Automatically finding clusters in Normalized Cuts

Mariano Tepper[a,*], Pablo Musé[b], Andrés Almansa[c], Marta Mejail[a]

[a] *Departmento de Computación, FCEN, Universidad de Buenos Aires, Argentina*
[b] *Instituto de Ingeniería Eléctrica, FI, Universidad de la República, Uruguay*
[c] *CNRS LTCI, Telecom ParisTech, France*

Abstract

Normalized Cuts is a state-of-the-art spectral method for clustering. By applying spectral techniques, the data becomes easier to cluster and then k-means is classically used. Unfortunately the number of clusters must be manually set and it is very sensitive to initialization. Moreover, k-means tends to split large clusters, to merge small clusters, and to favor convex-shaped clusters. In this work we present a new clustering method which is parameterless, independent from the original data dimensionality and from the shape of the clusters. It only takes into account inter-point distances and it has no random steps. The combination of the proposed method with normalized cuts proved successful in our experiments.

Keywords: clustering, Normalized Cuts, a contrario detection

1. Introduction

Clustering is an unsupervised learning method in which a set of observations is assigned into subsets (called clusters) so that observations within the same cluster are similar in some sense. Clustering is an interesting problem for many domains, such as image and signal analysis, bioinformatics or medical sciences. It has been applied for image segmentation [1, 2, 3], object class and shape

*corresponding author
Email addresses: `mtepper@dc.uba.ar` (Mariano Tepper), `pmuse@fing.edu.uy` (Pablo Musé), `andres.almansa@telecom-paristech.fr` (Andrés Almansa), `marta@dc.uba.ar` (Marta Mejail)

recognition [4, 5], gene network analysis [6] and internet databases analysis [6, 7], among others. Nowadays, the need for exploratory data analysis has become of extreme importance due to the increase in both volume and variety of data. Many domains are in need of computational techniques that do not rely on strong a priori knowledge.

Despite its intuitive simplicity, it is extremely hard to provide a formal definition of what a cluster is. Different authors use different definitions. Very often definitions are derived from the algorithm being used, instead of the opposite.

Theoretical methods to assess or classify clustering algorithms have been developed with interesting results [8, 9]. Unfortunately, the lack of a unified definition makes it difficult to find a unifying complete framework. An excellent overview of the clustering field was published in 2009 by Jain [10]. Here we briefly review two of the main trends in clustering algorithms.

Variations of the minimal spanning tree or limited neighborhood set approaches have been extensively explored [11, 3]. The criteria in most works are based on local properties of the graph. Since perceptual grouping implies an assessment of local properties versus global properties, exclusively local methods must be discarded or patched. For example, Felzenszwalb's method [3] makes use of the minimal spanning tree but is forced to add an ad hoc global criterion to correct local observations.

Normalized Cuts [12, 13] is a state-of-the-art spectral method which provides a partition of the graph. It analyzes a function that is a trade-off between global and local graph properties by analyzing the Laplacian of the graph, see Section 2 for further details.

As early as in 1971 Zahn established in a pioneering work [14] the conceptual grounds on which this work is based. He faced the problem of finding perceptual clusters according to the "proximity" gestalt principle [15]. He proposed three key arguments:

1. **Only inter-point distances matter**. The characteristics of the metric space should not be used. In other terms, we must look for solutions

that do not rely on any assumptions about the chosen metric. It imposes graphs as the only suitable underlying structure for clustering.

2. **No random steps**. Results must remain equal for all runs of the detection process. In particular, random initializations are forbidden.
3. **Independence from the exploration strategy**. The order in which points are analyzed must not affect the outcome of the algorithm.

The ultimate goal of our work is to propose a clustering method which follows Zahn's principles. In recent years, a theory for the quantitative analysis of gestalts, called Computational Gestalt [16], was developed and since then many refinements followed. By studying Zahn's principles in the light of the Computational Gestalt theory, we introduce a perceptually driven clustering method. It only takes into account inter-point distances and it has no random steps. As a consequence, it is independent from the original data dimensionality and from the shape of the clusters. Although the proposed approach is a general clustering method, in this work we combine it with normalized cuts.

This work is structured as follows. In Section 2 we review spectral graph theory and in Section 3 we present a new parameterless clustering method. In Section 4 we show results and then provide final remarks in Section 5.

2. Spectral Graph Methods

Formally, we want to find clusters in a feature set $X = \{x_i \in \mathbb{R}^H\}_{i=1\dots N}$ where H is the dimension of the feature space. In the following the terms point and feature have the same meaning and we will use one or the other depending on the context.

Here we briefly recall spectral graph concepts used for clustering. For further details, see the extended overview of the spectral graph theory by Chung [17].

Let $G = (V, E)$ be an undirected graph with a vertex set $V = \{v_i\}_{i=1\dots N}$, where N is the cardinal of V, and an edge set $E \subseteq V \times V$. We consider graphs to be non-reflexive, i.e. $\forall\ v \in V,\ (v, v) \notin E$. We also define an edge weighting function $\omega : E \to \mathbb{R}$ such that $\forall\ e \in E,\ \omega(e) > 0$.

We denote by $W \in \mathbb{R}^{N \times N}$ the matrix defined by $W_{ij} = \omega((v_i, v_j))$, where $1 \leq i, j \leq N$. We denote by $D \in \mathbb{R}^{N \times N}$ the diagonal matrix such that $D_{ii} = \sum_j W_{ij}$. The Laplacian of G is defined to be the matrix

$$\mathcal{L} = D^{-1/2}(D - W)D^{-1/2} = I - D^{-1/2}WD^{-1/2}. \tag{1}$$

$\mathcal{L}$ is symmetric positive semidefinite [1] and its eigenvalues lie in the interval $[0, 2]$. Von Luxburg [18] analyzed the advantages of this form of the Laplacian.

Ng et al. [1] and Fowlkes et al. [12] studied the utility of employing multiple eigenvectors of $\mathcal{L}$ to embed each feature into an M-dimensional space ($M \ll N$). To build the embedding, we compute the $N \times N$ matrices A and Λ such that $\left(D^{-1/2}WD^{-1/2}\right) A = A\Lambda$ and the values $\lambda_i = \Lambda_{ii}$ are sorted in ascending order. The columns of A are the eigenvectors of $\mathcal{L}$ and $1 - \lambda_i$ are its eigenvalues. The M-dimensional embedding is the result of keeping the first M columns of A forming the matrix A_M. The normalized form of A_M is defined as

$$\overline{A}_M \overset{\text{def}}{=} D^{-1/2}A_M. \tag{2}$$

The resulting spectral clustering procedure is as follows:

1. build G_o from $V = X$ using $\omega((v_i, v_j)) = d(x_i, x_j)$ where d is an application specific kernel distance,
2. compute the embedding $\overline{A}_M$ from G_o,
3. apply a clustering algorithm to the rows of $\overline{A}_M$, as they form tight clusters.

Each feature is finally assigned to the cluster to which its corresponding row belongs. This procedure provides a relaxation of minimizing the Normalized Cut of the original graph [13], given by

$$\mathrm{NCut}(C_1, C_2) \overset{\text{def}}{=} \frac{\mathrm{f}(C_1, C_2)}{\mathrm{f}(C_1, V)} + \frac{\mathrm{f}(C_1, C_2)}{\mathrm{f}(C_2, V)} \tag{3}$$

where $C_1, C_2 \in V$, $C_1 \cap C_2 = \emptyset$ and $\mathrm{f}(C, C') = \sum_{u \in C, v \in C'} \omega((u, v))$.

Ng et al. [1] and Fowlkes et al. [12] use k-means for the final clustering stage. It is well known that k-means presents two drawbacks: (1) it is very sensitive to the initialization and (2) the number of clusters has to be manually specified.

The former is usually addressed by performing several random initializations and then choosing the optimal partition with respect to Ncut, see Figure 1. The latter is an open question as choosing k is a difficult model-selection problem. Popular approaches to circumvent this issue are Akaike's Information Criterion (AIC), Bayesian Information Criterion (BIC) [19] or Minimum Description Length (MDL) [20]. While these methods rely on firm theoretical background, results usually differ from a human made choice.

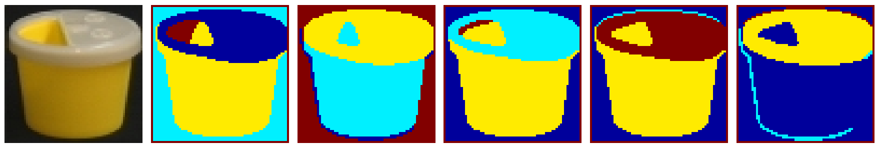
Figure 1: Example of image segmentation from 3×3 color patches, obtained using Normalized Cuts and k-means with $k = 4$. In each example k-means was executed 5 times and the segmentation that minimizes the Normalized Cut is chosen. Due to the clusters difference in density and in cardinality, the method yields unstable results.

Nadler and Galun [21] showed that spectral methods may not reveal clusters of different sizes and scales. This assertion holds when clusters are intertwined or sufficiently near each other. However in practice, when clusters (even of different sizes and scales) are well separated, spectral methods perform well.

Ozertem et al. [22] propose to use the Mean Shift algorithm to build the adjacency matrix. Mean Shift [2] detects clusters by computing what Comaniciu and Meer call attraction basins. Vertices that lie on different attraction basins are removed from the graph. The matrix is built by using the groups found with Mean Shift, instead of directly using a kernel distance, for determining locality.

3. A Contrario Clustering

In this section we present a method that overcomes the issues presented in the previous section. It is not dependent of random initializations and clusters are detected without manually specifying its number. The presented method is general, in the sense that it can be applied to any feature set X equipped with a suitable distance d.

We start by presenting a method in which ours is inspired. Then we introduce theoretical definitions in Section 3.1. The automatic choice of detection thresholds is discussed in Section 3.2. Sections 3.3 and 3.4 address the simplification and revision, respectively, of the obtained clusters.

In the scope of the Computational Gestalt programme [23], Cao et al. proposed a cluster detection method [5]. The main idea is to measure the statistical significance of a set of points as being a cluster. Let us recall its basic definition.

For $k \leq N \in \mathbb{N}$ and $p \in [0,1]$, let us denote by

$$\mathcal{B}(N,k;p) \overset{\text{def}}{=} \sum_{j \geq k}^{N} \binom{N}{j} p^j (1-p)^{N-j} \tag{4}$$

the tail of the binomial law. See Desolneux et al.for a thorough study of the binomial tail and its use in the detection of geometric structures [23].

Let $\mathcal{R}$ be a set of H-dimensional hyper-rectangles parallel to the coordinates axes and centered at the origin.

Definition 1. *Let $C \subset X$ be a subset of k points out of the N data points. We call number of false alarms (NFA) of C,*

$$\text{NFA}(C) \overset{\text{def}}{=} |\mathcal{R}| \cdot N \cdot \min_{\substack{x_i \in C \\ R \in \mathcal{R} \\ C \subset R + x_i}} \mathcal{B}(N-1, k-1, \pi_i) \tag{5}$$

where $R + x_i$ is the rectangle R translated to x_i and $\pi_i = \Pr(x \in x_i + R)$ is its probability. We say that C is an ε-meaningful group if $\text{NFA}(C) < \varepsilon$.

The term $|\mathcal{R}| \cdot N$ is the number of tests and the remaining part of Equation 5 represents a Probability of False Alarms (PFA). $\mathcal{B}(N-1, k-1, \pi_i)$ corresponds to the probability that at least $k-1$ out of the $N-1$ remaining points fall into $x_i + R$. The detection algorithm consists in exploring the group of points given by a dendrogram, identifying ε-meaningful groups as clusters and then performing an additional pruning, based on the inclusion properties of the dendrogram. A similar technique will be described in Section 3.3.

Notice that the above detection rule does not conform to Zahn's first argument. Indeed, inter-points distances are not directly taken into account and hyper-rectangles are used instead.

Another drawback of this approach is that a set $\mathcal{R}$ of hyper-rectangles parallel to the axes must be properly chosen. This choice is application specific since $\mathcal{R}$ is intrinsically related to cluster size/scale. For example, an exponential choice for the discretization of the rectangle space is made by Cao et al. [5] introducing a bias for small rectangles (since they are more densely sampled).

Each cluster must be fitted by an axis-aligned hyper-rectangle $R \in \mathcal{R}$, meaning that clusters with arbitrary shapes are not detected. Even hyper-rectangular but diagonal clusters may be missed or oversegmented.

The probability law π_i for each hyper-rectangle $R \in \mathcal{R}$, assuming no specific structure in the data, must be known a priori or estimated. The complexity of computation of the probability of an hyper-rectangle then depends on the dimension H and suffers from the "curse of dimensionality".

3.1. Graph-based A Contrario Clustering

We now propose a new method to find clusters in graphs that is independent from their shape and from the dimension H. We first build a weighted undirected graph $G = (V, E)$ where $v_i \in V$ is identified with a feature $x_i \in X$ in a metric space (X, d) and the weighting function ω is defined in terms of the corresponding distance function, namely $\omega((v_i, v_j)) = d(x_i, x_j)$.

Although the method may be applied to any metric space, in this work we will use the rows of $\overline{A}_M$ as the input features $X = \{x_i\}_{i=1\ldots N}$, by identifying x_i with the i-th row of $\overline{A}_M$ and by defining d as the usual Euclidean distance in $\mathbb{R}^M$. Notice that this distance comes from the embedding described in Section 2 and may represent a more complex semidistance in the original feature space.

A subgraph G' of G is a connected graph $G' = (V', E')$ in which $V' \subseteq V$ and $E' \subseteq V' \times V'$.

Definition 2. *We define the* non-compactness *of a subgraph G' as*

$$c(G') \stackrel{\text{def}}{=} \frac{\Omega(E')}{\Omega(E)} \quad \textit{where} \quad \Omega(E) = \sum_{e \in E} \omega(e). \tag{6}$$

We say G' is p-compact if $c(G') = p$.

Non-compactness is an estimation of the local vertex density, which plays the role of the relative volume π_i of the hyper-rectangles in Definition 1. It can be interpreted in the spirit of non-parametric density estimation. Parzen methods [24] locally estimate density at a given point by computing the distances to its neighbors

$$\mathrm{p}(x) \approx \frac{1}{Nh} \sum_{i=1}^{N} Q\left(\frac{||x - x_i||}{h}\right), \tag{7}$$

where Q is a smoothing kernel. Choosing a kernel Q that evaluates to one in G' and to zero elsewhere, and picking $x \in X$, yields

$$\mathrm{p}(x) \approx \frac{1}{Nh^2} \sum_{i=1}^{K} \omega((v, v_i)), \tag{8}$$

where $v \in V$ is the vertex associated to feature x. Finally

$$\Pr(v \in V') \approx \frac{1}{Nh^2} \sum_{j=1}^{K} \sum_{i=1}^{K} \omega((v_j, v_i)). \tag{9}$$

By normalizing $\Pr(v \in V')$, $c(G')$ is obtained.

In our approach, non-compactness plays an important role in cluster detection. Informally, a cluster is a subgraph with two properties: (1) its vertices are sufficiently near each other with respect to the remaining vertices, i.e. small non-compactness, and (2) the number of its vertices is sufficiently large.

A detection scheme must propose a balance between the density of the cluster and its size. A small set must be very dense to be perceptually noticeable while larger sets are more clearly perceived even if they are less dense. A set of points will be more striking, and more compact, as it gets farther away from the rest of the points.

The non-compactness of a subgraph models how tight its vertices are but is not sufficient to characterize clusters. This can be seen in Figure 2 where $G'_1 = (V'_1, E'_1)$ and $G'_2 = (V'_2, E'_2)$ are two different subgraphs such that $\Omega(E'_1) = \Omega(E'_2)$. Suppose they are respectively embedded in two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ such that $\Omega(E_1) = \Omega(E_2)$, then G'_1 would have the same non-compactness as G'_2 while having more nodes.

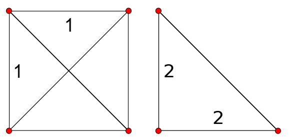

Figure 2: Two different subgraphs $G'_1 = (V'_1, E'_1)$ and $G'_2 = (V'_2, E'_2)$ such that $\Omega(E'_1) = \Omega(E'_2) = 4 + 2\sqrt{2}$. Suppose they are respectively embedded in two graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ such that $\Omega(E_1) = \Omega(E_2)$, G'_1 would have the same non-compactness as G'_2 while having more nodes.

Suppose we wanted to statistically model all possible instances of clustered data. First, we would need a model for each possible cluster shape (e.g. Gaussian). Each individual model would have its own parameter set. Next, since we seek for a general model, it must support data distributed in multiple clusters with different shapes. In other terms, we need to integrate the individual models into a mixture. For example, k-means is aimed to detect mixtures of k Gaussian distributions.

These different mixtures can be seen as a parametric family $\mathcal{F}$ of distributions. This family is parametric on the number of clusters and on the parameters of the distribution modeling each cluster. Such a family has colossal cardinality and parameter set. Even if we restrict ourselves to a mixture of k Gaussian clusters for H dimensional data, the problem is hard enough. Since there are $H(H+1)/2 + H$ parameters for each Gaussian (determined by an $H \times H$ covariance matrix and $H \times 1$ mean vector), the number of parameters is $k(H(H+1)/2 + H)$ for each k. A family of such mixtures, parametric on k, is already untestable in practice.

Assume we were to model the problem described above as a classical hypothesis test:

$\mathcal{H}_1$*: the observed features have a particular distribution* $F \in \mathcal{F}$*, i.e. the data is clustered.*

$\mathcal{H}_0$*: the observed features are distributed more randomly, e.g. uniformly.*

Modeling $\mathcal{H}_1$ means to model $\mathcal{F}$. Modeling $\mathcal{F}$, even if it was feasible, would require an a priori characterization of what a cluster is. Due to the reasons that have just been exposed, we are not interested in defining such a model. We prefer instead to follow the classical claims of the Computational Gestalt School [23]; our detection algorithm will be driven by the Helmholtz principle: no perception in white noise. We will only concentrate on modeling $\mathcal{H}_0$ and consider that low probabilities of occurrence under $\mathcal{H}_0$ are indeed causal instead of casual.

Testing randomness to detect clusters is not a new concept. To cite a few, Hoffman, Jain et al. [25, 26] perform such kind of tests. The works by Cao et al. [5] and by Desolneux et al. [23] also follow this line of research.

We are interested in a general clustering method but, in accordance to our model, in applications where the cluster shapes and the number of clusters are known a priori, the full hypothesis test could be performed, using similar but more simple a contrario techniques.

Given a subgraph G' and its non-compactness, we consider its number of vertices k as a realization of a random variable K. We can then model the probability that a p-compact subgraph G' has at least k vertices due to a realization of randomness. We denote this probability by $\Pr(G' \mid \mathcal{H}_0)$.

Definition 3. *Let G' be a p-compact subgraph of G, we define the probability of false alarms (PFA) of G' as*

$$\mathrm{PFA}(G') \overset{\mathrm{def}}{=} \Pr(G' \mid \mathcal{H}_0) = \mathcal{B}(|V|, |V'|; p), \tag{10}$$

where $|\cdot|$ denotes the cardinality of a set.

The probability of false alarms quantifies the unlikeliness of occurrence of a p-compact subgraph G' of G with at least $|V'|$ nodes among $|V|$ under $\mathcal{H}_0$. In other terms the unlikeliness of occurrence of a configuration of at least $|V'|$ features among $|V|$ with density p.

There have been previous attempts to automatically detect the number of clusters in Normalized Cuts (or spectral clustering), e.g. by Zelnik-Manor and Perona [27]. Their work has similar goals but starts from different requirements: they indeed make use of the characteristics of the eigenspace. Moreover, since their method is based on selecting the minimum of an alignment cost function, it is not able to detect non-clustered data as such. Our method is specifically designed for this task.

3.2. Learning detection thresholds

To detect unlikely dense subgraphs, a threshold is necessary on the PFA. In the classical *a contrario* framework, a new quantity is introduced: the Number of False Alarms (NFA), i.e. the product of the PFA by the number of tests (see Definition 1). The NFA has a more intuitive meaning than the PFA, since it is an upper bound on the expectation of the number of false detections [23]. The threshold is then easily set on the NFA.

To use the NFA one has to be able to compute (or at least analytically estimate) the number of tests being performed. In our setting, this is not possible, since it is equivalent to computing the number of subgraphs for a graph, which is an astronomical quantity (e.g. approximately 10^{300} for $N = 1000$).

An alternative solution, proposed by Burrus [28], consists in estimating the threshold directly on the PFA by Monte Carlo simulations, following the actual search heuristics instead of trying to bound the total number of tests in a full search. Furthermore, this solution allows to estimate not only a global threshold but partial thresholds, by splitting the detected subgraphs into different categories, each with a dedicated threshold. In this work we follow this approach.

Let $\mathcal{G}$ be the set of all subgraphs of a graph G and let $\mathcal{J}_K : \mathcal{G} \to \{1, 2, \ldots, K\}$ be a hash function used to divide $\mathcal{G}$ into K categories. We define the exploration strategy $\mathcal{S} \subseteq \mathcal{G}$ as a set of subgraphs to be analyzed during detection and learning. In Section 3.3 we analyze exploration strategies in depth. As an example we define the basic universal strategy $\mathcal{S}_U = \mathcal{G}$.

We already stated that our detection algorithm is ruled by the principle of no perception in white noise. It is therefore clear that subgraphs in $\mathcal{S}$ with a PFA that is likely to occur in white noise have to be discarded. A direct approach would be to generate several random graphs, compute the PFA of their subgraphs and select a threshold such that all of them are discarded (up to a certain confidence level).

Given a hash function $\mathcal{J}_K$ and an exploration strategy $\mathcal{S}$, Algorithm 1 performs exactly that same procedure to compute a set of thresholds $\delta(\varepsilon) = \{\delta_k(\varepsilon)\}_{k=1\ldots K}$. Thresholds are first initialized. For each category k, we run Q simulations. For each simulation $q = \{1 \ldots Q\}$, the number d_q of subgraphs G' such that $\mathcal{J}_K(G') = k$ and $\mathrm{PFA}(G') < \delta_k(\varepsilon)$ is counted. Then the empirical mean m_k and deviation s_k of $d = \{d_q\}_{q=1\ldots Q}$ are computed.

An upper bound u of the expectation μ_k of m_k is computed by performing a Student's t-test. We are looking to approximate ε/K by u and $\delta_k(\varepsilon)$ is adjusted accordingly. We refer to Burrus [28] for further details.

Algorithm 1 Compute $\delta(\varepsilon)$ for N vertices using exploration strategy $\mathcal{S}$ by Q Monte Carlo simulations.

```
1:  for all $k \in \{1 \ldots K\}$ do
2:      initialize $\delta_k(\varepsilon)$
3:      repeat
4:          for all $q \in \{1 \ldots Q\}$ do
5:              build a graph $G_q$ by sampling $N$ vertices from the distribution of $\mathcal{H}_0$.
6:              apply $\mathcal{S}$ to $G_q$.
7:              $d_q \leftarrow \#\{G' \in \mathcal{S},\ \mathrm{PFA}(G') < \delta_k(\varepsilon)\ \wedge\ \mathcal{J}_K(G') = k\}$
8:          end for
9:          Compute the empirical mean $m_k$ and deviation $s_k$ of $d = \{d_q\}_{q=1\ldots Q}$
10:         Compute a confidence interval on the expectation $\mu_k$ of $d$ using the
            property $\mathrm{P}(\mu_k \leq m_k) = \mathrm{F}_{Q-1}\left(\frac{m_k - \mu_k}{s_k}\sqrt{Q-1}\right)$ where $\mathrm{F}_n(x)$ is the dis-
            tribution function of a Student law with $n$ degrees of freedom.
11:         For a chosen confidence level, if the estimated upper bound of $\mu_k$ is
            greater than $\varepsilon/K$, increase $\delta_k(\varepsilon)$ otherwise decrease $\delta_k(\varepsilon)$.
12:     until convergence of $\delta_k(\varepsilon)$
13: end for
```

Definition 4. *We say that a subgraph G' is an ε-meaningful cluster if*

$$\mathrm{PFA}(G') < \delta_{\mathcal{J}_K(G')}(\varepsilon) \tag{11}$$

for a properly computed set of thresholds $\delta(\varepsilon)$. We define the number of false alarms (NFA) of G' as

$$\mathrm{NFA}(G') \overset{\mathrm{def}}{=} \frac{\varepsilon}{\delta_{\mathcal{J}_K(G')}(\varepsilon)} \mathrm{PFA}(G') \tag{12}$$

Notice that subgraphs consisting of a single node must certainly not be detected. From Definition 3 they cannot be detected, i.e. $\mathcal{B}(|V|, 1; 0) = 1$. As we look for rare events, subgraphs with probability of occurrence in noise equal to 1 are never detected.

Lemma 1. *The expected number of ε-meaningful clusters in a random graph G is lower than ε.*

Proof. By construction of $\delta_k(\varepsilon)$, the number of meaningful subgraphs G' in a random graph G is lower than ε/K. By linearity of expectation, if there are less than ε/K errors in average for each category, then there are globally less than ε errors in average. $\square$

Figure 3 depicts the profile of the learned set of thresholds $\delta(\varepsilon)$ for the exploration strategy explained in Section 3.3 and different graph sizes. There are N vertices, $K = 10$ categories and the hash function for a graph $G' = (V'E')$ is simply $\mathcal{J}_K(G') = \lfloor (|V'| + 1) \cdot K/N \rfloor$. Medium-small and compact subgraphs are more frequent than large and compact subgraphs causing that thresholds for the first categories are more restrictive than thresholds for the last ones. Note that the evolution of the set of thresholds $\delta(\varepsilon)$ with size is smooth, allowing the computation of intermediate sets of thresholds by interpolation.

The role of the hash function is to correct a bias that might be introduced by the exploration strategy. If the sizes of the clusters one seeks to detect are well enough sampled by the exploration strategy, choosing $K = 1$ should suffice.

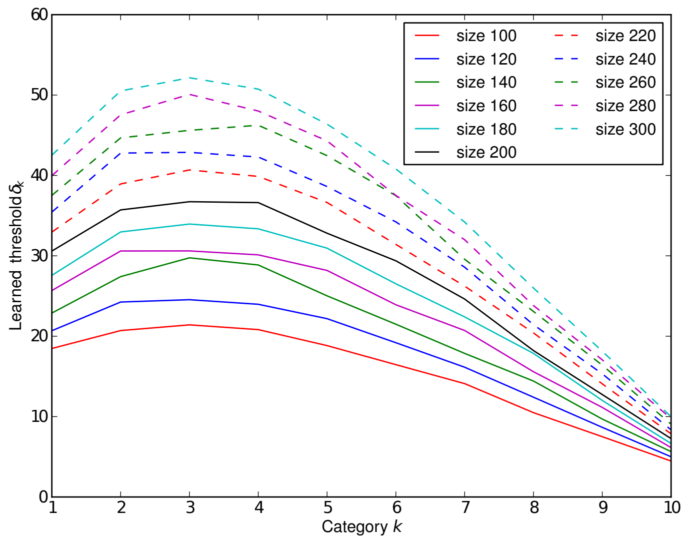

Figure 3: Learned set of thresholds $\delta(\varepsilon)$ for $\varepsilon = 1$ for sizes ranging from $N = 100$ to $N = 300$ (in negative logarithmic scale). Thresholds evolve smoothly with size, consequently they can be safely interpolated for intermediates sizes. Larger subgraphs are rarer and thresholds are therefore less restrictive than for smaller subgraphs.

3.3. Eliminating redundancy

While each meaningful cluster is relevant by itself, the whole set of meaningful clusters exhibits, in general, high redundancy [5]. Indeed, a very meaningful cluster G_1 usually remains meaningful when it is slightly enlarged or shrunk into a graph G_2. If, e.g. $G_2 \subset G_1$, this question is easily answered by comparing NFA(G_1) and NFA(G_2), see Definition 4. The NFA is used instead of the PFA to allow comparisons that span different categories. The group with the smallest NFA must of course be preferred. The above criterion is implemented by the following pruning rule:

```
for all $\varepsilon$-meaningful clusters $G_1$, $G_2$ do
  if $G_2 \subset G_1 \ \vee\ G_1 \subset G_2$ then
    eliminate $\arg\max\left(\mathrm{NFA}(G_1), \mathrm{NFA}(G_2)\right)$
  end if
end for
```

that indeed produces the desired result but is computationally intractable. Moreover, for a given graph, exploring the whole set of its subgraphs to compute each PFA is already intractable. An exploration algorithm is therefore needed. Hierarchical clustering methods are well suited for this task.

Definition 5. *A hierarchy $\mathcal{T}$ of a graph $G = (V, E)$ is a set such that $\forall\ T \in \mathcal{T}$*

$$\exists\ v \in V,\ T = \{v\}, \qquad \textit{or} \qquad \exists\ T_1, T_2 \in \mathcal{T},\ T = T_1 \cup T_2. \tag{13}$$

Depending on the direction they build the hierarchy, these clustering methods can be agglomerative (bottom-up) or divisive (top-down). The former are usually computationally simpler.

Any hierarchical algorithm [10] can be used. In this work we focus on the agglomerative algorithm called minimal spanning tree. Zahn's work [14] concentrates on stating the good properties of minimal spanning trees to detect perceptual clusters. The construction of the minimal spanning tree starts by considering each single point as a cluster and iteratively merges the pair of clusters containing the closest nearest-neighbor points. It can be found using Kruskal's method (Figure 4).

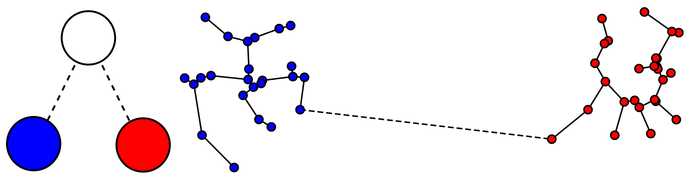

Figure 4: Part of a minimal spanning tree. The blue node set and the red node set are linked by the dashed edge, creating a new node in the minimal spanning tree.

We will restrict ourselves to explore the node sets contained in a hierarchy and to compute PFAs on the subgraphs induced by them. Finally we apply the pruning rule profiting from the inclusion properties of the tree structure.

Definition 6 (Exploration strategy)**.** *Given a graph $G = (V, E)$, we denote by $G_C = (C, E')$ the subgraph such that*

$$\forall\ (c_a, c_b) \in E,\ c_a \in C\ \wedge\ c_b \in C \Rightarrow (c_a, c_b) \in E', \tag{14}$$

and we say that G_C is induced by C. For a given hierarchy $\mathcal{T}$ we define the exploration strategy as $\mathcal{S}_{\mathcal{T}} = \{G_T\}_{T\in\mathcal{T}}$.

Definition 7 (Maximal ε-meaningful cluster)**.** *For a learned set of thresholds $\delta(\varepsilon)$, we say that $G_T \in \mathcal{S}_{\mathcal{T}}$ is a maximal ε-meaningful cluster if*

1. *G_T is an ε-meaningful cluster, see Definition 4,*
2. *all meaningful descendants are less meaningful,*
   $\forall\, T' \in \mathcal{T},\ T' \subset T,\ \mathrm{NFA}(G_{T'}) > \mathrm{NFA}(G_T)$*,*
3. *all meaningful ancestors are less meaningful,*
   $\forall\, T' \in \mathcal{T},\ T \subset T',\ \mathrm{NFA}(G_{T'}) \geq \mathrm{NFA}(G_T)$*.*

The proposed clustering algorithm simply consists on detecting maximal ε-meaningful clusters. Definition 7 is closely related to the maximality rule by Cao et al. [5] but is simpler and it might be regarded as an implementation of the exclusion principle [23]. In our experiments, we found no need to include a measure of meaningfulness for a pair of subgraphs.

According to Definition 6 the subgraph G_C is the largest subgraph in G not containing more vertices than C. Why not use instead the partial trees provided by the hierarchy as in Figure 4? Because in the father (represented in white) inter-cluster edges (dashed line) are underrepresented with respect to intra-cluster edges (solid lines).

To explain this situation, let $G = (V, E)$ be a hypothetical base graph. Let A be the blue node set and B the red node set respectively and let them induce subgraphs $G_A = (A, E_A)$ and $G_B = (B, E_B)$. Let us denote the father of both G_A and G_B by $G_{A\cup B} = (A \cup B, E_A \cup E_B \cup E_{AB})$ where $E_{AB} = \{(v_A, v_B) \in E,\ v_A \in A\ \wedge\ v_B \in B\}$. Then,

$$c(G_{A\cup B}) = \frac{\Omega(E_A) + \Omega(E_B) + \Omega(E_{AB})}{\Omega(E)}. \tag{15}$$

Let us compare $\mathrm{PFA}(G_{A\cup B})$ with $\mathrm{PFA}(G_A)$. The non-compactness $c(G_{A\cup B})$ grows by $\frac{\Omega(E_B)+\Omega(E_{AB})}{\Omega(E)}$ with respect to $c(G_A)$. Since B is tight, $\Omega(E_B)$ is small and the growth is mainly determined by $\Omega(E_{AB})$. Meanwhile, $E_A \cup E_B \cup E_{AB}$

grows by $|E_B| + |E_{AB}|$ with respect to E_A. If $|E_{AB}|$ is small (in our case 1), the growth in size is mainly determined by $|E_B|$. The same reasoning can be applied to $\mathrm{PFA}(G_{A\cup B})$ and $\mathrm{PFA}(G_B)$. In summary, $\Omega(E_{AB})$ has to be very large to compensate for the relatively large growth in size. If it is not the case, the father will be more meaningful than its two children. Only very separated clusters will be detected separately. To correct this situation, inter-cluster edges have to be better sampled. A reasonable choice is using the subgraphs induced by A and B.

Comaniciu and Meer [2] state that "arbitrarily structured feature spaces can be analyzed only by nonparametric methods since these methods do not have embedded assumptions". They classify nonparametric clustering methods into two classes: *hierarchical clustering* and *density estimation.* Regarding the first class, they regard the detection of clusters in a hierarchy as being a non-trivial task. The proposed approach, maximal ε-meaningful clusters, merges these two main trends. It detects clusters in a hierarchy by using density estimation. The hierarchy provides candidate groups in a natural manner thus, from one side, allowing a reduced effort in the density estimation step and, from the other side, providing the cardinality of such groups as important information.

3.4. Revising elongated clusters

Non-elongated clusters are preferred by our detection algorithm. Elongated clusters are separated in several non-elongated maximal meaningful clusters as in Figure 5(a). Detecting non-elongated clusters is equivalent to detecting clusters with the same intrinsic dimension as the embedding. Since in the embedding (see Section 2) some clusters are indeed elongated, a revision is needed.

For a given graph $G = (V, E)$, let $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ be two subgraphs of G. We define

$$\mathrm{L}(G_1, G_2) \overset{\text{def}}{=} \min\left(\max_{e\in E_1} \omega(e),\ \max_{e\in E_2} \omega(e)\right). \tag{16}$$

This equation is very similar to the one proposed by Felzenszwalb [3] to detect clusters in a hierarchical structure except there is no extra scale parameter.

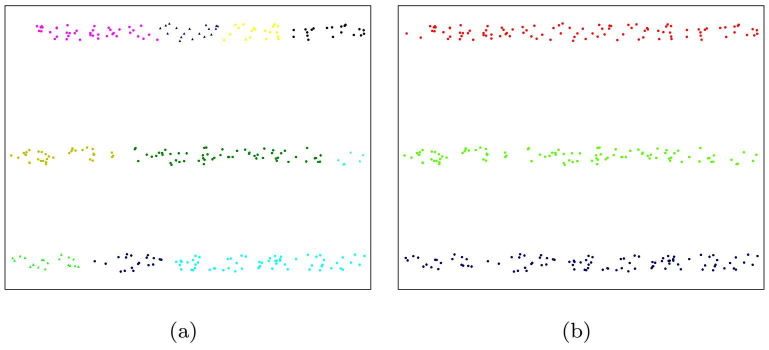

Figure 5: Maximal meaningful clusters (a) before and (b) after revising elongated clusters. In (a) 10 clusters are found while in (b) only 3 remain.

Let $G_T = (V_T, E_T)$ be a maximal ε-meaningful cluster, $G_F = (V_F, E_F)$ its father and $G_S = (V_S, E_S)$ its sibling in $\mathcal{T}$. Let us define

$$G_{T\cup S} \overset{\text{def}}{=} (V_F,\ E_T \cup E_S \cup E_{TS}) \tag{17}$$

where $E_{TS} = \{e \in E_F,\ \omega(e) \leq \mathrm{L}(G_T, G_S)\}$. Long edges connecting the extremes of the G_F are eliminated from $G_{T\cup S}$ by using a local connection between G_T and G_S. The effect of this local connection rule is shown in Figure 6.

If $\mathrm{PFA}(G_{T\cup S}) < \mathrm{PFA}(G_T)$, G_F is replaced in $\mathcal{S}_\mathcal{T}$ by $G_{T\cup S}$ and maximality in $\mathcal{S}_\mathcal{T}$ is recomputed. This procedure is repeated until convergence. Note that convergence is guaranteed since changes are propagated up in the tree, stopping at the root in the worst case. This heuristic is able to correct for oversplitting of elongated clusters present in the embedding, as seen in Figure 5(b).

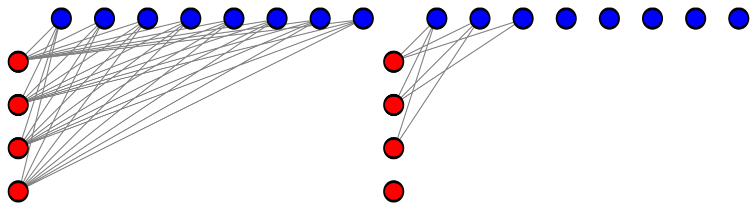

Figure 6: Effect of locally connecting clusters. The subgraphs G_T (in blue) and G_S (in red) are siblings in $\mathcal{T}$. For clarity, only inter-cluster edges are depicted. Left, their father G_F in $\mathcal{T}$. Right, the locally connected graph $G_{T\cup S}$.

Algorithm 2 summarizes the complete proposed detection approach.

Algorithm 2 For a point set X and an appropriate kernel distance d compute the set $\mathcal{M}$ of maximal ε-meaningful clusters.

```
1:  build $G_o$ from $X$ using $d$
2:  compute the embedding $\overline{A}_M$ from $G_o$ {see Equation 2}
3:  build $G_e$ from $X_e$ using the rows of $\overline{A}_M$ and Euclidean distance in $\mathbb{R}^M$
4:  compute the hierarchy $\mathcal{T}$ from $G_e$
5:  $\mathcal{M} = \emptyset$
6:  for all $G_T \in \mathcal{S}_\mathcal{T}$ do
7:    if $G_T$ is maximal $\varepsilon$-meaningful then
8:      add $G_T$ to $\mathcal{M}$
9:    end if
10: end for
11: repeat
12:   choose $G_T \in \mathcal{M}$
13:   find its sibling $G_S$, its father $G_F$ in $\mathcal{S}_\mathcal{T}$ and compute $G_{T \cup S}$
14:   if PFA$(G_{T \cup S}) <$ PFA$(G_T)$ then
15:     replace $G_T$ (and possibly $G_S$) by $G_{T \cup S}$ in $\mathcal{M}$
16:     replace $G_F$ by $G_{T \cup S}$ in $\mathcal{S}_\mathcal{T}$
17:   end if
18: until no more replacements are performed
```

4. Experimental Results

As every spectral clustering technique, there are two main values to be tuned: the scale of the kernel distance and parameter M (see Equation 2 and Algorithm 2). The former was fixed by manually choosing the scale that yields the best results with k-means. Anyhow, once the scale fixed, all compared methods analyze the transformed feature space (i.e. the embedding) an should provide results independently of that choice. The parameter M can be interpreted as an estimation of the number of groups [1]. For example, when using k-means, M is set usually set equal to k. In the case of meaningful clusters, the value of M was determined empirically and can be seen as an overestimation of the largest possible number of groups.

The proposed approach is successful at finding perceptually clear 2D clusters even when clusters have arbitrary shapes (Figure 7). By only fixing the parameters needed to compute the embedding, i.e. its dimension M and the variance of the Gaussian kernel used for the distance, the clustering is successful. No extra parameter is needed as fixing $\varepsilon = 1$ is sufficient for stable detections.

Figure 7(e) presents an interesting case since some maximal clusters are not meaningful (represented in cyan, black and orange). The scene is composed of a mixture of three Gaussians. Peripheral points, i.e. located in low density areas, are harder to merge. Note that spectral methods are unable to deal with highly intertwined clusters as features are mapped to a single manifold in the Euclidean embedding.

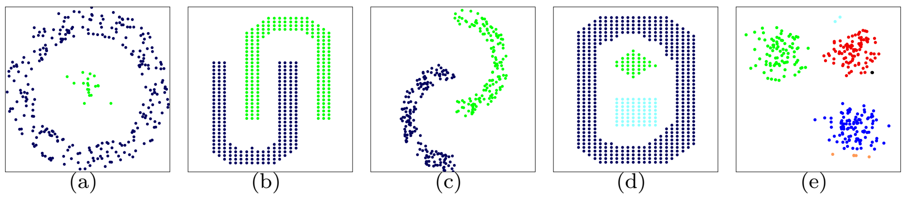

Figure 7: 2D points clustering examples. In (a), (b), (c) and (d) all groups are meaningful. In (e) maximal clusters are shown: only red, green and blue groups are meaningful while cyan, black and orange are not and are finally discarded by our algorithm.

Figure 8 depicts a comparison between results using k-means and our algorithm. Both start from the same embedding. We consider that there are 15 clusters in Figure 8(a). For k-means the correct number of clusters was set. The combination of groups with very high density and groups with low density causes the random initialization in k-means to fail, see Figure 8(b). It creates under-split clusters, e.g. rectangle A, and over-split clusters, e.g. rectangle B. Maximal ε-meaningful clusters perform correctly with no parameter tuning (i.e.

the number of clusters is automatically found by the algorithm), see Figure 8(c).

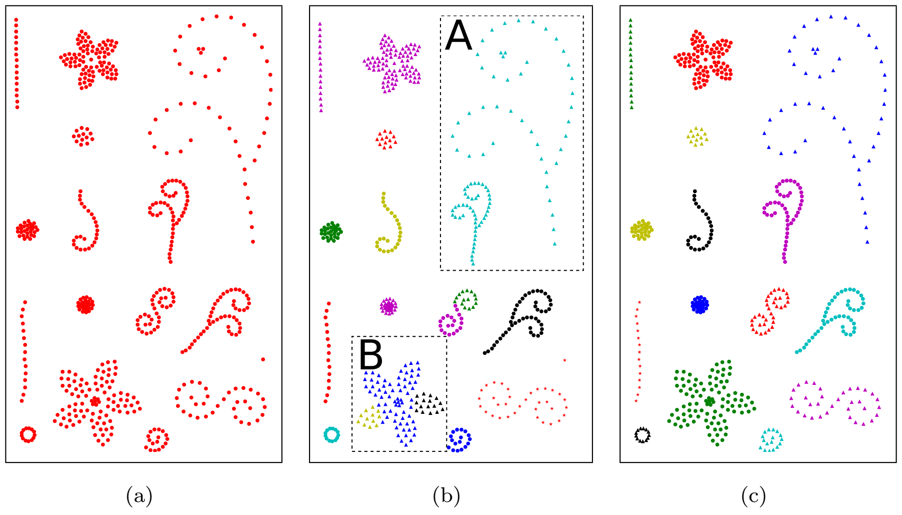

Figure 8: (a) Original scene. Starting from the same embedding, result with (b) k-means, where we manually set the correct number of clusters, and (c) maximal meaningful clusters. k-means incorrectly merges some clusters (zone A) and incorrectly splits others (zone B).

The next experiment aims at comparing our results with Mean Shift [2, 29]. Mean Shift performs a non-parametric density estimation (using sliding windows) and finds its local maxima. Clusters are determined by what Comaniciu and Meer call "basins of attraction" [2]: points are assigned to a local maximum following an ascendent path along the density gradient[1].

Figure 9 presents an experiment were Mean Shift is used to cluster the dataset in Figure 8(a). Different density estimations were performed, by varying the kernel size. Clearly, results are suboptimal. The main disadvantage we see in the density estimation step is that a global kernel size must be chosen. Such a strategy is unable to cope with clusters of different densities and spatial sizes. Choosing a small kernel causes to correctly detect dense clusters at the price of oversplitting less denser ones. On the contrary, a large kernel corrects the oversplitting of less denser clusters but introduces undersplitting for the denser

[1] `www.mathworks.com/matlabcentral/fileexchange/10161-mean-shift-clustering`

ones. Our method also uses non-parametric density estimation, but the scale is not fixed in advance. As shown by the Parzen windows interpretation of the non-compactness, the "kernels" we use are determined by the candidate sets given by the hierarchy and thus multiscale density estimation if performed.

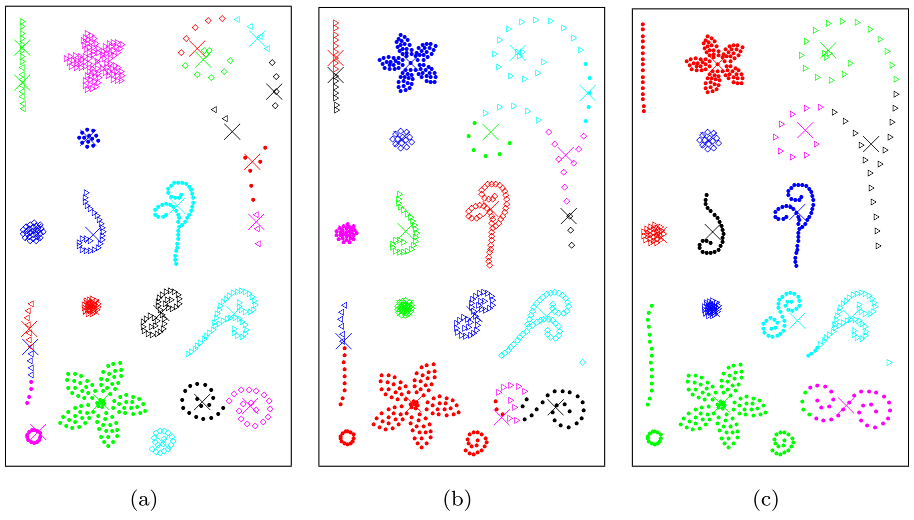

Figure 9: Results with Mean Shift for the point set from Figure 8(a). The local maxima of the estimated density are signaled by x's. Even when varying the kernel size, results are clearly suboptimal.

In Figure 10 we show segmentation results on synthetic images. A precision should be made regarding this experiment as well as all segmentation experiments in this paper: our goal here is not to present a new segmentation method, but just to illustrate the performance of proposed clustering technique by means of segmentation examples. For this reason, we simply consider that the vectors to be clustered are the set of single color image pixels or 3×3 color image patches, depending on the experiment. Notice that there is no term imposing image spatial connectivity of clusters.

If the random initialization procedure picks an appropriate seed, Normalized Cuts with k-means may perform reasonably well when setting the correct k (Figure 10). Still, sometimes the resulting clusters can be degraded by noise, as in the top figure on the second column. When k is not well chosen ($k = 3$ in our example) results are poor, as on the third column. Results on the fourth column show that our method is successful without any further parameter tuning.

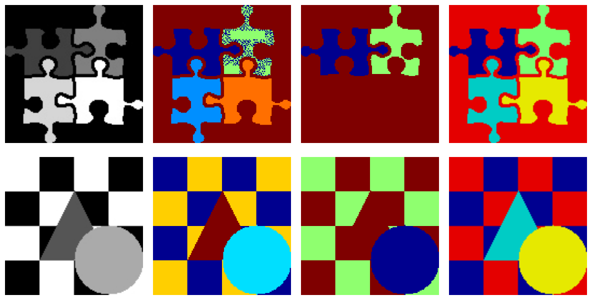

Figure 10: Comparison of image segmentations. On the first column,original images with 5 and 4 regions respectively. On the second column, segmentations obtained with k-means by correctly setting k to 5 and 4 respectively. On the third column, segmentations obtained with k-means by setting $k = 3$. On the fourth column, segmentations obtained with our method.

Figure 11 presents more image segmentation results. Results on the second column are among the best we could obtain with k-means, by carefully choosing k by hand (we set k to 3, 6, 6, 6 and 11 respectively). In general, results are correct. In all cases except for the one on the first row, the number of clusters k had to be overestimated with respect to the number of visually perceived regions. In some cases, some regions are overconnected (see the blue region on the first row and the red region on second row) a fact that could not be corrected by slightly increasing k.

Results on the third column were obtained with Zelnik-Manor and Perona' method (ZMP) [27][2]. In this case the features are individual color pixels, since we did not find better results by using patches. In one case, on the fifth row, the method oversegments the image, while in the others the image is undersegmented. In these cases, boundaries between regions seem somewhat away from perceived regions.

The proposed algorithm, whose results are depicted on the fourth column, performs well in all cases, being able to correctly separate perceptually evident

[2]`http://webee.technion.ac.il/~lihi/Demos/SelfTuningClustering.html`

clusters. Contrarily to the other two methods, small clusters are not arbitrarily merged to the closest larger cluster but remain undetected. In simpler terms, some patches are detected as not belonging to any cluster: they are considered as noise. In accordance to this claim and since we are clustering patches (without any kind of rotation invariance) the ones that lie on the boundaries between objects are not classified. This is a desirable feature since boundary patches are of different nature from non-boundary patches.

Figure 12 presents more segmentation results on images from the COIL-100 database. The same remarks from the previous examples hold. In general, our method correctly finds the clusters and outperforms ZMP, although in some cases relatively big areas remain detected as unclustered patches.

The Berkeley Segmentation Dataset [30] is often used to perform segmentation experiments. Nowadays, to our knowledge, an exhaustive review of the clustering methods reported in the literature shows that there exists no clustering approach that is able to correctly and automatically segment all images in such a varied and complex dataset. Hence, we selected a subset of this database that we consider that should be easier to segment. For the sake of completeness, experiments on this subset are also included.

For the final set of experiments we compare our method with the algorithm by Cour et al. [31]. They use the Normalized Cut framework, but using a multiscale decomposition[3]. The final embedding for clustering is constructed by using the information on the different scales and applying inter-scale constraints to ensure overall consistency. The final clustering step is performed by using the discretization algorithm by Yu and Shi [32], which seeks the discrete solution closest to the continuous optimum by rotating the normalized eigenvectors. In their algorithm the number of clusters is an input parameter and must be equal to the dimensionality of the embedding; this choice does not necessarily lead to optimal results. Moreover, as pointed out by Zelnik-Manor and Perona, this iterative method can easily get stuck in local minima and thus does not reliably

[3]`www.seas.upenn.edu/~timothee/software/ncut_multiscale/ncut_multiscale.html`

find the optimal alignment [27]. This claim was confirmed in our experiments.

In Figures 13, 14 and 15 we use Cour's algorithm to construct the embeddings and then compare the clustering results obtained with different methods: (a) Yu and Shi' algorithm (YS) [32]; (b) Zelnik-Manor and Perona' algorithm (ZMP), restricted to the final assignment of points to clusters [27]; (c) maximal meaningful clusters, revising elongated clusters (MMC+R); (d) maximal meaningful clusters, without revising elongated clusters (MMC–R), i.e. by omitting lines 11 to 18 on Algorithm 2.

It is important to note that we do not propose a specifically designed method to solve the final assignment problem in Normalized Cuts, but a general clustering algorithm. In this work, we use this algorithm to cluster sets of point within the Normalized Cuts framework.

In Figures 13 and 14, the spectral embedding is constructed with $M = 3$. This is based on two reasons. First, choosing three regions seems to be a reasonable choice in both experiments. Second, visual inspection of point clouds is easier (otherwise, dimensionality reduction techniques should be applied and results would actually depend also on the performance of these techniques).

By looking directly at the embeddings in Figures 13 and 14, it is straightforward to see that the clusters detected by YS and by ZMP differ from the results that one should have expected. The proposed method yields detections which seem to be more adequate to the point clouds structure. In Figure 13 we perceive that the segmentation obtained with the elongated clusters revision step (Section 3.4) is globally better than the one which omits this step. The opposite situation occurs in Figure 14, where disabling the revision allows to detect the balcony. As a side effect, the sky is split in three regions, which roughly correspond to different brightness resulting from the *degradé* of the sky.

In Figure 15, the embeddings are constructed with $M = 10$. In some cases, YS and ZMP perform better while in others the proposed method produces more satisfactory results. All methods oversplit or undersplit clusters in different cases. In general, we think there is no clear winner for these relatively complex scenes. However, both in ZMP and in the proposed approach, contrarily to

YS, the number of clusters is not chosen in advance. Moreover, in contrast to ZMP, our method is general in the sense that it was not specifically designed for Normalized Cuts and can be used in other scenarios.

5. Final Remarks

The proposed method satisfies Zahn's requirements for a perceptual clustering technique. On the one hand, the algorithm does not involve any random choice since it is completely deterministic. On the other hand, once distances have been computed, the method is independent from the dimension of the points (in this case, the dimension of the embedding). Its running time is not affected by an increase in dimensionality: it does not suffer from the "curse of dimensionality". In addition using a more complicated, time consuming distance function is transparent to our method.

The number of clusters is automatically determined, eliminating a classical parameter that is usually hard to choose. It is replaced by ε which has a more intuitive meaning: it controls the average number of false detections. Tuning its value is not necessary since setting $\varepsilon = 1$ is sufficient in practice.

Detection thresholds are easily computed from ε by performing Monte Carlo simulations. These thresholds are well adapted to accept/reject non-clustered data. Experimental results support this claim. Indeed, our method correctly finds the number of clusters and the detected clusters are perceptually significant. Moreover, detections are highly stable since clusters have NFAs well below the estimated thresholds.

Results show that the technique is shape independent. Although our base algorithm has a bias towards non-elongated clusters, a simple heuristic rule is able to correct that situation and yield correct results for a wide range of shapes.

Finally, the exploration rule in Section 3.3 allows for a reasonable computational complexity of $O(N^2 \cdot \log N)$, detailed in Appendix A. When N is large, although the total complexity is a low-degree polynom, handling a fully connected graph is costful, no matter how simple the computed operations are.

The implementation in its current state can not handle graphs with more than twenty thousands nodes. The computation time of maximal meaningful clusters for a graph of such size takes between one and three minutes.

Appendix A. Temporal complexity

Kruskal's algorithm for computing the minimal spanning tree has a complexity of $\mathrm{O}(|E| \cdot \log |E|)$, as the edge set E in G must be sorted. There are faster algorithms such as Prim's but optimal computation of the minimum spanning tree is not the goal of this work. Kruskal's algorithm is sped-up by using a union-find algorithm on a disjoint-sets data structure [33]. After sorting edges, union-find allows to build the minimal spanning tree $\mathcal{T}$ in quasi-linear time. More precisely, its worst-case complexity is $\mathrm{O}(|E| \cdot \alpha(|E|))$ where α is the extremely slow-growing inverse Ackermann function. In practice $\alpha(M) < 4$.

Computing the set of edges of each node in $\mathcal{T}$ can be done in $\mathrm{O}(|E| \cdot \log |E|)$. The computation of the binomial tail is done by using the incomplete beta function which is constant in time [34].

Since there are $|V|$ nodes in G, $|\mathcal{T}| = 2|V| - 1$. Computing PFA requires therefore $2|V| - 1$ computations. All nodes in $\mathcal{T}$ are examined during the maximality check, which also amounts to $2|V| - 1$ computations. Maximal meaningful clusters algorithm itself is therefore linear in the number of nodes, i.e. $\mathrm{O}(|V|)$.

As G is fully connected, $|V| \leq |E| = \frac{|V| \cdot (|V|-1)}{2}$ and since $|V| = |X| = N$, $\mathrm{O}(|E| \cdot \log |E|) = \mathrm{O}\left(N^2 \cdot \log(N)\right)$.

Acknowledgments

We thank the reviewers for their comments and suggestions, which significantly contributed to improve the quality of the publication. We acknowledge financial support by CNES (R&T Echantillonnage Irregulier DCT/SI/MO-2010.001.4673), FREEDOM (ANR07-JCJC-0048-01), Callisto (ANR-09-CORD-003), ECOS Sud U06E01, ARFITEC (07 MATRH) and the Uruguayan Agency for Research and Innovation (ANII) under grant PR-POS-2008-003.

References

[1] A. Ng, M. Jordan, Y. Weiss, On spectral clustering: Analysis and an algorithm, in: Adv. Neural Inf. Process. Syst., 2001, pp. 849–856.

[2] D. Comaniciu, P. Meer, Mean shift: a robust approach toward feature space analysis, IEEE Trans. Pattern Anal. Mach. Intell. 24 (2002) 603–619.

[3] P. Felzenszwalb, D. Huttenlocher, Efficient graph-based image segmentation, Int. J. Comput. Vis. 59 (2004) 167–181.

[4] B. Leibe, K. Mikolajczyk, B. Schiele, Efficient clustering and matching for object class recognition, in: Proc. British Mach. Vis. Conf., 2006.

[5] F. Cao, J. Delon, A. Desolneux, P. Musé, F. Sur, A unified framework for detecting groups and application to shape recognition, J. Math. Imaging and Vis. 27 (2007) 91–119.

[6] I. Dhillon, Y. Guan, B. Kulis, Weighted graph cuts without eigenvectors a multilevel approach, IEEE Trans. Pattern Anal. and Mach. Intell. 29 (2007) 1944–1957.

[7] G. Flake, R. Tarjan, K. Tsioutsiouliklis, Graph clustering and minimum cut trees, Internet Mathematics 1 (2004) 385–408.

[8] R. Kannan, S. Vempala, A. Vetta, On clusterings: Good, bad and spectral, Journal of the ACM 51 (2004) 497–515.

[9] J. Kleinberg, An impossibility theorem for clustering, in: Adv. Neural Inf. Process. Syst., 2002, pp. 446–453.

[10] A. K. Jain, Data clustering: 50 years beyond k-means, Pattern Recognition Letters 31 (2010) 651–666.

[11] S. Bandyopadhyay, An automatic shape independent clustering technique, Pattern Recognition 37 (2004) 33–45.

[12] C. Fowlkes, S. Belongie, F. Chung, J. Malik, Spectral grouping using the nyström method, IEEE Trans. Pattern Anal. and Mach. Intell. 26 (2004) 214–225.

[13] J. Shi, J. Malik, Normalized cuts and image segmentation, IEEE Trans. Pattern Anal. and Mach. Intell. 22 (2000) 888–905.

[14] C. T. Zahn, Graph-theoretical methods for detecting and describing gestalt clusters, Transactions on Computers C-20 (1971) 68–86.

[15] M. Wertheimer, Laws of organization in perceptual forms, Routledge and Kegan Paul, pp. 71–88.

[16] A. Desolneux, L. Moisan, J. M. Morel, Edge detection by helmholtz principle, J. Math. Imaging and Vis. 14 (2001) 271–284.

[17] F. Chung, Spectral Graph Theory (CBMS Regional Conference Series in Mathematics, No. 92), American Mathematical Society, 1997.

[18] U. von Luxburg, M. Belkin, O. Bousquet, Consistency of spectral clustering, Annals of Statistics 36 (2008) 555–586.

[19] K. Burnham, D. Anderson, Model selection and multimodel inference: a practical information-theoretic approach, Springer, 2nd edition, 2002.

[20] A. Barron, J. Rissanen, B. Yu, The minimum description length principle in coding and modeling, IEEE Trans. Inf. Theory 44 (1998) 2743–2760.

[21] B. Nadler, M. Galun, Fundamental limitations of spectral clustering, in: Adv. Neural Inf. Process. Syst., volume 19, 2007, pp. 1017–1024.

[22] U. Ozertem, D. Erdogmus, R. Jenssen, Mean shift spectral clustering, Pattern Recognition 41 (2008) 1924–1938.

[23] A. Desolneux, L. Moisan, J. M. Morel, From Gestalt Theory to Image Analysis, volume 34, Springer-Verlag, 2008.

[24] K. Fukunaga, Introduction to Statistical Pattern Recognition, Academic Press, second edition, 1999.

[25] R. Hoffman, A. K. Jain, A test of randomness based on the minimal spanning tree, Pattern Recognition Letters 1 (1983) 175–180.

[26] A. K. Jain, X. Xu, T. K. Ho, F. Xiao, Uniformity testing using minimal spanning tree, in: Int. Conf. on Pattern Recognit., 2002, pp. 281–284.

[27] L. Zelnik-Manor, P. Perona, Self-tuning spectral clustering, in: Adv. Neural Inf. Process. Syst., volume 17, MIT Press, 2004, pp. 1601–1608.

[28] N. Burrus, T. M. Bernard, J. M. Jolion, Image segmentation by a contrario simulation, Pattern Recognition 42 (2009) 1520–1532.

[29] K. Fukunaga, L. Hostetler, The estimation of the gradient of a density function, with applications in pattern recognition, IEEE Trans. Inf. Theory 21 (2003) 32–40.

[30] D. Martin, C. Fowlkes, D. Tal, J. Malik, A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics, in: Proc. IEEE Int. Conf. on Comput. Vis., volume 2, 2001, pp. 416–423.

[31] T. Cour, F. Benezit, J. Shi, Spectral segmentation with multiscale graph decomposition, in: Proc. IEEE Conf. Comput. Vis. Pattern Recognit., volume 2, Washington, DC, USA, 2005, pp. 1124–1131.

[32] S. Yu, J. Shi, Multiclass spectral clustering, in: Proc. IEEE Int. Conf. on Comput. Vis., 2003, pp. 313–319 vol.1.

[33] R. Tarjan, J. Van Leeuwen, Worst-case analysis of set union algorithms, Journal of the ACM 31 (1984) 245–281.

[34] W. Press, S. Teukolsky, W. Vetterling, B. Flannery, Numerical Recipes in C, Cambridge University Press, Cambridge, UK, 2nd edition, 1992.

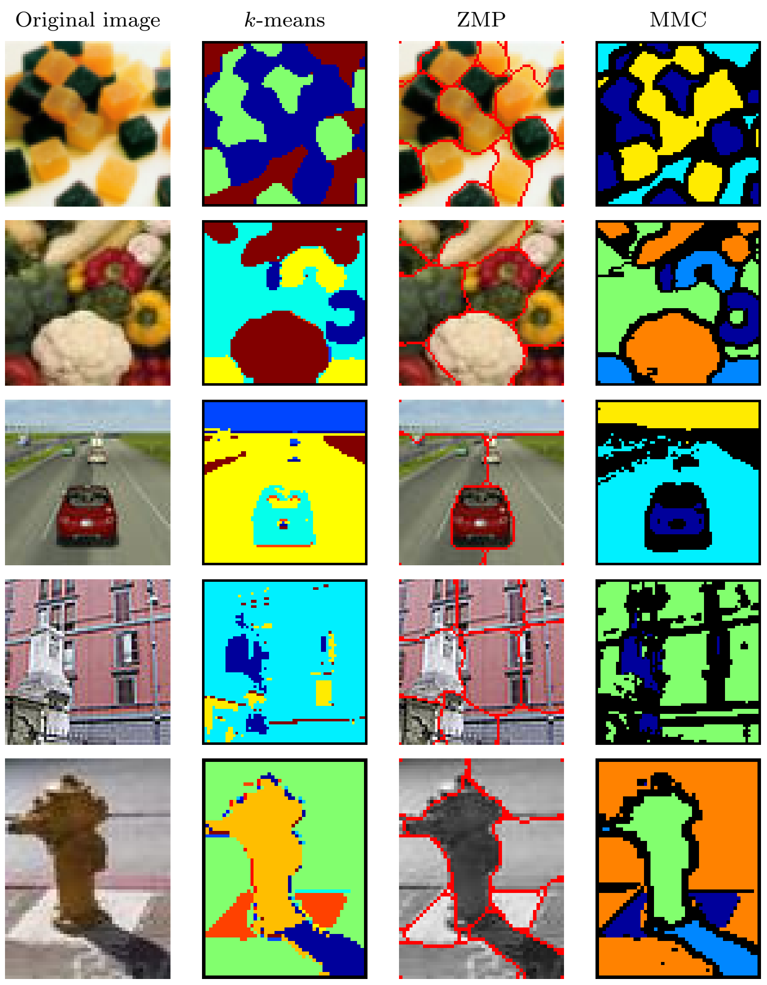

Figure 11: Comparison of image segmentations from 3×3 color patches. Results with k-means were obtained by tuning k by hand. In all examples, for maximal meaningful clusters non-detected areas are depicted in black.

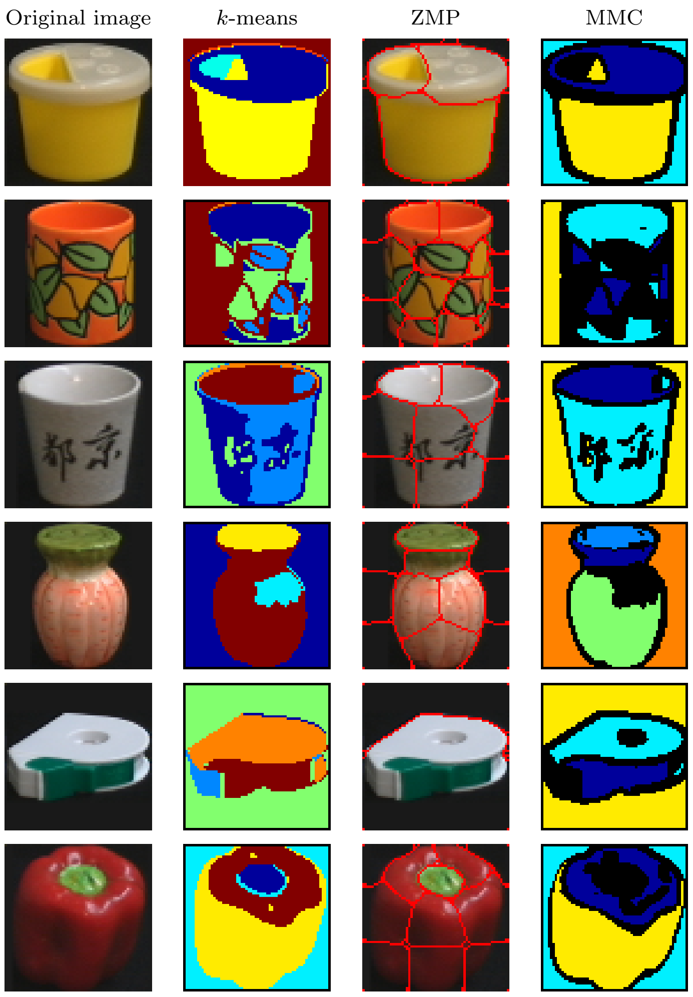

Figure 12: Comparison of image segmentations from 3×3 color patches. Results with k-means were obtained by tuning k by hand. In all examples, for maximal meaningful clusters non-detected areas are depicted in black.

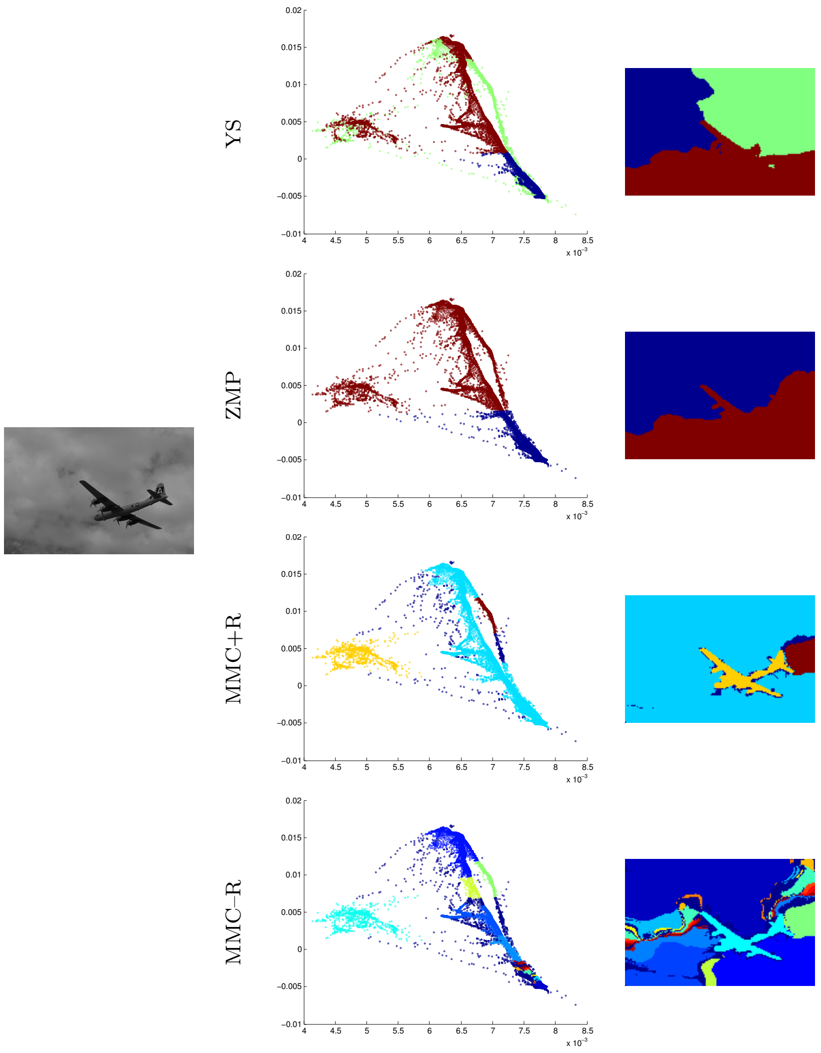

Figure 13: In the original image, we perceive two or three main regions: the plane and one or two regions on the textured sky. On the center column, the point cloud on the left, which corresponds to the airplane, is clearly separated from the rest. Neither YS nor ZMP detect it as an individual cluster. The proposed method is able to detect it as a separate cluster.

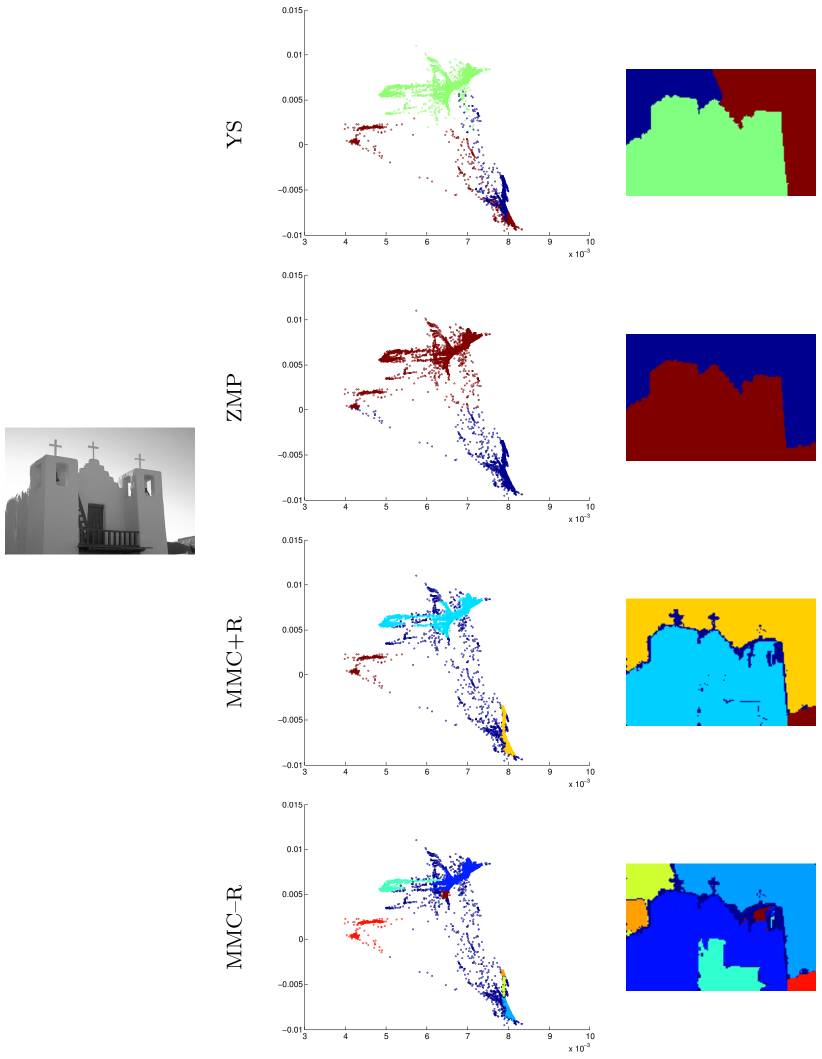

Figure 14: Four main regions are perceived in the original image: sky, church, balcony and bottom right dark area. On the center column, the point cloud on the left, which corresponds to bottom right are on the original image, is clearly separated from the rest. Neither YS nor ZMP detect it as an individual cluster. The proposed method is able to detect it as a separate cluster. In MMC–R, the balcony stands out as a separate region, and the sky is split in three regions (due to the *degradé* of the sky).

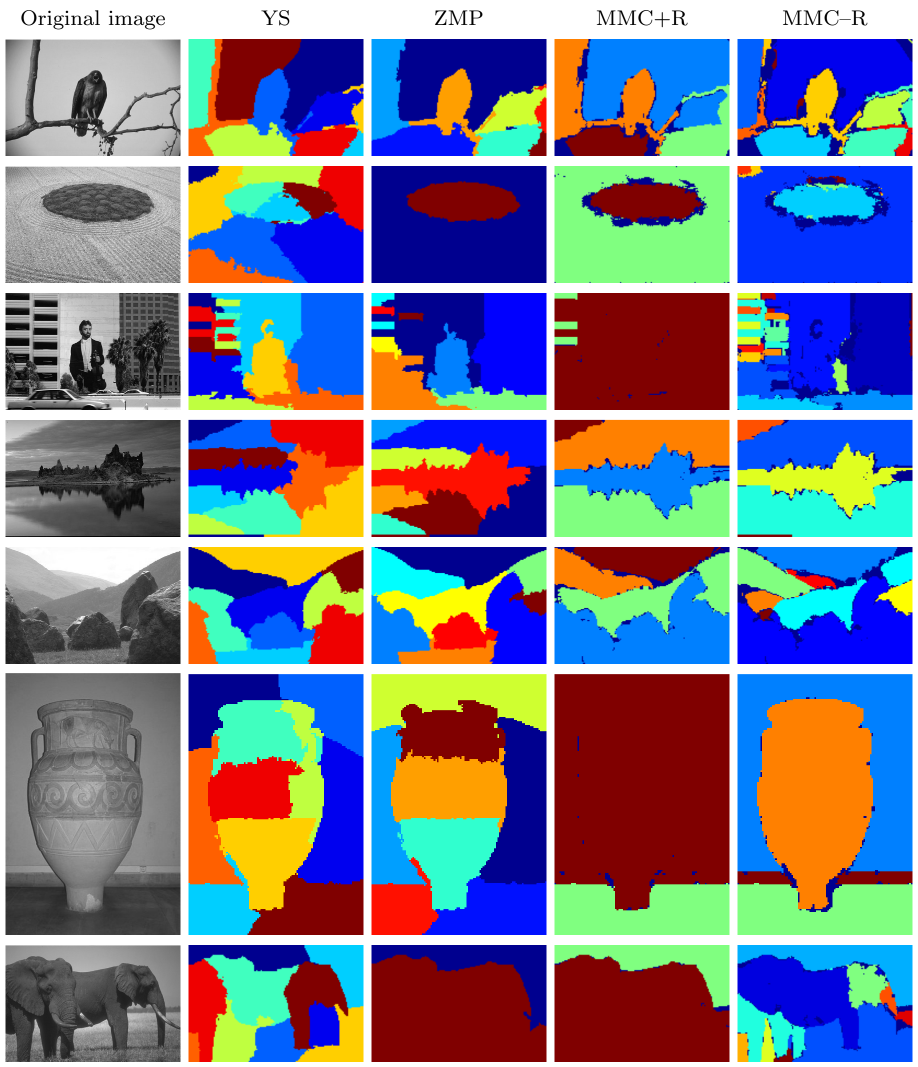

Figure 15: All compared methods produce better results for some images and worst ones for others. None of them clearly outperforms the others: depending on the image, clusters are over or undersplit.