Viscous-Hyperconnected Attribute Filters: A First Algorithm

Ugo Moschini and Michael H.F. Wilkinson

Institute for Mathematics and Computing Science,
University of Groningen, P.O. Box 407, 9700 AK, Groningen, The Netherlands*
{u.moschini,m.h.f.wilkinson}@rug.nl

Abstract. In this paper a hyperconnectivity class that tries to address the leakage problem typical of connected filters is used. It shows similarities with the theory of viscous lattices. A novel algorithm to perform attribute filtering of viscous-hyperconnected components is proposed. First, the max-tree of the image eroded by a structuring element is built: it represents the hierarchy of the cores of the hyperconnected components. Then, a processing phase takes place and the node attributes are updated consistently with the pixels of the actual hyperconnected components. Any state-of-the-art algorithm can be used to build the max-tree of the component cores. An issue arises: edges of components are not always correctly preserved. Implementation and performance are presented. A possible solution is put forward and it will be treated in future work.

Keywords: hyperconnected components, max-tree, attribute filtering

1 Introduction

In image analysis the notion of connectivity [12] defines how pixels are grouped together into connected components. In classical connectivity, those are regions of the image made of pixels that share the same intensity and are path-wise connected. Such regions are also called flat-zones. Attribute filters are operators that filter an image at the level of connected components: flat-zones are removed or preserved through filtering operations [3, 9, 10]. Issues might arise with connected filters based on the definition of connectivity mentioned above. For example, a structure that a human observer would consider as a single connected component might be broken into multiple components by noise or other artefacts. Another example is the leakage problem. It occurs when two different objects are connected by some bridging elements, such as noise or irrelevant image structures. Several solutions have been proposed, some based on connectivity [2, 15, 17], and some which are related but not strictly connected [16]. Connectivity-based solutions first modify the image with some anti-extensive operator and compute connected components based on the modified image [4]. A drawback of this approach is that it boils down to performing an attribute filter based on standard connectivity on the modified image [20] in all practical cases. Leakage is related with

* This work was funded by the Netherlands Organisation for Scientific Research (NWO) under project number 612.001.110.

The final publication is available at link.springer.com.
DOI: 10.1007/978-3-319-18720-4_56.

(a) (b) (c) (d)

Fig. 1: (a) original image; images after deletion of viscous-hyperconnected components (b) and standard connected components (c) and (d) with area smaller than 10, 10 and 1000 pixels, respectively. Even for large values of area threshold, the thin white stripes of the veil are preserved in (c) and (d) due to the leakage problem. The cores of hyperconnected components were obtained eroding the image with a 5x5 disk.

a more general problem present in standard connectivity: spatial relationships among components as well as their grey level connection are not taken into account. That is, overlapping among flat-zones at the same intensity is not possible because the zones would be considered as one component. Hyperconnectivity [8, 14, 21, 23] and attribute-space connectivity [20] were created to overcome these limitations. In this paper, we will use a hyperconnectivity class as a possible solution to the problem of leakage. It shows similarities with viscous lattices [13] and it is referred to as viscous hyperconnection. Different viscosity indexes are given by different sizes of structuring element used in the extensive dilation operation that defines a viscous lattice. Fig. 1 shows the different output of connected and hyperconnected filtering, which illustrates that leakage is stopped by using viscous-hyperconnected components. In this work, we propose an algorithm based on max-trees [9] to perform attribute filtering of viscous-hyperconnected components. The implementation is analysed and performance and drawbacks are discussed.

2 Viscous-Hyperconnectivity class

Two axiomatics for hyperconnectivity were defined in the recent years by Wilkinson in [21, 23] and by Perret et al. in [8]. It was noted in [8] that the definition of an overlap criterion as in [21] is not either required or needed to derive any new property for hyperconnections. In this paper we restrict the definition of a hyperconnectivity class to the following three axioms:

Definition 1 *A hyperconnectivity class $\mathcal{H} \subseteq \mathcal{L}$, with $\mathcal{L}$ a complete lattice is a class with the following properties:*

1. $\mathbf{0} \in \mathcal{H}$
2. *$\mathcal{H}$ is sup-generating*
3. *$\mathcal{H}$ is chain-sup complete,*

The final publication is available at link.springer.com.
DOI: 10.1007/978-3-319-18720-4_56.

Informally, the second property states that every possible subset of the image space can be constructed as supremum of elements in $\mathcal{H}$. The third property states that the supremum of any totally ordered subset $\mathcal{H}$ also is in $\mathcal{H}$. We refer to the works in [14,23] for a more thorough explanation. Hyperconnectivity can generalise a large number of operators from edge-preserving connected filters to structural filters [5, 21, 22] into a single framework. This vast generalization leads to the problem of defining meaningful hyperconnections useful for image processing purposes since a very broad range of possibilities is open up. A recent theory [6] has been developed that unifies connectivity and hyperconnectivity defining axiomatics for both.

In this section a hyperconnectivity class is illustrated as a possible solution to the problem of leakage typical of connected filters. This class and its connection to viscous lattices [13] have been already put forward in [21]. It is inspired by the process of constrained reconstruction from markers in [19], a solution to prevent objects linked by narrow structures from being erroneously reconstructed. To define this hyperconnectivity class, let us start from defining a structuring element B as a ball centred on the origin, and $\mathcal{C}$ some connectivity class on $\mathcal{P}(\mathcal{L})$, the power set of a lattice $\mathcal{L}$. The element B will be considered a flat and connected structuring element in the rest of the paper. Consider the following set:

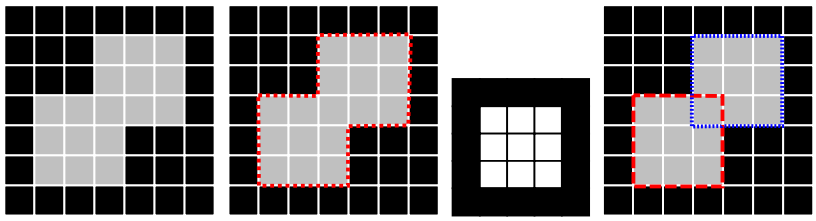
$$\mathcal{H}_B = \{\emptyset\} \cup S \cup \{\mathcal{H} \in \mathcal{P}(\mathcal{L}) \mid \exists C \in \mathcal{C} : \mathcal{H} = \delta_B C\} \tag{1}$$

It represents the set of all dilates by B of all connected sets, augmented with the empty set $\emptyset$ and all the singletons S, in the same way as in the theory of viscous lattices. This set is the viscous-hyperconnectivity class. This is equivalent to stating that the intersection of elements belonging to $\mathcal{P}(\mathcal{L})$ contains at least one translate of B. In viscous-hyperconnectivity, any image is constructed from a series of hyperconnected components which all lie within the opening $\gamma_B f$ and a series of singletons which lie in $f - \gamma_B f$. Fig. 2b shows a connected component in standard connectivity that it is split in the two hyperconnected components in Fig. 2d: in fact, the structuring element in Fig. 2c does not fit at the locations where the two square structures intersect, thus forming two components. Although viscous reconstruction (reconstruction with criteria) from markers has already been studied and linked to viscous hyperconnected filtering in [11], there is currently no method to perform *hyperconnected attribute filtering* [23] based on viscous-hyperconnectivity. A solution is proposed in the next section.

3 Viscous-Hyperconnected Filtering

The max-tree [9] structure is commonly used to perform efficient attribute computation and filtering on the image connected components. Each of the nodes of the tree represents a peak component of a grey level image f. Peak components at a grey level h are connected components of threshold sets at level h for the image f. The image is a set of nested peak components: every node has a single parent pointer to a node at a grey level below its own. Nodes contain usually auxiliary data used to calculate measures of the component, e.g. area or shape features. These measures are referred to as attributes. The root of the tree represents the image background, representing the

The final publication is available at link.springer.com.
DOI: 10.1007/978-3-319-18720-4_56.



(a) (b) (c) (d)

Fig. 2: (b) the dashed line highlights the boundary of the connected component in (a); (c) 3x3 square structuring element; dashed and dotted lines highlight the two viscous-hyperconnected components in (a) according to the structuring element in (c).

component with lowest intensity in the image. The leaves of the tree represent the highest intensities. The viscous hyperconnected components of an image are also ordered in the same way as their counterparts in standard connectivity. Indeed, the ordering is still given by the flat-zone intensity. The fact that components at the same intensity can overlap, without being merged together, does not interfere with the ordering. The cores of viscous-hyperconnected components can readily be represented in a hierarchical structure like the max-tree. According to the hyperconnectivity class in Section 2, a viscous-hyperconnected set is either a singleton or a dilate of a connected set. A viscous-hyperconnected set can be seen as made of a connected *core* and a *shell*, that is the difference between a core and its dilate. Cores of components are created by eroding the original image with some structuring element that drives the viscosity. The computation of the tree of viscous-hyperconnected components is summarised in the following. First, a max-tree of the eroded image containing the cores is built and attributes are computed. Then, for every pixel p, nodes n are determined so that p is in the shell of the core components corresponding to the nodes n. The attributes of such nodes are updated consistently with those pixels belonging to the shell. Fig. 3a shows a grayscale image with the connected component in light grey highlighted by a red dashed line. With B the structuring element as in Fig. 3b, the light grey component corresponds to two hyperconnected components, shown in Fig. 3c. The same happens for the component in dark grey. The cores corresponding to the components in the eroded image are shown in Fig. 3d. The max-tree of cores with area attributes is shown in Fig. 3e. Parent pointers are indicated with arrows. Fig. 3f indicates the pixels that are part of the overlapping section of the shells after dilation of the cores: their contribution will be counted in the attributes of both dark and light grey hyperconnected components. Filtering is done on the max-tree nodes and the result is dilated by the same structuring element. In our example, the tree of hyperconnected components in Fig. 3g is filtered and components with area larger than 5 pixels are preserved (Fig. 3h). Note that singleton nodes are not represented in the tree, even though they are also hyperconnected components. If they satisfy the filter criterion, they are added directly to the filtered image. Fig 3i shows that the singleton node of unit area corresponding to the white pixel at the centre of the image would have two parent nodes. That is, in general, the actual

The final publication is available at link.springer.com.
DOI: 10.1007/978-3-319-18720-4_56.

structure would not be a tree but a directed acyclic graph, if singletons are considered. A more rigorous definition of the graph structure that originates from hyperconnected components will be given in future work. For the purpose of this work, we analyse the tree of viscous-hyperconnected components that are not singletons.

3.1 The algorithm

The implementation of the tree of cores of hyperconnected components of an image f starts with building a max-tree MTϵ_B of the eroded image $\epsilon_B f$, where B is a flat and connected structuring element, with its origin in the centre. The cores contain a subset of all the pixels of the whole hyperconnected components. MTϵ_B represents the nodes corresponding to the cores: just the attributes are updated with the pixels belonging to the shell of a component. In principle, any state-of-the-art algorithm that builds max-trees can be used to build MTϵ_B and store the auxiliary data (area of the component in the rest of the paper) needed for attribute computation. Sequential or parallel algorithms that follow bottom-up flooding or top-down merging approaches can be used according to the image resolution and data type. Once MTϵ_B is computed, it undergoes a post-processing phase. The goal is to update the attributes of the core nodes consistently with the complete viscous-hyperconnected components. The pseudo-code is detailed in Alg. 1 and Alg. 2 and it has three main steps. At first, for every pixel p in the image, the nodes of MTϵ_B that intersect B_p are determined, where B_p is the structuring element centred on p. Only the nodes at level larger than or equal to $\epsilon_B f(p)$ will be processed. This procedure can be seen as a scanning of the opened image $\delta_B \epsilon_B f$: it represents the set of the dilates by B of the connected sets (cores) in $\epsilon_B f$. Later, the attributes of the nodes are updated. Every node contains a flag `isProcessed` to signal itself as processed after the update of its attributes. In the last step before starting to process the next pixel $p + 1$, all the flags are cleared and set to `false`.

Let us examine more in detail the procedure in Alg. 2. Initially, the node n_p that the current pixel p belongs to is retrieved. This node together with the nodes that correspond to the direct neighbour pixels (e.g. 4-connected or 8-connected) of p at level less than or equal to $\epsilon f(p)$ are marked as processed. Since these pixels are directly connected to p, their contribution has already been counted in the computation of the attributes of n_p in MTϵ_B. For the remaining nodes n lying within B at level larger than or equal to $\epsilon f(p)$, if a node n has not been flagged as processed, the first common ancestor n_{anc} between n and n_p is computed. The `while` loop at line 14 of Alg. 2 implements the attribute updating phase. The node structure is traversed from the node n down to (and not including) the common ancestor. Attributes are updated along the way with the contribution of p and nodes are set as processed. The flag `isProcessed` prevents that a node attribute is updated multiple times with the value of pixel p, when descending from the other nodes lying within B. The descent could stop at the level of pixel p in $\epsilon_B f$ only for the nodes that have n_p in their root path. In fact, a node m different from n_p may exists at the same level of p. Fig. 4 illustrates that. In this case, in MTϵ_B the contribution of pixel p was not counted in the attributes of m and of all the other nodes down to the common ancestor. Once the attributes are computed, the updated max-tree structure is filtered using the attributes of the hyperconnected components. The result is then dilated by B, to have a correct image restitution of the filtered image. Finally, if

The final publication is available at link.springer.com.
DOI: 10.1007/978-3-319-18720-4_56.

Algorithm 1 Process the max-tree of cores of viscous-hyperconnecteced components MTϵ_B.

```
1: procedure PROCESSTREE(Image ε_B f, Nodelist nlist)
2:     for all pixels p ∈ ε_B f do
3:         PutNodesWithinSEIntoNodeList(p, nlist, ε_B f);
4:         ComputeAttributes(p, nlist, ε_B f);
5:         ClearIsProcessedFlag(nlist);
6:     end for
7: end procedure
```

the singletons meet the filtering criterion, they are merged with the result in a manner consistent with the filtering rule used.

4 Implementation notes

4.1 Eroding the input image

The first step of the algorithm is the creation of $\epsilon_B f$, for some structuring element B. The algorithm in [18] was used to compute efficiently erosion operations with arbitrary flat structuring elements: it performs independently of the image content and the number of intensities. Shapes tested in our algorithm are crosses, rectangles and ellipses of different sizes. The same algorithm was used to perform the final dilation to generate the output image.

4.2 Computing the nodes within B_p and the common ancestor

An important issue is to update the list of nodes that intersect B_p in an efficient way. Assume G is the number of levels in the image. We use an array *nodeList* of length G, where each entry contains a linked list of pointers to nodes in the max-tree. The linked lists store the nodes currently within B_p. The max-tree node structure must have two fields: an integer `numPxInNode` that indicates how many pixels currently within B_p belong to the given node and a pointer `listpos` that points to its entry in *nodeList*. Initially, all lists in *nodeList* are empty, all `numPxInNode` fields are set to 0 and all `listpos` pointers are set to `null`. Starting from the first pixel p at the first location, when a node within B_p must be added to *nodeList*, it is checked if its `numPxInNode` field is set to 0. If so, a new list entry is created in *nodeList* at the proper level, and the pointer `listpos` is updated to point to this entry node. If `numPxInNode` is not zero, the node is already in *nodeList* and the counter `numPxInNode` is simply incremented. A similar situation occurs when a node must be removed. The counter `numPxInNode` is decremented. If `numPxInNode` is zero, through the pointer `listpos` the entry in *nodeList* is accessed and removed, while `listpos` is set to `null`. Adding and removing entries is in constant time. Summarising, at each first pixel p for every image scan line, all the nodes within B_p are inserted into *nodeList*. Each time the processing move on to the next pixel, the nodes at the left-hand edge of B_p are removed and those ones at the right edge of B_{p+1} are added. Function `getFirstCommonAncestor()` finds

The final publication is available at link.springer.com.
DOI: 10.1007/978-3-319-18720-4_56.

Algorithm 2 Compute the attributes of the viscous-hyperconnected components.

```
1: procedure COMPUTEATTRIBUTES(Pixel p, NodeList nlist, Image ε_B f)
2:     n_p ← GetTreeNode(p);
3:     n_p.isProcessed ← true;
4:     for all 4-/8-connected neighbour pixels pneigh of pixel p do
5:         if ε_B f(pneigh) ≤ ε_B f(p) then
6:             n_pn ← GetTreeNode(pneigh);
7:             n_pn.isProcessed ← true;
8:         end if
9:     end for
10:    for all nodes n ∈ nlist do
11:        if n != ROOT ∧ n.isProcessed == false ∧ n.Level ≥ ε_B f(p) then
12:            n_anc ← GetFirstCommonAncestor(n, n_p);
13:            n_curr ← n;
14:            while n_curr != ROOT ∧ n_curr.isProcessed == false do
15:                if n_curr.Level > n_anc.Level then
16:                    UpdateAttributes(n_curr, p);
17:                    n_curr.isProcessed ← true;
18:                    n_curr ← n_curr.Parent;
19:                end if
20:            end while
21:        end if
22:    end for
23: end procedure
```

the common ancestor of two nodes examining all the parent pointers starting from them until a common node is reached. Retrieving the ancestor node ought to be implemented more efficiently in constant time after a linear pre-processing of the tree as in [1]. The algorithm has roughly a complexity equal to $O(N_f \cdot G \cdot N_B)$, with N_f and N_B the number of pixels in the image f and in B, respectively.

5 Filtering results and performance

Fig. 1 shows an example of hyperconnected filtering. The structuring element used is a disk of 5 pixel diameter. The filter applied is an area opening, preserving the components with area larger than some threshold value. In Fig. 1b and Fig. 1c, the difference between connected and hyperconnected filtering is evident. Both images were filtered with area threshold larger than 10 pixels. The thin white stripes on the veil of the woman are preserved in Fig. 1c where connected filtering was applied. Even for larger area threshold values (1000 pixels in the case of Fig. 1d), the leakage problem prevents the stripes from being removed. Leakage occurs linking the stripes together to other bright structures in the face or in the background, thus making a single component with large area. With viscous-hyperconnected filtering, an area threshold equal to 10 pixels achieves the result in Fig. 1b. Even smaller area thresholds could have been used because the thin structures have a width of a few pixels and they would be anyway split into singletons by the structuring element used. Leakage through the thin

The final publication is available at link.springer.com.
DOI: 10.1007/978-3-319-18720-4_56.

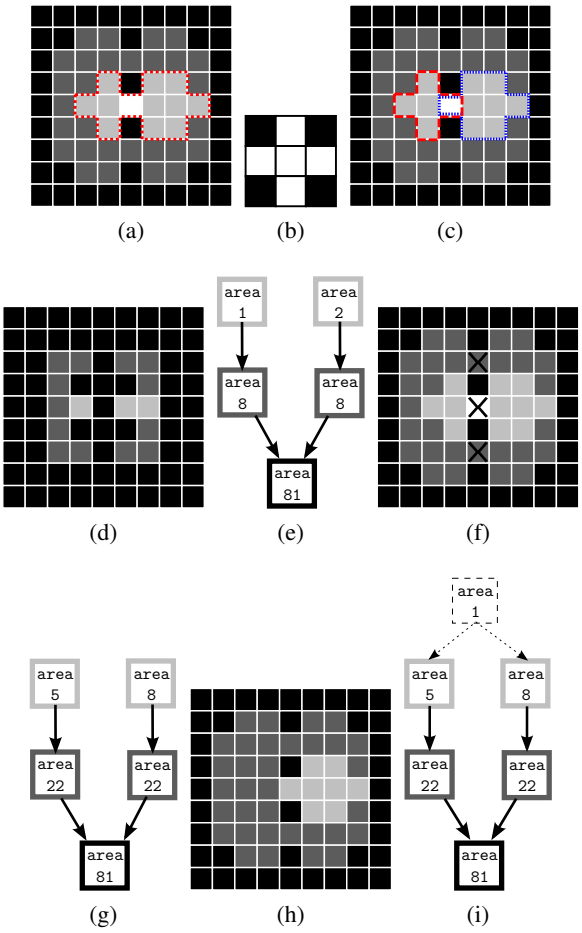
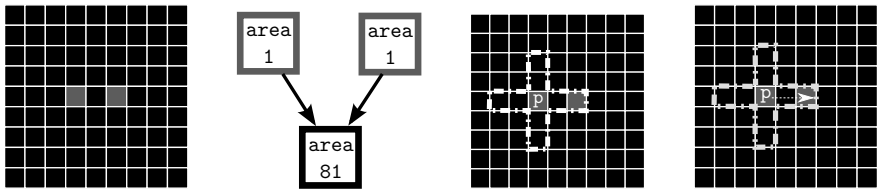


Fig. 3: (a) grayscale image with highlighted one connected component with area=12; (b) cross structuring element; (c) two viscous-hyperconnected components given (b); (d) cores of (a); (e) max-tree of the cores of hyperconnected components in (d); (f) the black crosses indicate the pixels shared by the hyperconnected components; (g) max-tree of hyperconnected components with updated attributes; (h) filtered image: hyperconnected components with area $\geq$ 5 pixels are preserved; (i) directed acyclic graph if the singleton node is added.

white stripes in the veil is therefore stopped. The plot in Fig. 5 shows the wall-clock times needed to perform the hyperconnected filtering for a few images. Area attribute was computed. The algorithm was implemented in C and timings were taken on an Intel Core i7-2670QM@2.20Ghz laptop with second level cache of 6MB. The code is sequential and it runs on a single core. In the plot, the diameter of a disk structuring element varies from 5 to 250 pixels. The images tested are 8-bit images portraying a

The final publication is available at link.springer.com.
DOI: 10.1007/978-3-319-18720-4_56.



(a) (b) (c) (d)

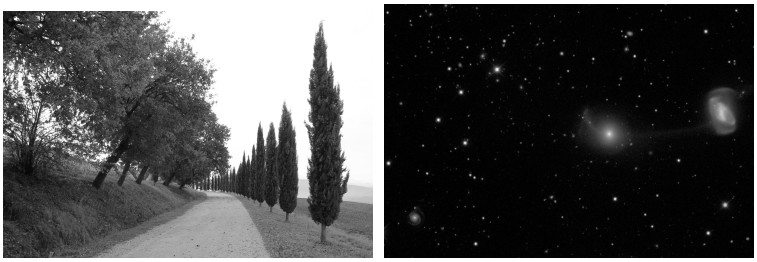
Fig. 4: (a) erosion by a 5x5 cross B; (b) max-tree of (a); (c) B overlapped on the location of pixel p; (d) the node at the dark grey intensity not containing p in (b) must be updated with the contribution of p, in spite of being at the same intensity level.

mountain landscape, a merger galaxy from the Sloan catalogue Data Release 7 and a Tuscan countryside road at three different resolutions. The time to create the eroded image, the max-tree of the cores, and the final dilation is a fraction of the total time, on average five per cent of the total execution time. Erosions and dilations are computed with the algorithm in [18] and MTϵ_B with a flooding approach based on hierarchical queue. Yet with considerably large sizes of the diameter of the structuring element, about 100 pixels, the updated max-tree for the 2 and 3 Megapixels images is created in less than 20 seconds. Much smaller diameters are usually used when processing an image. A test was run also on the original 16-bit image of the galaxy. The time spent in processing the tree with a disk of 5 pixel diameter is long, about 300 seconds. It goes slightly up to 350 seconds with a disk of 250 pixel diameter. Accessing the array *nodeList* dominates the computation time. In case of high bit-depth integer or floating point images, the *nodeList* data structure should not be implemented as an array of length G. It should be possible to avoid any data structure at all, by flagging conveniently the nodes within B_p. This is currently under study.

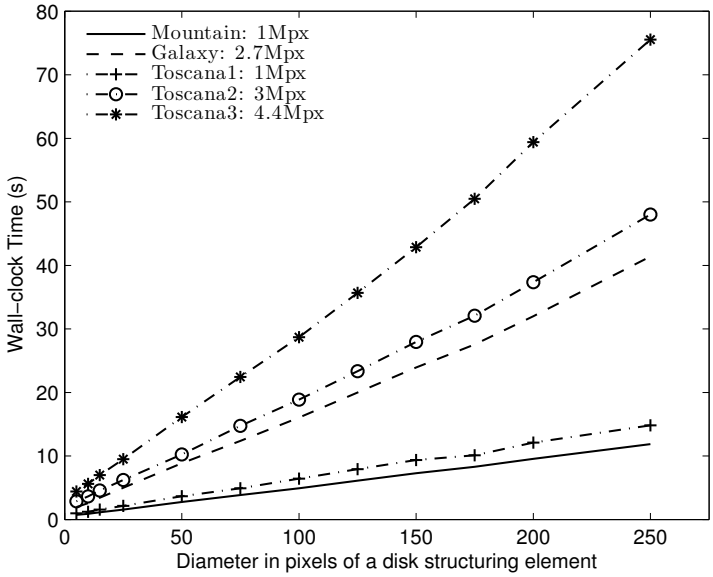
6 Edge preservation issues

A drawback of the proposed algorithm is that the filtered output image is a subset of the opened $\gamma_B f$ image, except when singletons are included in the result. This reduces the edge-preserving feature of connected filtering approaches. The effect is visible in Fig. 6d. The way attributes are computed has to be modified. The attribute accumulation phase should be changed by looking in $\bar{\delta}_f^1 \gamma_B f$ for pixels within a translate of $\delta^1 B$, with δ^1 the unitary dilation and $\bar{\delta}^1$ the geodesic dilation. Furthermore, we need to do both a dilation by B and a geodesic dilation in the last step of the algorithm, before dealing with the singletons. This was proposed in the case of reconstruction in [16, 19]. A complication arises since $\bar{\delta}_f^1 \gamma_B f$ may contain grey levels not present in $\epsilon_B f$, and therefore absent in the max-tree. Nodes ought to be added, and their attributes consistently updated. There is currently ongoing work that will address this issue. A possible solution could be also found in the frame of *direct* connected operators [7], in which the definition of strong connectivity resembles the idea of core of a component presented in Section 3.

The final publication is available at link.springer.com.
DOI: 10.1007/978-3-319-18720-4_56.



(a) (b)



(c)

Fig. 5: (a) and (b) show the images referred to as Toscana and Galaxy in the plot in (c). The Toscana image was down-sampled at three different resolutions. The plot shows the time spent to update the max-tree of cores.

7 Conclusions

In this paper we used a hyperconnectivity class that addresses the leakage problem, as an alternative of working on a viscous lattice. A novel algorithm to perform hyperconnected attribute filtering on the viscous-hyperconnected components belonging to this class is proposed. First, the max-tree of the image eroded by a structuring element B is built. It contains the cores of the viscous-hyperconnected components. Their attributes are updated at a later stage with the pixels that lie between the core components and

The final publication is available at link.springer.com.
DOI: 10.1007/978-3-319-18720-4_56.

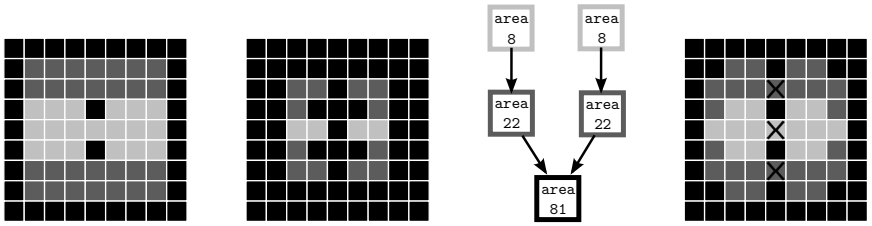


Fig. 6: (b) cores of (a) after erosion with a 3x3 cross structuring element; (c) max-tree of hyperconnected components; (d) image representation of the tree in (c): edges differs from the ones in (a).

their dilate, through accessing the nodes within B_p for every image pixel p. Any existing algorithm can be used to build the max-tree of the eroded image. The proposed filtering strategy can also be used with high bit-depth integer or floating point images. Performance tests showed the importance of efficiently retrieving the nodes within B, at every location in the image. The filtered image is dilated by B and singletons are merged, according to the filtering rule used. A rigorous definition of directed acyclic graph that originates when singletons are added must be given. A drawback of this approach is that the filters do not preserve the edges. A possible solution has been found and it will be subject for a further work in the near future.

References

1. Bender, M.A., Farach-Colton, M.: The lca problem revisited. In: Proceedings of the 4th Latin American Symposium on Theoretical Informatics. pp. 88–94. LATIN '00, Springer-Verlag, London, UK, UK (2000)
2. Braga-Neto, U., Goutsias, J.: A theoretical tour of connectivity in image processing and analysis. J. Math. Imaging Vis. 19(1), 5–31 (Jul 2003), http://dx.doi.org/10.1023/A:1024476403183
3. Breen, E.J., Jones, R.: Attribute openings, thinnings and granulometries. Comp. Vis. Image Understand. 64(3), 377–389 (1996)
4. Ouzounis, G.K., Wilkinson, M.H.F.: Mask-based second generation connectivity and attribute filters. Pattern Analysis and Machine Intelligence, IEEE Transactions on 29(6), 990–1004 (2007)
5. Ouzounis, G.K., Wilkinson, M.H.F.: Hyperconnected attribute filters based on k-flat zones. Pattern Analysis and Machine Intelligence, IEEE Transactions on 33(2), 224–239 (2011)
6. Perret, B.: Inf-structuring functions: A unifying theory of connections and connected operators. J. Math. Imaging Vis. 51(1), 171–194 (2015)
7. Perret, B., Cousty, J., Tankyevych, O., Talbot, H., Passat, N.: Directed connected operators: Asymmetric hierarchies for image filtering and segmentation. Pattern Analysis and Machine Intelligence, IEEE Transactions on (2014)
8. Perret, B., Lefevre, S., Collet, C., Slezak, E.: Hyperconnections and hierarchical representations for grayscale and multiband image processing. Image Processing, IEEE Transactions on 21(1), 14–27 (Jan 2012)

The final publication is available at link.springer.com.
DOI: 10.1007/978-3-319-18720-4_56.

9. Salembier, P., Oliveras, A., Garrido, L.: Anti-extensive connected operators for image and sequence processing. Image Processing, IEEE Transactions on 7, 555–570 (1998)
10. Salembier, P., Serra, J.: Flat zones filtering, connected operators, and filters by reconstruction. Image Processing, IEEE Transactions on 4, 1153–1160 (1995)
11. Santillán, I., Herrera-Navarro, A.M., Mendiola-Santibáñez, J.D., Terol-Villalobos, I.R.: Morphological connected filtering on viscous lattices. J. Math. Imaging Vis. 36(3), 254–269 (Mar 2010), http://dx.doi.org/10.1007/s10851-009-0184-8
12. Serra, J.: Image Analysis and Mathematical Morphology. II: Theoretical Advances. Academic Press, London (1988)
13. Serra, J.: Viscous lattices. In: Proc. Int. Symp. Math. Morphology (ISMM) 2002. pp. 79–90 (2002)
14. Serra, J.: Connectivity on complete lattices. J. Math. Imag. Vis. 9(3), 231–251 (Nov 1998), http://dx.doi.org/10.1023/A:1008324520475
15. Sofou, A., Tzafestas, C., Maragos, P.: Segmentation of soilsection images using connected operators. In: Int. Conf. Image Proc. 2001. pp. 1087–1090 (2001)
16. Terol-Villalobos, I.R., Vargas-Vzquez, D.: Openings and closings with reconstruction criteria: a study of a class of lower and upper levelings. J. Electronic Imaging 14(1), 013006 (2005), http://dblp.uni-trier.de/db/journals/jei/jei14.html#Terol-VillalobosV05
17. Tzafestas, C.S., Maragos, P.: Shape connectivity: Multiscale analysis and application to generalized granulometries. J. Math. Imag. Vis. 17, 109–129 (2002)
18. Urbach, E.R., Wilkinson, M.H.F.: Efficient 2-D gray-scale morphological transformations with arbitrary flat structuring elements. Image Processing, IEEE Transactions on 17, 1–8 (2008)
19. Wilkinson, M.H.F.: Connected filtering by reconstruction: Basis and new advances. In: 15th IEEE International Conference on Image Processing. ICIP 2008. pp. 2180–2183 (Oct 2008)
20. Wilkinson, M.H.F.: Attribute-space connectivity and connected filters. Image Vision Comput. 25(4), 426–435 (Apr 2007), http://dx.doi.org/10.1016/j.imavis.2006.04.015
21. Wilkinson, M.H.F.: An axiomatic approach to hyperconnectivity. In: Wilkinson, M.H.F., Roerdink, J.B.T.M. (eds.) Mathematical Morphology and Its Application to Signal and Image Processing, Lecture Notes in Computer Science, vol. 5720, pp. 35–46. Springer Berlin Heidelberg (2009), http://dx.doi.org/10.1007/978-3-642-03613-2_4
22. Wilkinson, M.H.F.: Hyperconnectivity, attribute-space connectivity and path openings: Theoretical relationships. In: Wilkinson, M.H.F., Roerdink, J.B.T.M. (eds.) Mathematical Morphology and Its Application to Signal and Image Processing, Lecture Notes in Computer Science, vol. 5720, pp. 47–58. Springer Berlin Heidelberg (2009)
23. Wilkinson, M.H.F.: Hyperconnections and openings on complete lattices. In: Soille, P., Pesaresi, M., Ouzounis, G.K. (eds.) Mathematical Morphology and Its Applications to Image and Signal Processing, Lecture Notes in Computer Science, vol. 6671, pp. 73–84. Springer Berlin Heidelberg (2011), http://dx.doi.org/10.1007/978-3-642-21569-8_7

The final publication is available at link.springer.com.
DOI: 10.1007/978-3-319-18720-4_56.