Image Partitioning with Graph Pyramids 1)

Yll Hazhimusa and Walter G. Kropatsch
Pattern Recognition and Image Processing Group 183/2,
Institute for Computer Aided Automation, Vienna University of Technology
{yll, krw}@prip.tuwien.ac.at

Abstract:
We present a hierarchical partitioning of images using a pairwise similarity function on a graph-based representation of an image. This function measures the difference along the boundary of two components relative to a measure of differences of the components' internal differences. This definition tries to encapsulate the intuitive notion of contrast. Two components are merged if there is a low-cost connection between them. Each component’s internal difference is represented by the maximum edge weight of its minimum spanning tree. External differences are the smallest weight of edges connecting components. We use this idea for building a minimum spanning tree to find region borders quickly and effortlessly in a bottom-up way, based on local differences in a specific feature.

1 Introduction

Wertheimer [17] has formulated the importance of wholes (Ganzen) and not of its individual elements as: “There are wholes (Ganzen), the behaviour of which is not determined by that of their individual elements, but where the part-processes are themselves determined by the intrinsic nature of the whole” [18], and introduced the importance of perceptual grouping and organization in visual perception. Low-level cue image segmentation cannot and should not produce a complete final “good” segmentation. The low-level coherence of brightness, color, texture or motion attributes should be used to come up sequentially with hierarchical partitions [15]. Mid and high level knowledge can be used to either confirm these groups or select some for further attention. A wide range of computational vision problems could make use of segmented images, where such segmentations rely on efficient computation. For instance motion estimation requires an appropriate region of support for correspondence operation. Higher-level problems such as recognition and image indexing can also make use of segmentation results in the problem of matching. It is important that a grouping method has the following properties [3]:

- capture perceptually important groupings , which reflect global aspects of the image,
- be highly efficient, running in time linear in the number of image pixels,

1) This paper has been supported by the Austrian Science Fund under grants P14445-MAT and P14662-INF
In a regular image pyramid the number of pixels at any level \( k \), is \( r \) times higher than the number of pixels at the next reduced level \( k + 1 \). The so called reduction factor \( r \) is greater than one and it is the same for all levels \( k \). If \( s \) denotes the number of pixels in an image \( I \), the number of new levels on top of \( I \) amounts to \( \log_r(s) \). Thus, the regular image pyramid may be an efficient structure for fast grouping and access to image objects in bottom-up and top-down processes [14]. However, regular image pyramids are confined to globally defined sampling grids and lack shift invariance. Bister et.al. [1] concludes that regular image pyramids have to be rejected as general-purpose segmentation algorithms. In [13, 7] it was shown how these drawbacks can be avoided by irregular image pyramids, the so called adaptive pyramids, where the hierarchical structure (vertical network) of the pyramid was not “a priori” known but recursively built based on the data. Moreover in [12, 4] it was shown that the irregular pyramid can be used for segmentation and feature detection.

Each level represents a partition of the pixel set into cells, i.e. connected subsets of pixels (CC). The construction of an irregular image pyramid is iteratively local [11, 6]. This means that we use only local properties to build the hierarchy of the pyramid. On the base level (level 0) of an irregular image pyramid the cells represent single pixels and the neighborhood of the cells is defined by the 4 (8)-connectivity of the pixels. A cell on level \( l + 1 \) (parent) is a union of neighboring cells on level \( l \) (children). This union is controlled by so-called contraction kernels (decimation parameters) [9]. Every parent computes its values independently of other cells on the same level. This implies that an image pyramid is built in \( O(\log(\text{image diameter})) \) time. Neighborhoods on level \( k + 1 \), are derived from neighborhoods on level \( k \). Two cells \( c_1 \) and \( c_2 \) are neighbors if there exist pixels \( p_1 \) in \( c_1 \) and \( p_2 \) in \( c_2 \) such that \( p_1 \) and \( p_2 \) are 4-neighbors, as seen in Figure 1a). We assume that on each level \( k + 1 \) \((k \geq 0) \) there exists at least one cell not contained in level \( k \). In particular, there exists a highest level \( h \). In general the top of the pyramid can have one vertex, i.e. an apex. We represent the levels as dual pairs \((G_k, \overline{G}_k)\) of plane graphs \( G_k \) and its dual (plane) graph \( \overline{G}_k \) (Figure 1b)). The vertices of \( G_k \) represent the cells and the edges of \( G_k \) represent the neighborhood relations of the cells on level \( k \), depicted with square vertices and dashed edges in Figure 1b). The edges of \( \overline{G}_k \) represent the borders of the cells on level \( k \), depicted with solid lines in Figure 1b), possibly including so called
pseudo edges needed to represent the neighborhood relation to a cell completely surrounded by another cell. Finally, the vertices of $G_k$ (the circles in Figure 1b), represent meeting points of at least three edges from $G_k$, solid lines in Figure 1b). The sequence $(G_k, \overline{G_k}), 0 \leq k \leq h$ is called a (dual) graph pyramid, where $h$ is the highest level in the pyramid. Moreover the graph is attributed $G(V, E, a_v, a_e)$, i.e. $a_v : V \rightarrow \mathbb{R}^+$ and $a_e : E \rightarrow \mathbb{R}^+$. We use a weight for $attr_e$ measuring the contrast between the two end points.

The aim of this paper is to build in parallel a minimum weight spanning tree (MST) to find region borders quickly and effortlessly in a bottom-up 'stimulus-driven' way based on local differences in a specific feature, like in preattentive vision. For more in depth on the subject see the book of Jolion [8]. This goal is reached by using the selection method for contraction kernels proposed in [5] to achieve logarithmic tapering, local construction and shift invariance. Borůvka's algorithm [2] with the dual graph contraction algorithm [9] is used for building in parallel (hierarchical) a minimum weight spanning tree (of the region) and at the same time to preserve topology. The topological relation seems to play an even more important role for vision tasks in natural systems than precise geometrical position. The plan of the paper is as follows. In Sec. 2 we give the merging decision criteria and in Sec. 2.1 we prove that the proposed algorithm builds the MST. Sec. 3 reports on experimental results.

2 A Hierarchy of Partitioning

The goal is to find partitions $P_k = \{ CC^k_1, CC^k_2, \ldots, CC^k_n \}$ on level $k$ of the pyramid, such that these elements satisfy certain properties. We use the pairwise comparison of neighboring vertices (regions) to check for similarities [3, 4]. A pairwise comparison function, $Comp(CC^k_i, CC^k_j)$ is true, if there is evidence for a boundary between $CC^k_i$ and $CC^k_j$, and false when there is no boundary. Note that $Comp(CC^k_i, CC^k_j)$ is a boolean comparison function for pairs of partitions. The definition of $Comp(CC^k_i, CC^k_j)$ depends on the application. This function measures the difference along the boundary of two components relative to a measure of differences of components' internal differences, and tries to encapsulate the intuitive notion of local contrast: a contrasted zone is a region containing two connected components whose inner differences (internal contrast) are less than differences with between them (external contrast).

Every vertex $u_k \in G_k$ is a representative of a “homogeneous” region $CC^k$ on the base level of the pyramid, i.e. this region is represented by a $MST(u_k) = CC^k$. The internal contrast of the $CC^k$ is the largest dissimilarity measure i.e. the largest edge weight of the $MST(u_k)$ of the vertex $u_k \in G_k$, defined as

$$\text{Int}(CC^k) = \max\{attr_e(e), e \in MST(u_k)\}.$$  \hfill (1)

Let $u_{k,i}, u_{k,j} \in V$ be the end vertices of an edge $e \in E$. The external contrast between two components $CC^k_i, CC^k_j \in P_k$ is the smallest dissimilarity between components $CC^k_i$ and
$CC_i^k$ i.e. the smallest edge weight connecting $MST(u_{k,i})$ and $MST(u_{k,j})$ defined as

$$Ext(CC_i^k, CC_j^k) = \text{min}\{\text{attr}_e(e), e = (u_{k,i}, u_{k,j})|u_{k,i} \in MST(u_{k,i}) \land u_{k,j} \in MST(u_{k,j})\}. \quad (2)$$

This definition is problematic since it uses only the “smallest” edge weight between the two components, making the method very sensitive to noise, but in practice this limitation works well as shown in Sec. 3. In Fig. 1c) is shown a simple example of internal contrast, $Int(CC_i^k)$ of the component $CC_i^k$, as the maximum of weights of the solid line edges, and external contrast, $Ext(CC_i^k, CC_j^k)$, as the minimum of weights of the dashed line edges connecting component $CC_i^k$ and $CC_j^k$. Vertices $u_{k,i}$ and $u_{k,j}$ are representative of the components $CC_i^k$ and $CC_j^k$, and hold as attribute the maximum edge weight of its $MST$, whereas the edge $e$, connecting the vertices holds the minimum edge weight. By contracting the edges of $MST(u_{k,i})$ one arrives at the vertex $u_{k,i}$, analogously for $MST(u_{k,j})$. The pairwise comparison between two connected components $CC_i^k$ and $CC_j^k$ is defined by:

$$\text{Comp}(CC_i^k, CC_j^k) = \begin{cases} 
\text{True} & \text{if } Ext(CC_i^k, CC_j^k) > PInt(CC_i^k, CC_j^k), \\
\text{False} & \text{otherwise},
\end{cases} \quad (3)$$

where the $PInt(CC_i^k, CC_j^k)$ is the minimum internal contrast defined as,

$$PInt(CC_i^k, CC_j^k) = \text{min}(\text{Int}(CC_i^k) + \tau(CC_i^k), \text{Int}(CC_j^k) + \tau(CC_j^k)). \quad (4)$$

For the function $\text{Comp}(CC_i^k, CC_j^k)$ to be true i.e. for the border to exist, the external contrast difference must be greater than the internal contrast. The reason for using a threshold function $\tau(CC^k)$ is that for small components $CC^k$, $Int(CC^k)$ is not a good estimate of the local characteristics of the data, in the extreme case when $|CC^k| = 1$, $Int(CC^k) = 0$. Any non-negative function of a single component $CC$ can be used for $\tau(CC^k)$ [3].

### 2.1 Building A Hierarchy of Partitions: The Algorithm

First we give two lemmas that provide the basis of the minimum weight spanning tree algorithms which help us in proving the Proposition 1. Proofs can be found in [16].

**Lemma 2.1** Consider a vertex $v$ in a weighted connected graph $G$. Among all the edges incident on $v$, let $e$ be one of minimum weight. Then, $G$ has a minimum weight spanning tree that contains $e$.

**Lemma 2.2** Let $T$ be an acyclic subgraph of a weighted connected graph $G$ such that there exists a minimum weight spanning tree containing $T$. If $G'$ denotes the graph obtained by contracting the edges of $T$, and $T_{\text{min}}'$ is a minimum weight spanning tree of $G'$, then $T_{\text{min}}' \cup T$ is a minimum weight spanning tree of $G$.

With the definition of the comparison function $\text{Comp}(\cdot,\cdot)$ we can now build the hierarchy of partitions as follows:
Algorithm 1 - Algorithm: Hierarchy of Partitions

**Input:** Attributed graph $G_0$.

1: $k = 0$
2: repeat
3: for all vertices $u \in G_k$ do
4: \[ E_{\text{min}}(u) = \arg\min\{\text{attr}_e(e) | e = (u, v) \in E_k \text{ or } e = (v, u) \in E_k\} \]
5: end for
6: for all $e = (u_{k,i}, u_{k,j}) \in E_{\text{min}}$ with $\text{Ext}(CC_i^k, CC_j^k) \leq \text{PInt}(CC_i^k, CC_j^k)$ do
7: include $e$ in contraction edges $N_{k,k+1}$
8: end for
9: contract graph $G_k$ with contraction kernels, $N_{k,k+1}: G_{k+1} = C[G_k, N_{k,k+1}]$.
10: for all $e_{k+1} \in G_{k+1}$ do
11: set edge attributes $\text{attr}_e(e_{k+1}) = \min\{\text{attr}_e(e_k) | e_{k+1} = C(e_k, N_{k,k+1})\}$
12: end for
13: $k = k + 1$
14: until $G_k = G_{k-1}$

**Output:** A region adjacency graph (RAG) pyramid, where each vertex is representative of a $MST$ of a region.

Each vertex $u_k \in G_k$ i.e. $CC^k$ represents a connected region on the base level of the pyramid, and since the presented algorithm is based on Borovka’s algorithm [2], it builds a $MST(u_k)$ of each region, i.e $N_{0,k}(u_k) = MST(u_k)$. The idea is to collect the smallest weighted edges $e$ (4th step) that could be part of the $MST$, and then to check if the edge weight $\text{attr}_e(e)$ is smaller than the internal contrast of both of the components ($MST$ of end vertices of $e$) (6th step). If these conditions are fulfilled then these two components will be merged (7th step). Two regions will be merged if the internal contrast, which is represented by its $MST$, is larger than the external contrast, represented by the weight of the edge, $\text{attr}_e(e)$. All the edges to be contracted form the contraction kernels $N_{k,k+1}$, which then are used to create the graph $G_{k+1} = C[G_k, N_{k,k+1}]$ [10], so that the topology is preserved. In general $N_{k,k+1}$ is a forest. We update the attributes of those edges $e_{k+1} \in G_{k+1}$ with the minimum attribute of the edges $e_k \in E_k$ that are contracted into $e_{k+1}$ (11th step). The output of the algorithm is a pyramid where each level represents a RAG, i.e a partition. Each vertex of these RAGs is the representative of a $MST$ of a region in the image. The algorithm is greedy since it collects only the nearest neighbor with the minimal edge weights and merges them if Eq. 3 is false.

Let us prove that the Algorithm 1 builds the $MST$. The proof is based on Kruskal’s proof [16].

**Proposition 1** The Algorithm 1 constructs a minimum weight spanning tree of a weighted connected graph $G$. 
**Proof.** Let $G$ be the given nontrivial weighted connected graph. Also let us suppose that the criterion $Ext(CC^k_i, CC^k_j) \leq PINt(CC^k_i, CC^k_j)$ is fulfilled for all edges, this implies that the algorithm becomes Borůvka’s algorithm. Clearly, when Borůvka’s algorithm terminates the selected tree $T_{\text{min}}$ is a spanning tree. Thus we have to show that $T_{\text{min}}$ is indeed a minimum weight spanning tree of $G$ by proving that every $T_i$ constructed in the course of Borůvka’s algorithm is contained in a minimum weight spanning tree of $G$. Our proof is by induction on $i$. The subgraph $T_{i+1}$ is constructed from $T_i$ by adding an edge of minimum weight with exactly one end vertex in $T_i$. This construction ensures that all $T_i$’s are connected. As inductive hypothesis assume that $T_i$ is contained in a minimum spanning tree of $G$. If $G'$ denotes the graph obtained by contracting the edges of $T_i$ and $v'$ denotes the vertex of $G'$, which corresponds to the vertex set of $T_i$, then $e_{i+1}$ is in fact a minimum weight edge incident on $v'$ in $G'$. Clearly by Lemma 2.1 the edge $e_{i+1}$ is contained in a minimum weight spanning tree $T_{\text{min}}^i$ in of $G'$. By Lemma 2.2, $T_i \cup T_{\text{min}}^i$ is a minimum weight spanning tree of $G$. More specifically $T_{i+1} = T_i \cup \{e_{i+1}\}$ is contained in a minimum weight spanning tree of $G$ and the correctness of Borůvka’s algorithm follows. \[\square\]

## 3 Experiment Results on Grid Graphs

We use as attributes of edges the difference between pixel intensities $attr_e(u_i, u_j) = |I(u_i) - I(u_j)|$. For color images we run the algorithm by computing the distances (weights) in $RGB$ color space. To compute the hierarchy of partitions we define $\tau(CC)$ as $\tau(CC) = \alpha/|CC|$, where $\alpha = \text{const}$ and $|CC|$ is the number of elements in $CC$, i.e. the size of the region. The algorithm has one running parameter $\alpha$, which is used to compute the function $\tau$. A larger constant $\alpha$ sets the preference for larger components. A more complex definition of $\tau(CC)$, which is large for certain shapes and small otherwise would produce a partitioning which prefers certain shapes, e.g. using ratio of perimeter to area would prefer components that are not long and thin. For speed purposes we store in vertices the internal contrast and the size of the connected component (receptive field).

We use indoor $RGB$ images 'Lena’\(^1\)(512×512) and 'Object 45’\(^2\)(128×128), an outdoor image 'Monarch’\(^1\)(768×512) and a synthetic image (223×111) for the experiments. We found that $\alpha = 300$ produces the best hierarchy of partitions of the images shown in Fig. 2(a,d,g) and $\alpha = 1000$ for the image under (j), after the average intensity attribute of vertices is down-projected onto the base grid. Fig. 2 shows some of the partitions on different levels of the pyramid and the number of components. In all images there are regions of large intensity variability and gradient. This algorithm copes with this kind of variability. In contrast to [3] the result is a hierarchy of partitions as multiple resolution suitable for further goal driven, domain specific analysis. Since the algorithm preserves details in low-variability regions, a

---

\(^1\)Waterloo image database

\(^2\)Coil 100 image database
Figure 2: Some levels of the partitioning and the number of components.

noisy pixel would survive through the hierarchy. Of course, image smoothing in low variability regions would overcome this problem. We, however, do not smooth the images, as this would introduce another parameter into the method. The hierarchy of partitions can also be built from an oversegmented image to overcome the problem of noisy pixels. Note that the influence of $\tau$ in decision criterion is smaller as the region gets bigger. For an oversegmented image, where the size of regions are large, the algorithm becomes parameterless.

4 Conclusion and Outlook

In this paper we have introduced a method to build a hierarchy of partitions of an image by comparing in a pairwise manner the difference along the boundary of two components relative to the differences of components’ internal differences. Even though the algorithm makes simple greedy decisions locally, it produces perceptually important partitions in a bottom-up 'stimulus-driven' way based only on local differences. It was shown that the algorithm
can handle large variation and gradient intensity in images. Since our framework is general enough, we can use RAGs of any oversegmented image and build the hierarchy of partitions. External knowledge can help in a top-down segmentation technique. A drawback is that the maximum and minimum criterion is very sensitive to noise, although in practice it has a small impact. Other criteria like median would lead to an NP-complete algorithm. The algorithm has only one running parameter which controls the sizes of the regions. Our future work is to automatically extract this parameter from the image and also to define different comparison function which will prefer regions of specific shapes.

References