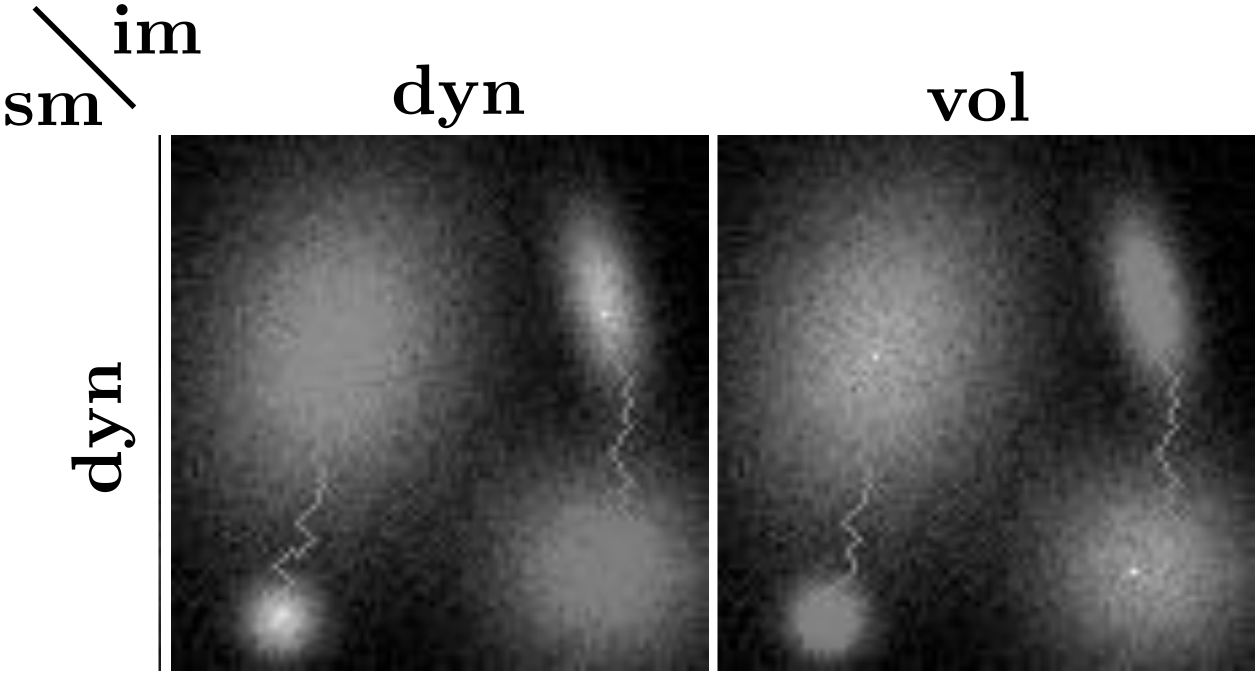
Barcode diagram of connected components

Component tree loss function – Toy Example 1

Optimization of $J_r(\mathbf{sm}, \mathbf{im}; 2)$

Input Image





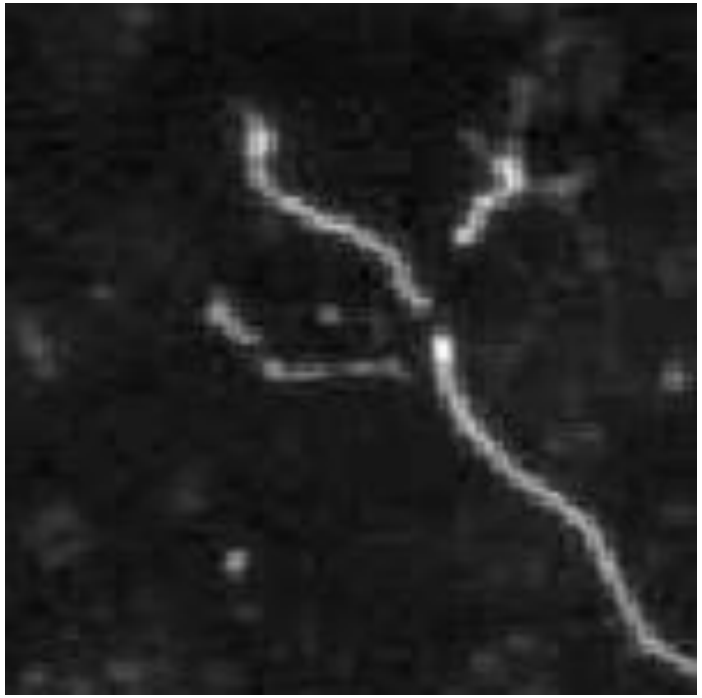
Component tree loss function – Toy Example 2

- **Combining different loss terms for image filtering**

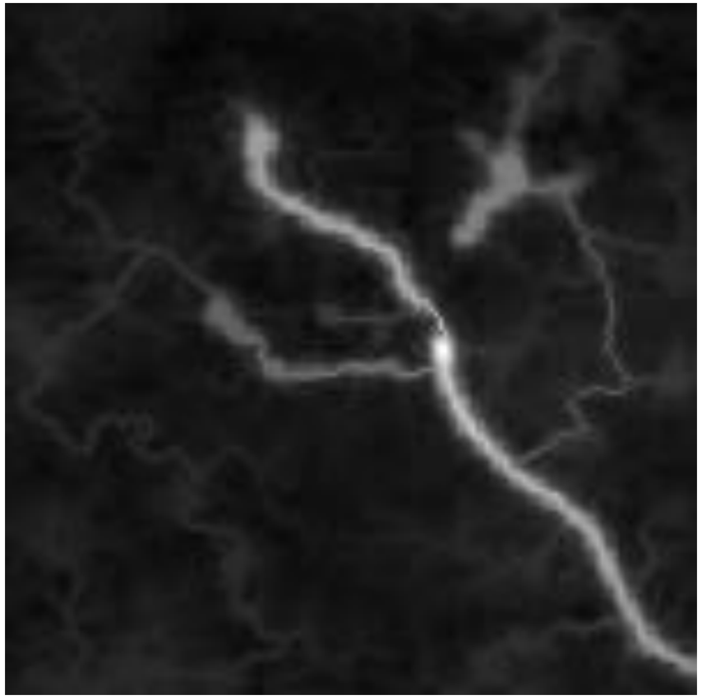
  Data fidelity + maxima loss + smoothness

$$||\mathbf{f}-\mathbf{y}||_2^2 + \lambda_1 J_r(\mathbf{dyn}(\mathbf{f}), \mathbf{dyn}(\mathbf{f}), 1) + \lambda_2 ||\nabla \mathbf{f}||_2^2$$

Input image y



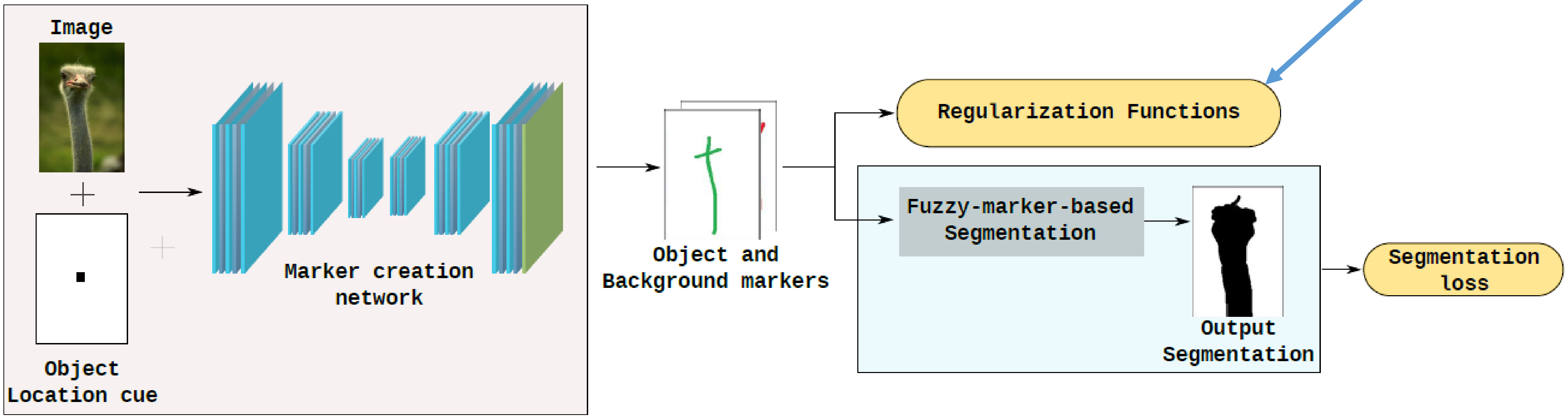
Optimized image f



Component tree loss function – Marker proposal

- **Context of interactive segmentation**
- **Propose interesting markers for the user**

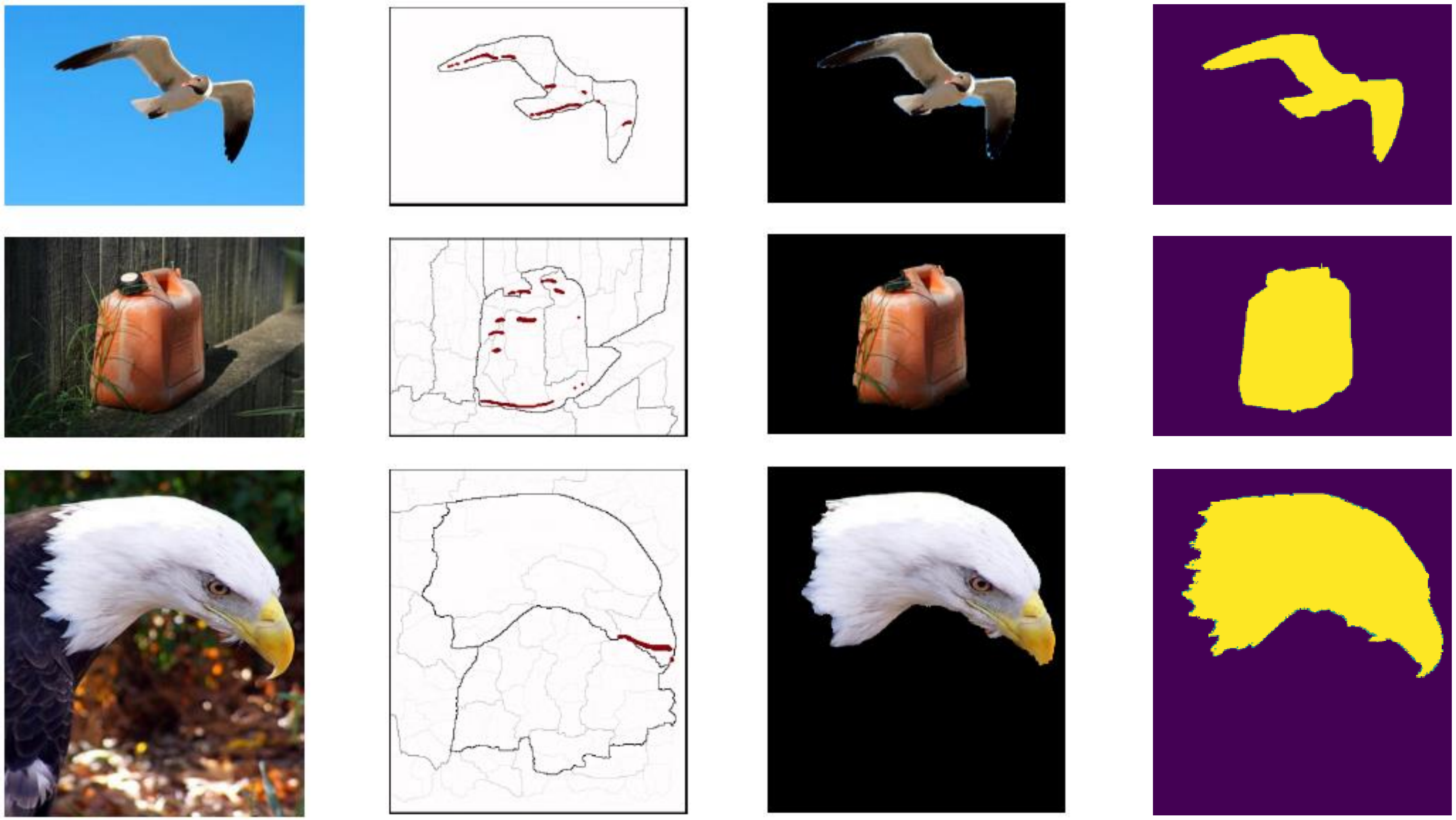
Small number of maxima



Fonseca et al. SIBGRAPI 2021

Component tree loss function – Marker proposal

Image | **Proposed markers** | **Segmentation** | **Ground-truth**



Plan

Conclusion and perspectives

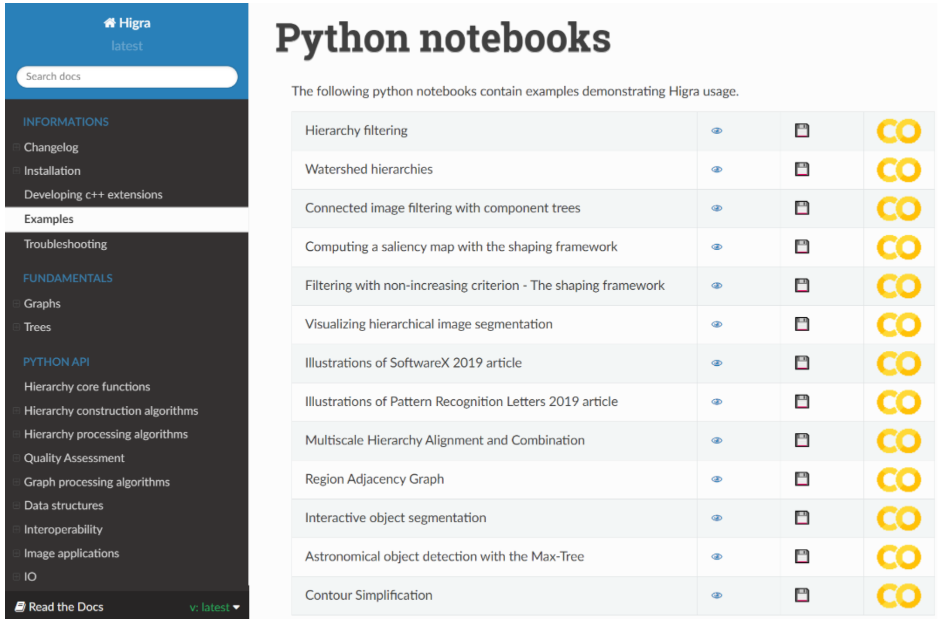
Summary

- **Optimal hierarchies**
  - Combinatorial & continuous optimization
  - Integration into machine learning methods
- **Perspectives**
  - Many new possibilites
  - More applications

Conclusion and perspectives

Higra : Hierarchical Graph Analysis

- **Open source library**
  - Python front-end
  - C++ back-end
- **A lot of hierarchical analysis methods**
  - Operate on sparse graphs
  - Specialized functions for images
- **Easy integration with machine learning libraries**
  - Seamless conversion with Numpy
  - Works with deep learning frameworks such as Pytorch




```
$> pip install higra
```

https://github.com/higra/Higra

Associated publications: Perret et al. SoftwareX 2019