Thin structures retrieval using anisotropic neighborhoods of superpixels: Application to Shape-From-Focus

Christophe Ribal, Sylvie Le Hégarat-Mascle, Nicolas Lermé

Journal of Visual Communication and Image Representation (2022)

Contents lists available at ScienceDirect

Journal of Visual Communication and Image Representation

journal homepage: www.elsevier.com/locate/jvci

Thin structures retrieval using anisotropic neighborhoods of superpixels: Application to Shape-From-Focus

Christophe Ribal[a], Sylvie Le Hégarat-Mascle[a], Nicolas Lermé[a,*]

[a] *Université Paris-Saclay, SATIE Laboratory UMR 8029, Avenue des sciences, 91190 Gif-sur-Yvette, France*

ARTICLE INFO

Article history:
Received -
Received in final form -
Accepted -
Available online -

Communicated by -

Keywords: Shape-From-Focus, Thin structures regularization, Anisotropic neighborhood, Superpixel, Tensor voting, RORPO

ABSTRACT

Shape-From-Focus (SFF) refers to the challenging inverse problem of recovering the scene depth from a given set of focused images using a static camera. Standard approaches model the interactions between neighboring pixels to get a regularized solution. Nevertheless, isotropic regularization is known to introduce undesired artifacts and to remove early thin structures. These structures have a small size in at least one dimension and are more numerous when considering superpixel preprocessing. This paper addresses the improvement of SFF regularization through the estimation of the presence of such structures and the construction of anisotropic neighborhoods sticking along image edges and proposes a flexible formulation over pixels or superpixels. A thoroughly study comparing different strategies for constructing these neighborhoods in terms of accuracy and running time for the targeted application is provided. Notably, experiments performed on a reference dataset show the overall superiority of the approach and its robustness against generated superpixels.

1. Introduction

For many image processing problems such as image segmentation or reconstruction, low-level information delivered by a single pixel is limited and prone to noise, corrupted data and all kinds of optic phenomena altering the original image. Therefore, taking into account a statistical relationship between spatially close pixels has been introduced relatively early in image processing [1]. A classical way to handle this is to model the two-dimensional (2D) field of pixels as a Markov Random Field (MRF). This allows for introducing a prior on the expected solution. Variational approaches are particularly used to provide solutions, by combining the prior and conditional probabilistic models into a single parametric functional to be minimized. However, due to the dimensionality of the solution space and depending on the form of the functional, finding a global minimizer of it often appears as a challenging task. The study [2] gives an insight by comparing several minimization algorithms (including graph cuts) on typical vision problems (including image segmentation and image reconstruction). It is well established and documented that standard Total Variation (TV) regularization (e.g. in image reconstruction [3]) or Potts regularization (e.g. in image segmentation [4]) using isotropic neighborhoods behave poorly on thin structures. Although they are ubiquitous in a number of applications, their detection remains very difficult because of their spatial sparsity, their small size and their potential complex geometry. Since these structures essentially consist of discontinuities, standard TV and Potts regularization tend to early remove them as regularization increases and are thus not adapted to handle them correctly [5].

In parallel with algorithmic developments, the volume and the diversity of data to exploit have greatly increased over the last years, therefore preprocessing have been proposed to reduce the computational burden. For instance, superpixel decomposition methods [6] have been developed for grouping pixels sharing similar radiometric intensities into homogeneous

*Corresponding author. E-mail: nicolas.lerme@universite-paris-saclay.fr

regions, and then drastically reducing the number of elements to process while preserving the geometrical information that is lost with multi-resolution approaches. For instance and specifically for segmentation problem, [7, 8] grow and merge regions from an initial set of superpixels that they call an oversegmentation of the initial image. A major drawback of a superpixel segmentation is that the usual hypothesis of a regular topological lattice is lost, as well as the regularity in size and shape of every lattice element. As a result, image segmentation approaches taking advantage of superpixels must cope with these problems and introduce new methods and spatial relationships. Often, superpixels are considered as neighbors when sharing a common border [9, 10, 11, 12]. The authors of [9] propose to minimize an energy via graph cuts based on the adjacency graph obtained from the watershed segmentation, where edges connecting two regions are weighted upon the common border length between these regions. Similarly, [12] propose to ease the classification of the high-dimensional noisy hyperspectral images by building a weighted graph based on superpixels. In [13], the authors compute saliency from MRF using the same concept of adjacency, and take into account in their algorithm the second-order neighborhood to ease the propagation of information between superpixels. Other superpixel approaches use patches to analyze the spatial content over a neighboring window and find the nearest matches in a set of reference patches [14]. In [15], the authors train a deep Hough forest from a set of superpixel patches in order to detect objects in aerial images.

Although our work on anisotropic neighborhood can be applied to several segmentation or reconstruction problems, we focus on the application Shape From Focus (SFF) in this paper. SSF is a popular method used for inferring the 3D shape of an object from a set of images with varying focus settings [16]. Such an approach only requires one fixed camera with a rather short depth of field and is able to move this camera or to change the focal distance of the optical system. SSF is therefore applicable in many real world applications including industrial inspection, micro manufacturing, robotic control, 3D model reconstruction, medical imaging systems and microscopy. In addition to its intrinsic interest, we focus on this application to illustrate the benefit of anisotropic neighborhoods since naive pixel-level estimates are hampered by the presence of homogeneous surfaces, thus definitively requiring some regularization to propagate the information from reliable areas to uncertain ones, while preserving thin structures. However, the regularization is all the more challenging that the number of labels (i.e., the number of discrete depth values) is important and that structures (and input data) are 3D.

Our first contribution is to propose different estimations of anisotropic neighborhoods on an irregular lattice such as the ones provided by superpixel segmentation. Our second contribution is to propose SSF based on superpixels. It allows us to illustrate the benefit of anisotropic neighborhood since SSF represents a sufficiently complex application so that results may depend on the type of considered neighborhood.

The rest of this paper is organized as follows. In Section 2, we specify the considered problem, namely SSF using superpixel segmentation. In Section 3, we detail the proposed path-based constructions of anisotropic neighborhoods, based on a preliminary estimation of local anisotropy and orientation either from Tensor Voting [17], or from RORPO [18]. Section 4 discusses the results and benefits of our approach in a comprehensive comparative study between isotropic and anisotropic neighborhoods both in terms of accuracy and time complexity. Finally, Section 5 draws main conclusions and perspectives.

2. Superpixels-based SFF

2.1. Basics of SSF

The core idea of SSF is that the closer an object is to the object focal plane (i.e., the more it is focused), the more it appears sharp. Conversely, the farther an object is from this object focal plane, the more it appears blurred. Therefore, SSF relies on a sharpness operator to find the depth where each point appears the more sharp, and reconstructs a depth image: In the absence of regularization (*blind* estimation), the depth of each pixel of the 2D scene maximizes the pixel's sharpness measure. Specifically, [16] approximates the sharpness curve (that represents the sharpness values versus the focus parameter values) with a Gaussian model, and interpolates (along the optical axis) the three focus measures centered on the maximum sharpness value to allow for a better depth estimation. To reduce the sensitivity to noise, some authors do the sharpness curve interpolation by using quadratic, cubic or polynomial interpolation [19] or Gaussian interpolation [3].

Nevertheless, *blind* depth estimation remains prone to noise and ambiguities since, in homogeneous or poorly textured areas, the measured sharpness will be quite low and unreliable. To overcome this limitation, some authors [20, 19, 3] have considered SFF in the variational framework. With regularization, the information extracted in the scene areas where SFF is reliable, such as in objects details, contours, is propagated from neighbors to ambiguous areas, such as homogeneous, overexposed or underexposed regions. As stated in Section 1, to overcome the alteration of thin structures, there is however a need for anisotropic regularization, all the more critical that we work with superpixels.

2.2. From pixel level to superpixel one

Let us consider the 3D space defined by an orthonormal basis $(\mathbf{e}_0, \mathbf{e}_1, \mathbf{e}_2)$ such that $\mathbf{e}_0$ and $\mathbf{e}_1$ are aligned with the image row and column dimensions and $\mathbf{e}_2$ will represent the focus dimension. The set of input image pixels, denoted $\mathcal{P}$, defines a cube in $(\mathbf{e}_0, \mathbf{e}_1, \mathbf{e}_2)$ having dimensions $n_{\text{row}} \times n_{\text{col}} \times n_{\text{foc}}$, where n_{row}, n_{col}, and n_{foc} are positive integers. We also assume without loss of generality that these three dimensions are sampled with a unit step even if for focus it implies a simple transformation.

Then, we assume that superpixels are invariant with focus dimension and we thus define them equivalently in 2D ($(\mathbf{e}_0, \mathbf{e}_1)$ plane) or 3D by replicating them along the focus dimension ($\mathbf{e}_2$). In the following, we denote by $\mathcal{S}$ the set of superpixels defined in 2D and by $\mathcal{S}^{\uparrow 3}$ the set of superpixels extended to 3D.

In this study, we aim at extending SSF variational formulation to superpixel level. However, the sharpness profiles and the superpixels themselves have to be preliminary computed at

pixel level. Specifically, any sharpness value shall be computed at pixel level by nature. A sharpness operator is a function computed at every pixel $p \in \mathcal{P}$ and a sharpness profile is a vector gathering the sharpness values obtained by varying the focus dimension for a given pair of (row, column) coordinates in $(\mathbf{e}_0, \mathbf{e}_1, \mathbf{e}_2)$. In the following, we denote by $f(p)$ the sharpness operator defined in pixel $p \in \mathcal{P}$ and by $.\downarrow$ the projection on $(\mathbf{e}_0, \mathbf{e}_1)$ such that $\mathcal{P}\downarrow$ and $p\downarrow$ are the projections of $\mathcal{P}$ and p respectively, and $\mathbf{f}(p\downarrow)$ denotes the sharpness profile at any pixel $p\downarrow \in \mathcal{P}\downarrow$. Then, the maximum of sharpness is estimated at any $p\downarrow \in \mathcal{P}\downarrow$ as $\max_{k\in[\![1,n_{\text{foc}}]\!]} \mathbf{f}_k(p\downarrow)$ where $\mathbf{f}_k$ denote the k^{th} component of sharpness profile $\mathbf{f}$. From maximum of sharpness, one can estimate the all-in-focus image defined on $\mathcal{P}\downarrow$. This image allows us to compute the superpixels, $\mathcal{S}$, that have a good sensitivity to the sharp edges of the scene, since in 2D space it picks the pixel that is the "sharpest", i.e. that has the highest contrast with its neighbors. The chance of constructing superpixels on blurred edges of the objects is minimized this way. From $\mathcal{S}$, the set of superpixels extended to 3D $\mathcal{S}^{\uparrow 3}$ is then derived by duplicating n_{foc} times any superpixel $s \in \mathcal{S}$ along the axis $\mathbf{e}_3$.

The sharpness values of $\mathcal{S}^{\uparrow 3}$ elements can then be derived from the mean sharpness values of the 3D pixels $p \in \mathcal{P}$ that compose it, and, for each superpixel in $\mathcal{S}$, a *blind* depth estimation is derived from the maximum of sharpness varying focus ($\mathcal{S}^{\uparrow 3}$ elements derived from a given superpixel $s \in \mathcal{S}$). In the following, the *blind* depth superpixel map is denoted $\hat{\mathbf{u}} = (\hat{u}_s)_{s\in\mathcal{S}}$ with $\hat{u}_s \in \mathbb{N}$ assuming (without loss of generality) that depth values are sampled as integer numbers. This depth map may be noisy and sensitive to the low sharpness profile of homogeneous regions of the scene, which we cope with our anisotropic neighborhood based regularization.

Finally, let us specify that, in our case, we consider the sharpness operator introduced in [21], namely the Summed Modified LAPlacian (SMLAP): $f(p) = \text{SMLAP}(p), \forall p \in \mathcal{P}$.

2.3. Energetic formulation

Usual energetic formulations map a realisation of the random field we search, i.e. $\mathbf{u}$ in SFF application, to a real number representing its inadequacy to correspond to the observations and prior knowledge. In our case, since the neighborhoods are also unknown, we made the energy depend also on a neighborhood field that maps a local anisotropic neighborhood to any field element (superpixels in our case). In the following, $\mathbf{u} \in \mathbb{N}^S$ is the researched depth field, V is the neighborhood field and $\mathbb{V}$ is the set of possible neighborhood fields. Then, we aim at finding a minimizer of

$$F(\mathbf{u}, V) = E_1(\mathbf{u}) + \alpha E_2(\mathbf{u}, V), \tag{1}$$

where $\alpha \in \mathbb{R}_{\geq 0}$ is an hyperparameter that need to be later tuned by the user. Specifically, the data fidelity term $E_1(\mathbf{u})$ is instantiated with a quadratic distance to the *blind* estimate $\hat{u}_s$:

$$E_1(\mathbf{u}) = \sum_{s\in\mathcal{S}} W_s (u_s - \hat{u}_s)^2, \tag{2}$$

where W_s depends on the dynamics of the sharpness profile normalized by its averaged value:

$$W_s \propto \left(\frac{\max\limits_{k\in[\![1,n_{\text{foc}}]\!]}(\mathbf{f}_k(s)) - \min\limits_{k\in[\![1,n_{\text{foc}}]\!]}(\mathbf{f}_k(s))}{\frac{1}{n_{\text{foc}}}\left(\sum\limits_{k\in[\![1,n_{\text{foc}}]\!]} \mathbf{f}_k(s)\right) - \min\limits_{k\in[\![1,n_{\text{foc}}]\!]}(\mathbf{f}_k(s)) + \epsilon} \right), \tag{3}$$

with $\epsilon \in \mathbb{R}_{>0}$ a small positive real number. With the weighting term W_s, the importance of the data fidelity term E_1 is decreased when the sharpness profile is homogeneous or when it presents a very low dynamic. Conversely, the areas with a sharpness profile with a precisely localized high response have high values of W_s reflecting the belief that they are trustful.

The regularization term $E_2(\mathbf{u}, V)$ is derived from the TV operator. For any $\mathbf{u} \in \mathbb{N}^S$ and any $V \in \mathbb{V}$, it is defined as

$$E_2(\mathbf{u}, V) = \sum_{s\in\mathcal{S}} \sum_{t\in V(s)} W_{st} |u_s - u_t|, \tag{4}$$

where W_{st} is a weighting function depending on the neighborhood field V:

$$W_{st} = \frac{1}{2}\left(\frac{1}{\sharp V(s)} + \frac{1}{\sharp V(t)}\right), \tag{5}$$

where $\sharp V(s)$ denotes the cardinality of the neighborhood at the superpixel s. The weighting term W_{st} aims at normalizing the regularization terms E_2 with respect to the size of the considered neighborhoods since this latter is no longer constant (as it was with usual 4 or 8-connectivity for instance at pixel level).

2.4. Optimization

Graph cuts optimization refers to the computation of minimum cuts/maximum-flows in a graph of appropriate topology for minimizing functionals arising in computer vision, e.g. composed of unary and pairwise terms. Compared to other combinatorial algorithms, graph cuts are very competitive both in terms of accuracy (global minimum is very well approached if not reached as for many binary problems) and running time (by avoiding stochastic iterative convergence) for a wide range of computer vision tasks [2]. Practically, graph cuts depict linear complexity in the number of sites of $\mathcal{S}$ [22]. Moreover, compared to continuous minimization algorithms, they are able to deal with regular or irregular lattices without any difficulties.

In the binary case, the key idea of graph cuts is to construct a two-terminals graph, where nodes are sites of $\mathcal{S}$ and edges encode relationships between nodes, in a such a way that any cut separating these terminals is equal to the value of the functional on the underlying binary labeling. In particular, when all pairwise terms are submodular, polynomial-time maximum-flow algorithms allow for efficiently finding the minimum-cut in a graph and thus a global minimizer of the functional for binary problems [22].

In the multi-labels case (such as in our case), efficient algorithmic schemes exist for finding minimizers of functionals. As explained in Section 3, we intend as a first attempt in this paper to minimize F (see Equation (1)) with V fixed. It is not difficult to see that the functional $u \mapsto F(u, V)$ is convex based on Equation (2), (3), (4) and (5). In such a situation, a global

minimizer of this functional can be be efficiently obtained by decomposing the problem into a set of subproblems only involving binary variables (as in the case of isotropic neighborhoods in [3]), where each one of them is solved standard graph cuts in the aforementioned binary case.

3. Anisotropic neighborhood construction

The construction of anisotropic neighborhoods can be decomposed into (i) the estimation of the presence of thin structures (in our case performed by a vesselness operator) discussed in Sections 3.1 and 3.2, and (ii) the actual computation of the neighbors of each superpixel discussed in Section 3.3. Let us recall that the neighborhoods are constructed on an irregular lattice of superpixels, and that a neighborhood relationship can be formalized on a graph representing the superpixels by vertices, by the edges interconnecting some vertices. The neighborhood is thus an application that maps the set of superpixels $\mathcal{S}$ to its powerset $2^{\mathcal{S}}$ without any specific constraint (e.g., bound on spatial distance) at this stage. We want to outline that it may thus be different to the notion of *adjacency* that refers to the existence of a common border between the superpixels and that allows for the definition of connected components.

Then, to estimate anisotropic neighborhoods, we will rely on a guidance map, denoted $\mathbf{g}$, that encodes the information of anisotropy and orientation for every superpixel $s \in \mathcal{S}$. Such a map must encourage the alignment of neighborhoods with the thin structures of the image. In the absence of knowledge of the scene objects, the estimation of $\mathbf{g}$ is not trivial at all. Indeed, considering simultaneous estimation of $\mathbf{g}$ and the segmentation or reconstruction, the resolution appears very complex if not intractable, and considering alternate estimation would require an iterative scheme ensuring the convergence in a controlled number of iterations. Therefore in this study, we rather focus on a single estimation of $\mathbf{g}$ as a first attempt, with obvious methodological and computational benefit, at the expense of defining an estimation sufficiently robust to the input data imperfections to yield some trustworthy guidance map. Simultaneous estimation of $\mathbf{u}$ and V is left for future work. More specifically, if the estimation of $\mathbf{g}$ bases on local estimates, its construction must be robust to noise in these latter. We investigate two options, the *Tensor Voting* (TVo) as presented by [17] and the *Ranking the Orientation Responses of Path Operators* (RORPO) vesselness operator as introduced by [18]. In what follows, $\mathbf{g}$ is a field of $\mathbb{R}^2$ vectors encoding both the direction and the saliency.

3.1. Tensor Voting-based guidance map

3.1.1. Tensor Voting basics

Tensor Voting (TVo) has been selected for its robustness to noise and efficiency for connecting thin structures like edges [17]. TVo relies on the Gestalt principles of perceptual organization (such as proximity, continuity and similarity) for designing the voting operation. Its formulation involves one scale parameter, $\sigma_T \in \mathbb{R}_{>0}$, setting the spatial range in which most of the energy of the TVo will be distributed. The basic idea is that casting a vote to other site locations allows the information of each tensor to be propagated, and then thanks to the voting step, the tensors are smoothed and their orientations refined. Voting operation is performed through voting kernels that have continuous and smoothly varying orientations of eigenvectors and decreasing eigenvalues, except at the origin of the kernel. For implementation purpose, the voting kernels are often discretized and stored into a precomputed field of tensors, which evaluates the values of the tensors cast from the voter on each point of a regular lattice. Appendix A specifies and gathers all the main equations useful for 3D TVo, that is much more complex than 2D one used in [23] for instance.

In [17], TVo involves the five following main steps. First step is the initial vote that requires the definition of the initial set of voters also called *tokens* and the set of cast locations (for vote). In the absence of orientation information, the set of tokens is usually converted into a sparse set of ball tensors that vote in every image site. Second step is a refinement step. Based on the previous sparse vote, the initial set of ball tensors can be refined into a set of stick tensors. For this, each tensor is projected on the stick tensor axis in the basis used for tensor decomposition. Third step is a dense voting in order to propagate the stick information at every point. It yields the dense tensor map. Then, fourth step projects the tensors on the three axes of the decomposition basis so that three saliency maps can be derived, encoding for surface, curve and junction saliency. From these maps, the final step of the algorithm derives the probabilities of presence of surfaces, curves and points.

3.1.2. Computation of guidance map

We adapt TVo to our SSF problem as follows. The *tokens* are the local maxima of sharpness profiles in every superpixel (in $(\mathbf{e}_0, \mathbf{e}_1)$ plane). To avoid redundancy between close maxima (inducing artificial reinforcement of these latter) of a same profile, a non-maximum suppression step is performed on sharpness profiles: Specifically, we only keep one maximum per continuous interval of focus values associated to sharpness values greater than 80% of the maximum sharpness. This way, we ensure that the *tokens* are all separated by a local minimum having value below 80% of the global maximum. This initialization provides a tensor map that is sparse in 3D, but dense in 2D.

Then, since the number of pairs in $(\mathcal{S}^{\uparrow 3} \times \mathcal{S}^{\uparrow 3})$ is very large, the vote for the orientation estimation is also restricted to the set of *tokens*. This allows for reducing the computational burden by removing the dense voting step, at the risk of a loss of accuracy when the initial depth estimations (and thus *tokens*) are erroneous.

Although TVo allows us to handle tensors defined in $\mathbb{R}^3$ for the vote, at the end (for decision after voting) we have to decide a single tensor for any 2D superpixel $s \in \mathcal{S}$. In our experiments, we found that the most convincing results are obtained when only considering the cumulated tensors (after voting) at the *blind* estimated depth $\hat{u}_s$. Indeed, while this leads to irregularities when $\hat{\mathbf{u}}$ is noisy, this also allows for gaps in the orientations estimated on the edges of the structures of the images, which could be beneficial. Then, for extracting the guidance map $\mathbf{g}$, for each superpixel $s \in \mathcal{S}$, we project the selected tensor (in $\hat{u}_s$) into image plane and derive the major eigenvector $\hat{\mathbf{e}}_{0s}$ in s and the two eigenvectors $(\lambda_{0s}, \lambda_{1s}) \in \mathbb{R}^2_{\geq 0}$ so that the saliency and orientation of the guidance map in s is computed

as follows:

$$\mathbf{g}_s = (\lambda_{0s} - \lambda_{1s})\hat{\mathbf{e}}_{0s}, \quad \forall s \in \mathcal{S}.$$

3.2. RORPO-based guidance map

As an alternative to TVo, we consider RORPO, a non linear operator based on mathematical morphology and used for thin structure detection (see [18] for more details).

3.2.1. RORPO basics

The idea of RORPO is to use a set of oriented filters with different orientations to analyze the image in terms of the response of multiple morphological operations. Indeed, for a thin structure, at least one dimension is substantially smaller than the other ones by definition. Thus, determining the image areas where only a small number of high responses are measured among the oriented filters discriminates the thin structures. These oriented filters, called path openings, are parameterized by structuring functions defining the set of connection relationships $\mathcal{R}$ between sites (pixels, or superpixels in our case). Since the RORPO is based on these path openings, let us briefly recall how these latter work, firstly on a binary image and secondly on a gray level image.

Denoting by $\mathcal{R}$ the set of connection relationships, for each connection relationship $\rightsquigarrow_\theta \in \mathcal{R}$, we remind how a path opening is defined for binary images in [18]: Given $\rightsquigarrow_\theta$ and a length $L \in \mathbb{R}_{>0}$, the path opening $\mathbb{O}_{\rightsquigarrow_\theta,L}(X)$ is the union of all paths connected by $\rightsquigarrow_\theta$ and of length L in the set of *true* pixels (1-valued) in the considered binary image X. Each path opening filters out the structures that are not aligned with the considered orientation. Thus, a thin structure will be deleted by at least one oriented filter, conversely to isotropic structures that will have an homogeneous answer to the set of path openings.

Then, to extend binary path openings to gray level images, one considers level sets, i.e. sets of sites having a value greater than the considered gray level: Given $\rightsquigarrow_\theta$ and a length $L \in \mathbb{R}_{>0}$, the gray level path opening of an image Y is defined as

$$\mathbb{O}_{\rightsquigarrow_\theta,L}(Y, s) = \max\left\{\tau \in \mathbb{R}_{>0} | s \in \mathbb{O}_{\rightsquigarrow_\theta,L}(Y_{\geq\tau})\right\},$$

where $Y_{\geq\tau}$ is the level set of Y at level τ.

In [18], RORPO implementation involves the following five main steps. The first step is the dilation of the gray level input image with respect to spatial adjacency. The second step deals with direction sampling. It boils down defining a finite set of connection relationships, denoted by $\rightsquigarrow_\theta$, such that two sites s and t are connected if and only if (i) they are adjacent and (ii) $\overrightarrow{st}$ vector's direction and the sampled direction θ are considered equal been given the imprecision angle ϕ_T threshold. The third step is the computation of the path opening results for the sampled directions and the fourth step ranks their responses as follows: For each site, the responses to the $\#\mathcal{R}$ path openings are ranked in decreasing order of magnitude, i.e. denoting RF_1 the maximum value and $RF_{\#\mathcal{R}}$ the minimum (last in the ordering) value. This ranking of the orientation responses of the path openings gave its name to the algorithm RORPO. Then, for each site, the RORPO value is the difference between maximum path opening value (RF_1) and the i largest response, (RF_i). In our case, we set $i = 4$. Finally, fifth step derives, for each site, an orientation by averaging the orientations of the three largest responses.

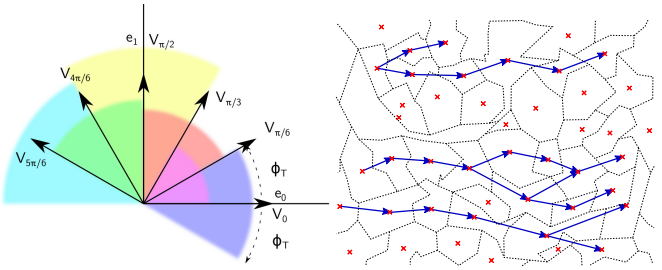


Fig. 1: Illustration of the 6 directions of $\mathcal{R}$ (left) and an example of path obtained with one structuring function $\rightsquigarrow_\theta$ (right). The connectedness $\rightsquigarrow_\theta$ is characterized by the vector $\mathbf{v}_\theta$ and the angular width ϕ_T. For this illustration, we have represented directed edges for positive displacements, but the paths are computed in both directions.

This formulation yields higher responses for thin structures that have a small number of high responses in path openings. Therefore, the value returned by the RORPO allows us to discriminate the saliency of thin structures. Let us now present the estimation of orientations used to derive the guidance map $\mathbf{g}$.

3.2.2. Computation of guidance map

Like TVo, our implementation of RORPO works with the data volume corresponding to the sharpness profiles in every superpixel. The choice of these input data is fully relevant for the RORPO that will detect structures presenting high gray level values and indeed we want to detect highest sharpness values.

For numerical convenience, path openings are only performed with 2D slices, i.e. at given focus value, which boils down researching structures in image plane. This is simply performed by restricting the connection relationships $\rightsquigarrow_\theta$ to be within image plane. Then, we consider six directions $\mathbf{v}_\theta$ in the image plane, characterized by their positive angle θ with the $\mathbf{e}_0$ axis: $\mathbf{v}_\theta = \cos(\theta)\mathbf{e}_0 + \sin(\theta)\mathbf{e}_1$ (see Figure 1). Note that, although usually the length L is a positive integer expressed in pixel unit, extending the case of pixel lattice to superpixel one, we instead consider that the length of the path is a real $L \in \mathbb{R}_{>0}$, computed as the sum of the distances between the superpixels' barycenters in the path.

Each of the connection relationships yields a path opening result. From this set of path openings, we firstly compute the RORPO index that is further interpreted as a saliency index and secondly the structure orientation. For the latter, we use a specific average operation such that orthogonal vectors cancel and vectors of opposite directions would not. The trick consists in considering polar coordinates and doubling the argument value of the vectors before averaging them, and dividing the argument of the averaged result by two. Mathematically, with complex notations, and omitting the normalization coefficient useless here, it is as follows:

$$\mathbf{v}_{RO}(s) \propto \left(\epsilon + \sum_{\rightsquigarrow_\theta \in \mathcal{R}'(s)} \mathbb{O}_{\rightsquigarrow_\theta,L}(Y, s)\exp(2i\theta)\right)^{\frac{1}{2}},$$

where $\epsilon > 0$ is a very small real number used for numerical stability of the expression, and $\mathcal{R}'(s) \subset \mathcal{R}$ is the set of orientations

of the three first answers in the rank filter at site s.

With this construction, we obtain a 3D volume of vectors that has the same dimensions as the grayscale input data and from which the 2D guidance map $\mathbf{g}$ is finally computed. For this, we average the $\mathbf{v}_{RO}$ directions varying the depth:

$$\mathbf{g}_s = \left(\sum_{t \in \mathcal{S}^{\uparrow 3}, t_\downarrow = s} |\mathbf{g}_t| \exp(2i \arg(\mathbf{g}_t)) \right)^{\frac{1}{2}}, \quad \forall s \in \mathcal{S},$$

where $t_\downarrow$ is the result of the projection of 3D site t on the 2D image plane and $\arg(c)$ denotes the argument of any complex number $c \in \mathbb{C}$. In previous equation, the angles are simply weighted by the norm $\mathbf{g}_t$, but more sophisticated weighting may also consider the distance with respect to the *blind* depth, e.g. using a weighting coefficient equal to $|\mathbf{g}_t| \exp(-\frac{2(t-\hat{u}_s)^2}{\Delta_h})$.

Compared to TVo, RORPO allows for a faster computation of the guidance map and is consistent with the notion of path-based neighborhood introduced in Section 3.3 that specify the construction of the neighborhoods from $\mathbf{g}$.

3.3. Path-based neighborhoods

This section depicts our contribution concerning the construction of anisotropic neighborhoods, i.e. the neighborhood field $V \in \mathbb{V}$ (see Section 2.3).

We propose path-based neighborhoods to fit into thin structures of the image, possibly one superpixel width. Being based on the adjacency graph $\mathcal{A}$, the neighborhood construction ensures that the neighbors of a superpixel defines a single connected component. We recall that two superpixels are adjacent when they share a common border at pixel level, thus the adjacency is a symmetric relationship: $s \in \mathcal{A}(t) \iff t \in \mathcal{A}(s)$, $\forall s, t \in \mathcal{S}$. Then, a path of length $n \in \mathbb{N}$ is an ordered list $(s_0, \ldots, s_n)$ of consecutive adjacent superpixels.

More formally, let us denote by $\Pi_K(s,t)$ the set of paths joining any pair of superpixels $(s,t) \in \mathcal{S}^2$, without any loop, and having length K: $\Pi_K(s,t) = (s_0, \ldots, s_K) \subset \mathcal{S}^{K+1}$, such that $\forall k \in [\![0, K[\![$, $s_{k+1} \in \mathcal{A}(s_k)$, $s_0 = s$, $s_K = t$, and $\forall j, k \in [\![0, K]\!]$, $s_j \neq s_k$. Similarly, we also define $\Pi(s,t)$, the set of paths joining superpixels s, t with any length, by extension:

$$\Pi(s,t) = \bigcup_{K \in \mathbb{N}} \Pi_K(s,t).$$

The proposed path-based neighborhoods relies on previously estimated guidance map $\mathbf{g}$ that contains the information about the orientation and saliency of the structures of the scene, useful to define neighborhoods in every superpixel. In particular, when the norm $\|\mathbf{g}_s\|$ at a given superpixel s is below a fixed threshold, the neighborhood in s is $V(s) = \mathcal{A}(s)$, i.e. an isotropic neighborhood that ensures adjacency. Otherwise, the set of neighbors is given by the union of the elements of two paths that expand from s to the two opposite directions corresponding to the orientation of $\mathbf{g}_s$. In the next subsections, we present the two options investigated for constructing the path-based neighborhoods.

3.3.1. Target-based neighborhood

Target-Based Neighborhood (TBN) is derived from paths that join, starting from a source superpixel $s \in \mathcal{S}$, two "targets" corresponding to distant superpixels $(t_0^*, t_1^*) \in (\mathcal{S} \setminus \{s\})^2$. Specifically, for $j \in \{0, 1\}$, the targets are selected with

$$t_j^* \in \underset{t \in \mathcal{S} \text{ s.t. } (-1)^j \langle \mathbf{g}_s, \overrightarrow{st} \rangle > 0}{\operatorname{argmin}} \|I(s) - I(t)\|_2^2 - \eta \|\overrightarrow{st}\|_2 \times \left| \leqslant \mathbf{g}_s, \overrightarrow{st} \geqslant \right|, \quad (6)$$

where $\langle ., . \rangle$ denotes the dot product between two vectors so that the constraint $(-1)^j \langle \mathbf{g}_s, \overrightarrow{st} \rangle > 0$ refers to an half-space domain, $\leqslant ., . \geqslant$ stands for the cosine similarity (also called normalized dot product) between two vectors, $\|.\|_2$ is the Euclidean norm of a vector, $|.|$ is the absolute value of a real number, and $\eta \in \mathbb{R}_{>0}$ an hyperparameter to set.

In Equation (6), the first term favors the superpixels s and t to share similar image intensities while the second one favors far targets being aligned with $\mathbf{g}_s$. Note that, for selecting close neighbors, the range of search is restricted to an ellipse centered at s with major axis aligned with $\mathbf{g}_s$ and that solutions are derived by Dynamic Programming (DP).

Then, the paths are selected among the two sets $\Pi(s, t_0^*)$ and $\Pi(s, t_1^*)$ joining s to t_0^* and t_1^*, respectively. For doing so, we formulate a cost function that the optimal path (denoted by p_j^*), $j \in \{0, 1\}$, has to minimize:

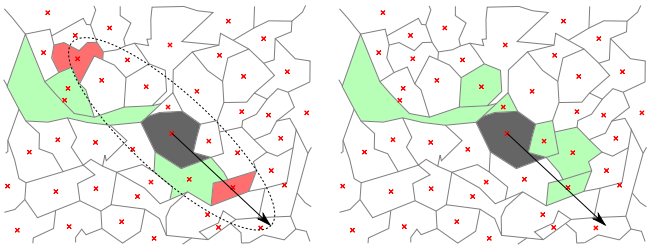
$$p_j^* \in \underset{p \in \Pi(s, t_j^*)}{\operatorname{argmin}} \sum_{k=0}^{|p|-1} \|I(p(k)) - I(p(k+1))\|_2^2, \quad (7)$$

where $|p|$ stands for the length of the path p, and $p(k)$ denotes the k^{th} element of it. The term to minimize in Equation (7) is large when the gray levels of successive superpixels along a path are dissimilar and small otherwise. We also use DP to derive a minimizer of previous equation, and the neighborhood $V(s)$ is finally constructed as the set of the superpixels in p_0^* or in p_1^*, but s: $V(s) = (p_0^* \cup p_1^*) \setminus \{s\}$. The adjacency along these paths being ensured by construction, the derived neighborhood forms a single connected component.

We give an illustration of this kind of neighborhood in Figure 2a. While this neighborhood ensures that the set of neighbors of a superpixel s forms a single connected component and favors paths oriented in the estimated direction of thin structures, there is no guarantee concerning the cardinality of these paths. Depending on the image content that influences the location of the target sites, one site may have a very small amount of neighbors in a very contrasted location, or conversely could possibly have a large number of neighbors if there exists an arbitrary long path with constant radiometry. In our experiments, we set $\eta = 30$.

3.3.2. Cardinal-based neighborhood

Cardinal-Based Neighborhood (CBN) is also a path-based neighborhood. However, instead of constraining the path extremities like TBN (see Section 3.3.1), it constraints path cardinality (and thus neighborhood cardinality): $\forall s \in \mathcal{S}$, $V(s)$ is the union (excepting element s) of two length-fixed paths $p_0^*, p_1^* \in \Pi_K(s, \cdot)$, with $\Pi_K(s, \cdot)$ denoting the set of paths of length $K \in \mathbb{N}_{>1}$ starting from s. Additionally, these paths are encouraged to expand in opposite directions.



(a) Target-Based Neighborhood (b) Cardinal-Based Neighborhood

Fig. 2: Toy example of path-based neighborhoods. The arrow indicates the orientation of $\mathbf{g}_s$ and superpixels of neighborhood are shown in color. For TBN, two "targets" (in red) are selected in an ellipse centered on the source superpixel s (in grey). For CBN, two paths with $K = 3$ elements are built to be aligned with $\mathbf{g}_s$, taking into account radiometric similarity with the site s.

As previously, we define a cost function presenting a tradeoff between fidelity to the thin structure orientation and fidelity to the gray level of originating superpixel s. For any $j \in \{0, 1\}$,

$$p_j^* \in \underset{p \in \Pi_K(s,\cdot)}{\operatorname{argmin}} \sum_{t \in p} \|I(s) - I(t)\|_2^2 + \eta' \psi_j(\overrightarrow{st}, \mathbf{g}_s),$$

where $\eta' \in \mathbb{R}_{>0}$ is an hyperparameter to set, and

$$\psi_j(\vec{u}, \vec{v}) = \begin{cases} \arccos\left(|\leqslant \vec{u}, \vec{v} \geqslant|\right) & \text{if } (-1)^j \langle \vec{u}, \vec{v} \rangle > 0, \\ +\infty & \text{otherwise,} \end{cases}$$

measures the angle between the vectors $\vec{u}$ and $\vec{v}$ and discriminates whether the dot product is positive or not.

Note that, in CBN, the cost function compares gray level and positions of each site of the path versus the site s instead of computing these differences on the adjacent sites on the path (as with TBN), to allow local deviations while ensuring global neighborhood orientation and gray level value. Figure 2b shows an example of neighborhood constructed with such an approach. In our experiments, we set $\eta' = 100$ and $K = 3$.

4. Experiments and results

4.1. Data

We test our approach on some scenes extracted from the public[1] Middlebury College dataset [24]. For each scene, a ground truth depth map and an all-in-focus RGB image are provided. Both images have 360×360 pixels. These images enable us to simulate the desired set of blurred images thanks to the defocusing algorithm [21], that is currently available as a Matlab source on MathWorks file exchange. In our simulations, the multiple images correspond to different focal object plane depths along the axis $\mathbf{e}_2$ (whereas $(\mathbf{e}_0, \mathbf{e}_1)$ is a basis for image plane). For simplicity and readability, we simulate images at focus values regularly sampled and set this step to be the unit. The maximum depth, denoted by $\Delta_h \in \mathbb{N}$, is therefore equal to n_{foc} that we set equal to 50. However, images taken at irregular steps could be considered as well without loss of generality.

[1] https://vision.middlebury.edu/stereo/data/

Then, the set of defocused images is assumed to be the only input data available, and we reconstruct depth values based on the following steps. Firstly, we compute the sharpness profiles in each pixel independently and from maximum of these profiles, we derive the *blind* estimate of all-in-focus image. Secondly, we compute the superpixels from this *blind* all-in-focus image. The number of superpixel algorithms proposed in the literature is rather important, including different kinds of superpixels that embed different properties, such as the adherence to the boundaries of the objects, the compactness or convexity of the resulting superpixels, their regularity, or the smoothness of their boundaries. We refer the reader to [6] to have an overview of the variety of superpixel algorithms. In practice, after a few comparisons, we focus on the superpixels provided by an algorithm called ETPS [25], since it is energy based (as the general framework adopted for our work) and offers relatively smooth and regular superpixels. Thirdly, as described in Section 2.2, the sharpness profile in each superpixel as well as the *blind* superpixel depth map $\hat{\mathbf{u}}$ are derived. Fourthly, we compute the guidance map $\mathbf{g}$ and construct the neighborhood field $\{V(s), \forall s \in \mathcal{S}\}$, based on the chosen method as described in Sections 3.1.2, 3.2.2, 3.3.1 and 3.3.2. Fifthly, V and $\hat{\mathbf{u}}$ allow us to instantiate our anisotropic regularization and to derive the regularized depth map results presented in the following next sections.

4.2. Evaluation criteria

The Ground Truth (GT) provided in Middlebury College dataset [24] is at pixel level. To perform evaluation, we duplicate the depth estimated for a given superpixel to each of its pixels. In the following, for the sake of clarity, we prefer not to change the variable name so that we also denote by $\mathbf{u}$ the estimated depth map at pixel level (the element lattice $\mathcal{P}$ or $\mathcal{S}$ removing ambiguity if any) and by $\tilde{\mathbf{u}}$ the GT.

Evaluation metrics. We focus on three complementary global metrics, namely RMSE (Root Mean Square Error) that has good additive properties, PSNR (Peak Signal to Noise ratio) derived from RMSE and SSIM (Structural Similarity Index Measure [26]) that bases on perception-model to measure the similarity between two images. The mathematical expressions of these metrics are as follows:

$$\text{RMSE}(\mathbf{u}, \tilde{\mathbf{u}}) = \sqrt{\frac{1}{\sharp\mathcal{P}} \sum_{p \in \mathcal{P}} (u_p - \tilde{u}_p)^2},$$

$$\text{PSNR}(\mathbf{u}, \tilde{\mathbf{u}}) = 20 \log_{10}\left(\frac{\Delta_h}{\text{RMSE}(\mathbf{u}, \tilde{\mathbf{u}})}\right),$$

$$\text{SSIM}_\Omega(\mathbf{u}, \tilde{\mathbf{u}}) = \frac{1}{\sharp\mathcal{P}} \sum_{p \in \mathcal{P}} \frac{\left(2\mu_{u,p}\mu_{\tilde{u},p} + C_1\right)\left(2\sigma_{u,\tilde{u},p} + C_2\right)}{\left(\mu_{u,p}^2 + \mu_{\tilde{u},p}^2 + C_1\right)\left(\sigma_{u,p}^2 + \sigma_{\tilde{u},p}^2 + C_2\right)},$$

where $\sharp\mathcal{P}$ stands for the cardinality of $\mathcal{P}$, Ω is a window centered at any pixel p and of size 7×7 in our case, $\mu_{u,p}$, $\mu_{\tilde{u},p}$ are the means over Ω centered at p of $\mathbf{u}$ and $\tilde{\mathbf{u}}$ values respectively, $\sigma_{u,p}^2$, $\sigma_{\tilde{u},p}^2$, and $\sigma_{u,\tilde{u},p}$ are the variances and covariance, respectively. Finally, the constants C_1 and C_2 are computed from Δ_h as $C_1 = (0.01\Delta_h)^2$ and $C_2 = (0.03\Delta_h)^2$ for numerical stability.

This is the version of SSIM specified in [26] with (according to their notations) $\alpha = \beta = \gamma = 1$. By computing the variances, covariance and mean values on a set of windows covering the whole image, SSIM incorporates comparison measurements of luminance, contrast and structure of images that allows to take into account important perceptual phenomena in its evaluation.

For result comparison, we remind that the lower the RMSE values are (in $[0, \Delta_h]$), the better the results are while for PSNR and SSIM criteria, higher values (in $\mathbb{R}_{\geq 0}$ and $[0, 1]$ respectively) reflect better performance.

Evaluation maps. Three complementary kinds of maps allow us to visualize the difficult areas. Firstly, depth error map, called E, will stress the image areas with poorest reconstruction. Secondly, neighborhood orientation map will represent saliency and direction information extracted from the guidance map, that allows us to evaluate qualitatively this latter. Thirdly, depth dynamic within neighborhoods, called Q_V, provides a measure of the neighborhood consistency in terms of depth. Pixel values of E and Q_V maps are computed as follows:

$$E(p) = |u_p - \tilde{u}_p|, \quad \forall p \in \mathcal{P},$$

$$Q_V(p) = \max_{q \in V(p)} |\tilde{u}_q - \tilde{u}_p|, \quad \forall p \in \mathcal{P},$$

where $V(p)$ at pixel level is simply the set of pixels that belong to any superpixel neighbors of the superpixel including p.

Concerning the interpretation of these maps, the lower the E values (in $[0, \Delta_h]$), the better the depth estimation at considered pixel. The orientation map is expected to be relatively smooth while following the sharp edges of the objects and aligning with the thin structures. Finally, in Q_V, low values (in $[0, \Delta_h]$) reflect a consistent neighborhood (without implying uniqueness of the solution). Note that a major benefit of Q_V criterion is that it does not require any neighborhood ground truth (which we obviously do not have).

4.3. Alternative approaches considered for comparison

To evaluate the benefits of our approach compared against isotropic neighborhoods or simplest anisotropic ones, we focus on the following alternative approaches.

Stawiaski's isotropic neighborhood. In [9], an isotropic neighborhood is computed such that the superpixels that share a common border are neighbors and their interactions are weighted by the length of this common border. This neighborhood corresponds to the adjacency relationship, with a weighting function. This formulation ensures that the set constituted by a superpixel and its neighbors is a single connected component, but it does not ensure that the barycenters of neighboring superpixels are close from each other: Very large superpixels may therefore be included in the neighborhood of a given superpixel s while most of the pixels that constitute them are actually far from s. Fortunately, such configurations are rare for regular superpixels.

Shape-based neighborhood inspired from [14]. Shape-based neighborhood is an intuitive method for building anisotropic neighborhoods. The "shape" refers to the approximation of the neighborhood as a parametric shape, namely ellipse in our case. The neighbors of a superpixel s are then the superpixels whose barycenter is included in the "shape" centered in s. We considered in our case parametric ellipses whose major semi axis directions are given by the guidance map in s. Practically, when saliency in superpixel s is very low, i.e. $\|\mathbf{g}_s\|$ is lower a given threshold (0.05), the direction is not reliable so that we rather define neighborhood as a disc, which boils down to the isotropic superpatch neighborhood of [14]. Note that we do not exploit further saliency information that appears noisier than direction, and set the eccentricity as a constant parameter of the model.

Finally, let us underline that since such neighborhoods are computed from barycenters positions, they do not enforce adjacency of the neighbors. In particular, when the superpixels are highly irregular, concave, or with low compactness, they may yield neighborhoods with disconnected components. However, with compact, regular and convex superpixels, shape-based neighborhood provides an efficient and intuitive method for building anisotropic neighborhoods.

Perfect neighborhood. For having an estimation of the possibly best performances brought by an anisotropic approach, we propose a "so-called" *perfect* anisotropic neighborhood. The latter is computed with respect to GT depth map $\tilde{\mathbf{u}}$ as follows. *Perfect* neighborhood is implemented as a shape-based neighborhood with a disc of given radius centered in $s \in \mathcal{S}$, where we remove the neighbors presenting a depth difference between the depths of GT and s higher than a fixed threshold $D_V = \frac{\Delta_h}{10} + 1$. Additionally, elements that do not belong to the s connected component are removed from the neighbors. Thus, *perfect* neighborhood refers to a neighborhood having good properties in terms of homogeneity, connectivity and shape, even if it is not unique.

4.4. Results

4.4.1. Global performance analysis

Let us first consider global performance obtained considering the whole Middlebury college dataset. Figure 3 shows the results achieved using 5000 superpixels (ETPS [25]), in terms of RMSE (allowing summing individual image performance), varying the regularization parameter α. We notice that the *perfect* neighborhood and the CBN, either from RORPO or TVo, yield the lowest RMSE values meaning they outperform all the other approaches for a wide range of regularization coefficients. Since *perfect* neighborhood was designed to evaluate the performance gain specifically related to anisotropic neighborhood (leaving apart the question of its estimation) with respect to isotropic one (represented by Stawiaski's approach), the results clearly underline the benefit of anisotropic neighborhood for regularization. A satisfactory result is that CBN provides almost as good results as *perfect* neighborhood (which we remind is unrealistic since it requires GT), stressing the performance of neighborhood estimation itself. Comparing with "ellip" that refers to the "Shape-based neighborhood inspired from [14]", we notice that these latest results are much worse, underlining

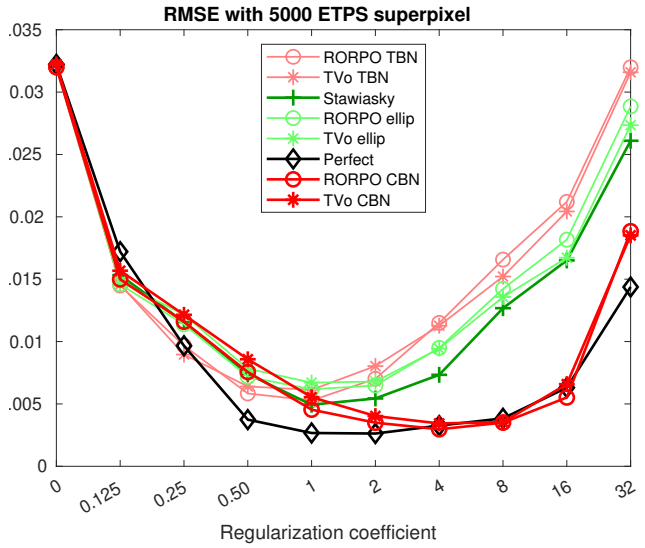


Fig. 3: Comparison of neighborhood anisotropy benefit measured through RMSE on the whole dataset. The results are achieved using 5000 ETPS superpixels [25] and different neighborhood estimations.

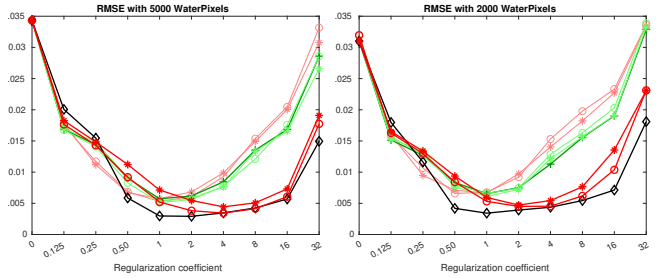


Fig. 4: Comparison of neighborhood anisotropy benefit measured through RMSE on the whole dataset. The results are achieved using either 5000 WP (left) or 2000 WP (right) using different neighborhood estimations. The legend is the same as in Figure 3.

the importance of a fine (not too simplistic) estimation. About TBN estimation, we notice it only leads to interesting results for low regularization ($\alpha < 1$). Finally, we also note that RORPO or TVo use for $\mathbf{g}$ estimation does not really impact the results, but a very slight advantage for RORPO. In conclusion, according to Figure 3, best performance is achieved by RORPO CBN, with a very noticeable robustness of the results with respect to regularization coefficient, $\alpha \in [2, 16]$. This robustness of CBN to the regularization parameter that is also confirmed by visual inspection of error maps, is one of the strengths of this approach against its alternatives.

We now check the result dependency to superpixel segmentation. However, to investigate how results are dependent on ETPS superpixels, we consider, as an alternative to ETPS superpixels [25], the WaterPixels (WP) proposed in [27]. Figure 4 shows curves analogous to those in Figure 3 considering either 5000 (like with ETPS superpixels) or 2000 WP, respectively. With respect to Figure 3, we notice the curves and conclusions are remarkably similar, but a slight loss of performance when the number of superpixels is lower (it can be seen looking at the lowest value achieved considering *perfect* neighborhood) and a more distinct advantage for RORPO with respect to TVo (when looking at the CBN curves).

To further investigate the performance variability with respect to scene and/or superpixels, Figure 5 and Figure 6 respectively show the PSNR and the SSIM obtained on each scene, for the best result obtained with a varying α. First of all, the remarks concerning the robustness to the two kinds of considered superpixels (ETPS and WP) or their number (5000 and 2000) still hold: Difficult scenes are the same and CBN achieves very interesting performance in most cases. Indeed, on some scenes such as *Aloe1*, *Books1*, *Wood11*, all approaches yield equivalent results, whereas in other scenes such as *Lampshade*, *Plastic1* or *Reindeer*, achieved results appear more sensible to neighborhood estimation. We also note that the two criteria PSNR and SSIM are complementary since differences of performance can be visible in only one of them, such as with scene *Midd11* or *Moebius1*. However, let us underline that in most scenes, the top trio is RORPO-CBN, TVo-CBN and quite obviously *perfect* neighborhood. These approaches outperform both isotropic neighborhood (represented by Stawiaski's approach) and naive anisotropic one (ellipse-based). Nevertheless we also confirm the fact that an isotropic neighborhood assumption is preferable to too naive anisotropic neighborhood estimation. In conclusion, despite the scene disparity inducing variable performance, the main conclusions concerning the benefit of anisotropic neighborhood fine estimation can also be drawn at scene level.

4.4.2. *Detailed analysis of two cases*

For further analysis, we present the corresponding error maps and neighborhood quality maps, focusing on some cases where the performance highly depends on the type of neighborhood, such as with the *Lampshade* scene and the *Reindeer* one.

Let us first consider the neighborhood estimation quality. As specified in Section 4.2, the values of the depth dynamic within a neighborhood are in $[0, \Delta_h]$, with low values reflecting a consistent estimation of neighborhood. From Figure 7, we clearly see that most of the heterogeneous neighborhoods are located at the borders of thin structures such as the lampshade rod or the reindeer antlers. We also note that lowest values are achieved for the perfect neighborhood (by construction) and then by CBN (either from TVo or RORPO guidance map) whereas both TBN and Stawiaski's neighborhood are much worse in terms of homogeneity.

Secondly, we compare the guidance maps provided by TVo and by RORPO. Figure 8 shows these maps in the cases of the two considered scenes, *Lampshade* and *Reindeer*. In both cases, we notice that the direction of the structures is rather well estimated although we also observe some noise. Comparing the two estimators, we note that while TVo looks smoother in terms of orientation (especially on *Reindeer* scene), RORPO both better detects the isotropic areas (in white) and highlights well the sharp areas of the scene. However, these observed differences seems to have only little impact on the neighborhood consistency as depicted in Figure 7 or on depth map reconstruction. In what follows, we now focus on RORPO algorithm.

Finally, let us observe the error maps versus regulariza-

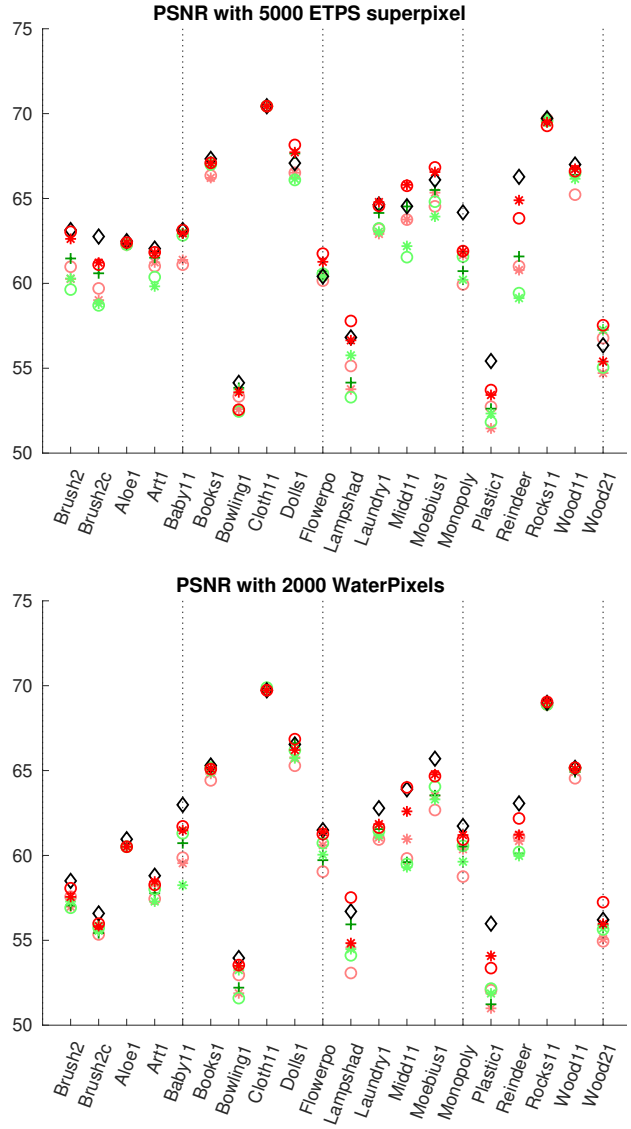


Fig. 5: Per scene best results in terms of PSNR measure for each neighborhood construction using either 5000 ETPS superpixels (top) or 2000 WP (bottom). The higher the value is, the better the result is.

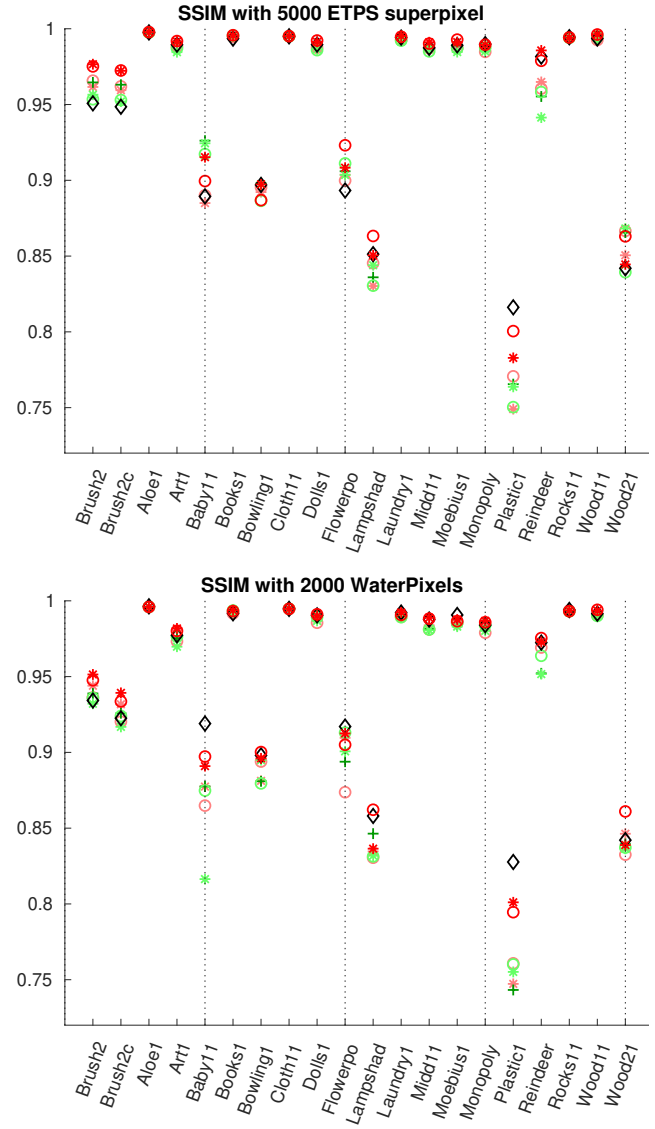


Fig. 6: Per scene best results in terms of SSIM for each neighborhood construction; 5000 ETPS superpixels (top) or 2000 WP (low). The higher the value is, the better the result is.

tion parameter α for our two scenes and the three methods of neighborhood estimation: *Perfect* (reference for benefit of anisotropic neighborhood), RORPO CBN and Stawiaski (reference for isotropic neighborhood). For *Lampshade* scene, we notice the very high noise level in the absence of regularization ($\alpha = 0$) that is progressively corrected by increasing α before new errors this time due to the removal of thin structures appear. This phenomena can be clearly seen in the case of Stawiaski's neighborhood with apparition of errors located on the vertical thin bar or rod for $\alpha > 1$. From this scene, we also notice that the optimal α values vary with the considered neighborhood; as expected, anisotropic neighborhoods allow for higher α values without reconstruction degradation (in particular for the thin structures). Specifically, in *Lampshade* scene, α values providing best results are equal to 4, 8 and 2 for the *Perfect*, RORPO CBN and Stawiaski' neighborhoods, respectively. *Reindeer* scene is much less noisy than *Lampshade* scene. However, regularization is again required to remove the *blind* estimation errors in the vertical right strip and in the bottom triangle, both been part or subparts of objects presenting a very homogeneous radiometry. Due to this lower initial level of noise, α "optimal" values are lower than in *Lampshade* scene, namely they are equal to 1, 2 and 0.5 for the *Perfect*, RORPO CBN and Stawiaski' neighborhoods, respectively. We notice that for higher values, regularization introduce depth errors on the antlers of the reindeer figure, all the more quickly as the neighborhood is isotropic (indeed with Stawiaski, bottom triangle errors cannot be corrected without degrading reindeer antlers). Using anisotropic neighborhoods, either *Perfect* or RORPO CBN, the degradation of thin structures is delayed so that we observe the existence of α values allowing for the correction of *blind* errors without introduction of new errors.

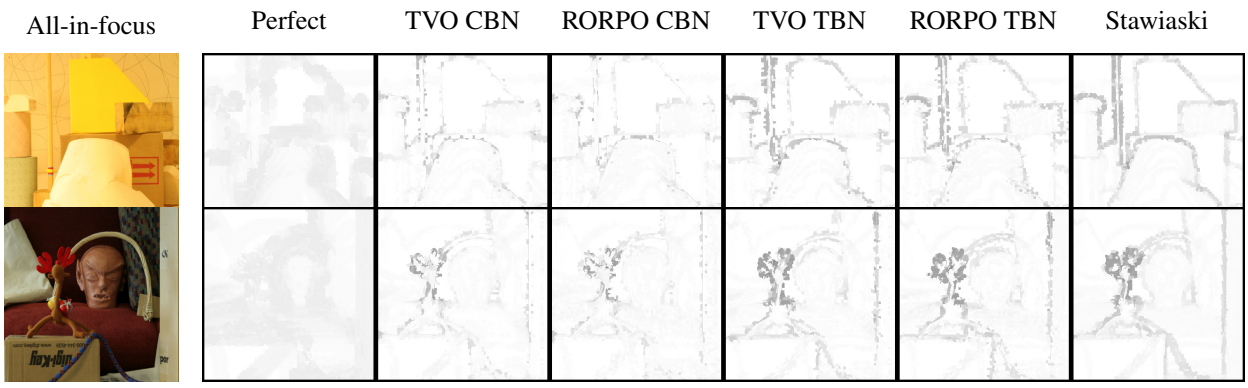


Fig. 7: Comparison of neighborhood quality Q_V for *Lampshade* (top row) and *Reindeer* (bottom row) scenes, from left to right: All-In-Focus image, Q_V maps for *perfect* neighborhood, TVo CBN, RORPO CBN, TVo TBN, RORPO TBN and Stawiaski's neighborhood. Dynamics has been reversed and spread in the interval $[0, 255]$ so that dark areas represent bad performance.

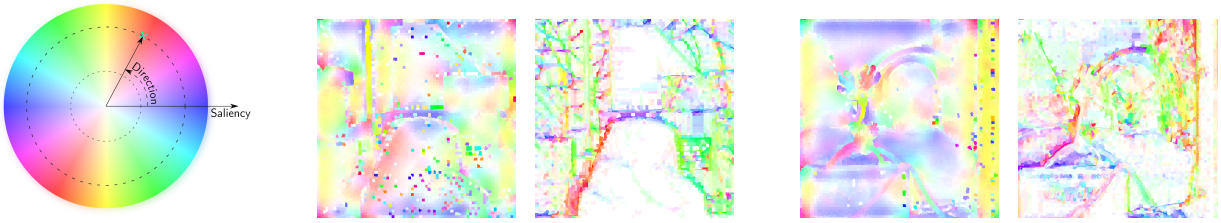


Fig. 8: Comparison of guidance maps **g** for *Lampshade* and *Reindeer* scenes, using use a color representation, such that the saturation and the hue encode respectively the saliency and the orientation; from left to right: Color wheel, TVo *Lampshade*, RORPO *Lampshade*, TVo *Reindeer*, RORPO *Reindeer*.

4.4.3. Superpixel versus pixel level

Finally, let us investigate the benefit of considering the superpixel level rather than the pixel one. For doing so, we consider again global performance statistics, namely the RMSE computed on the whole dataset.

In terms of complexity, the number of pixels is 360×360 versus 5000 superpixels (ETPS) in the considered experiments. Table 1 gives the mean running times in seconds computed over all the scenes of the Middlebury college dataset, for the four main steps of our approach: (i) superpixel segmentation, (ii) guidance map estimation (either based on RORPO or on TVo), (iii) neighborhood construction, and (iv) depth map optimization. These running times have been obtained on an Intel core i9-10900X @ 4.7 GHz, with 64 Go of RAM. Table 1 firstly confirms that RORPO is much faster (3 times) than TVo. Secondly, considering RORPO instead of TVo, the average running time for the global algorithm is 61.6 secs, i.e. about 1 minute per image. We consider this time as very encouraging since it was achieved with standard programming code, i.e. without optimization using GPU for instance. Thirdly, the running time for depth map optimization (using graph cuts) is very low thanks to the complexity reduction working with superpixels instead of pixels. For comparison, running the isotropic neighborhood depth map optimization at pixel level, the average running time is 36.8s, i.e. about 30 times slower. Thus, even without code optimization, the additional running time for anisotropic neighborhood estimation steps is compensated by the running time decrease for depth map optimization step.

In terms of performance, Figure 10 allows for comparison of the RMSE curves for three kinds of neighborhoods, namely

Superpixels segmentation	RORPO	TVo	Neighbohood construction	Depth optimization
13.2	28.1	94.6	19.0	1.3

Table 1: Mean running times (in seconds) of the main steps of our approach for computing segmentation of a scene at superpixel level.

	Lampshade		Reindeer	
	PSNR	SSIM	PSNR	SSIM
RORPO-CBN ETPS	**57.78**	**86.33**	63.83	**97.89**
4-connectivity pix. lev.	<u>54.61</u>	<u>83.81</u>	<u>63.94</u>	96.40
RORPO-CBN pix. lev.	53.32	83.00	**64.62**	<u>97.18</u>

Table 2: Results obtained for the scenes *Reindeer* and *Lampshade* with our proposed anisotropic neighborhood at superpixel level (first row), compared to isotropic neighborhood at pixel level (second row) and RORPO-CBN neighborhood at pixel level (last row). SSIM values are indicated in percentage. For each scene, best result is in bold and second best is underlined.

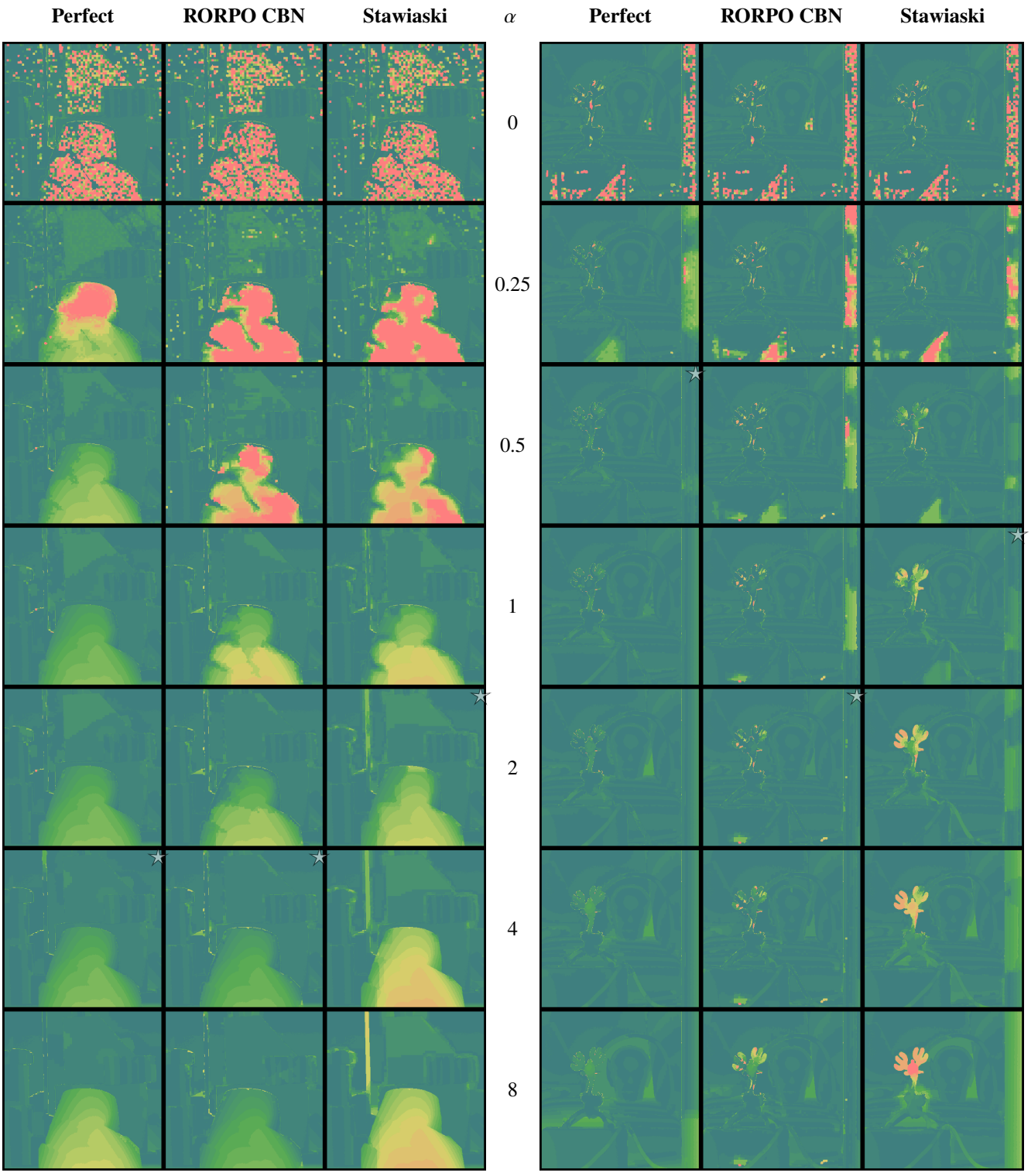


Fig. 9: Maps of depth error obtained for the scenes *Lampshade* (half left) and *Reindeer* (half right), for three neighborhood construction strategies (*Perfect*, RORPO CBN and Stawiaski) and different values of regularization parameter $\alpha \in \{0., 0.25, 0.5, 1, 2, 4, 8\}$. For better visualization, error value dynamic has been bounded to $\frac{2\Delta_h}{5}$.

Stawiaski (i.e., isotropic), RORPO CBN or RORPO TBN (representing best candidate for anisotropic neighborhoods) and *Perfect*, either at superpixel level (using 5000 ETPS superpixels) or at pixel level. First of all, from Figure 10, we notice an improvement of performance at superpixel level with respect to pixel one. This improvement is a very strong point since one could have expected that superpixels would introduce some spatial imprecision (at the benefit of complexity decrease), especially since the RMSE is measured at pixel level. Nevertheless, at least on the considered dataset, this preprocessing step is beneficial for the precise image reconstruction. This comment is confirmed in most cases when we examine individual scenes. For instance, for the two detailed cases *Lampshade* and *Reindeer*, the two first lines of Table 2 show the performance indicators PSNR and SSIM achieved by RORPO-CBN on ETPS superpixels and isotropic (4-connectivity) and we see that RORPO-CBN yields to significantly better result except in terms of PSNR on *Reindeer* scene where nevertheless the performance values are very close.

Then, we notice the potential benefit of anisotropic neighborhood with respect to isotropic one (at pixel level, Stawiaski's neighborhood boils down to 4-connectivity neighborhood) since *Perfect* neighborhood yields the best results. However, we also notice that, at pixel level, the difference of performance is very small, and that isotropic neighborhood yields slightly better result than RORPO TBN or RORPO CBN. A possible explanation is that the requirement to take into account anisotropic neighborhood is less pregnant at pixel level (due to the size of neighborhood with respect to objects in pixel numbers as well as the regularity of the lattice) and that neighborhood estimation is less efficient. Indeed it is based on *blind* depth estimation that may be much noisier at pixel level that at superpixel one. Besides, the performance may depend on the considered scene. For instance, Table 2 shows that on the *Reindeer* scene, RORPO-CBN at pixel level slightly outperforms isotropic pixel level both in terms of PSNR and SSIM indicators. These observations also open perspectives to understand the relationship between scene feature and scale of analysis (from pixel level to superpixel ones).

In conclusion, the benefit of presegmenting the scene in superpixels and then handling anisotropic neighborhood appears both in terms of global performance and in terms of robustness with respect to regularization parameter α. Besides the additional complexity introduced by neighborhood estimation (RORPO-CBN according to this study) is compensated by the complexity decrease when handling much less superpixels than pixels.

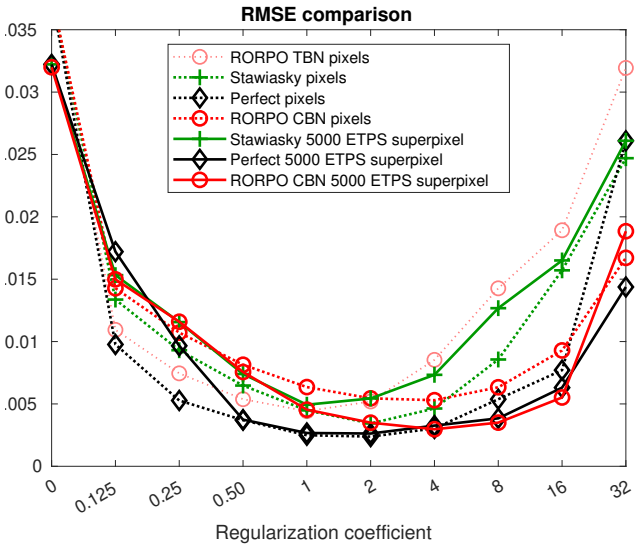


Fig. 10: Superpixel versus pixel level: Comparison in terms of RMSE computed on the whole dataset, for three kinds of neighborhoods.

5. Conclusion and perspectives

In this paper, we propose some new anisotropic neighborhoods that offer a flexible and generic formulation with respect to the site lattice (i.e. possibly irregular). For doing so, we select and customize two vesselness operators and we show their efficiency thanks to their properties of noise robustness or adaptability to thin structures. Finally, we evaluate and study the benefit of the constructed anisotropic neighborhoods in particular for thin structure preservation. Specifically, we consider SFF application and we evaluate our results on a reference dataset both according to quantitative criterion but also based on qualitative observation of evaluation maps.

Future works will involve the following perspectives. Firstly, we aim at studying the relationships between the hyperparameters characterizing the neighborhoods and the superpixel ones (regularity, number), also relating these parameters to the scale of scene main features and objects. Secondly, focusing on RORPO-CBN approach that appears to provide best performance and based on the evaluation of the running times per process, we will focus on the code optimization of the RORPO module. Thirdly, since the proposed anisotropic neighborhood construction can be useful for many energetic formulations of discrete inverse problem as confirmed by preliminary tests on binary segmentation [28], we aim at considering segmentation of thin structures such as frequently encountered in medical imaging (e.g., vessels) or remote sensing imagery (e.g., roads, rivers). Fourthly, in the proposed approach, neighborhood construction relies on guidance map itself estimated from a first (blind) evaluation of the solution. We aim at evaluating the benefit of using a current evaluation of the solution for our neighborhood construction process. Although such an alternate minimization seems attractive, first tests showed that the risk of divergence from the GT solution is real so that we will have to define rigorously the convergence conditions.

Appendix A 3D Tensor Voting

Let $\mathbb{R}^{3\times3}$ with an origin coordinate O in $\mathbb{R}^3$ be the considered vector space, endowed with a voting function $VF : \mathbb{R}^{3\times3}\times\mathbb{R}^3 \mapsto \mathbb{R}^{3\times3}$. A tensor can be represented by a matrix $\mathbb{T} \in \mathbb{R}^{3\times3}$. The voting operation VF builds a new tensor $\mathbb{T}'$ to the cast location $P \in \mathbb{R}^3$ and adds it to the tensor at this location, since tensors have good summation properties. The tensor $\mathbb{T}'$ is a combination of rotation and scaling of the source tensor $\mathbb{T}$, combinations

that are all derived from the *stick kernel*. Indeed, tensors can be decomposed in a basis of tensors, in which the stick tensor is the simplest element. The stick kernel refers to the voting operation of this stick tensor.

In tensor voting, a tensor is a second order symmetric tensor that can be represented by a positive semidefinite diagonalizable matrix $\mathbb{T} \in \mathbb{R}^{3\times3}$, whose eigenvectors are orthogonal. In addition to its coordinates, one tensor can be characterized either from six scalar values corresponding to the coefficients of the symmetric matrix or, from three eigenvalues and a rotation. This rotation defines the transformation of the orthonormal basis $(\mathbf{e}_0, \mathbf{e}_1, \mathbf{e}_2)$ to align with $(\hat{\mathbf{e}}_0, \hat{\mathbf{e}}_1, \hat{\mathbf{e}}_2) \in \mathbb{R}^{3^3}$, the set of eigenvectors sorted by decreasing eigenvalue. The *decomposition* of the matrix into a set of diagonal matrices is a key point introduced by [17]. By definition, the tensor is a diagonal matrix in the system $(\hat{\mathbf{e}}_0, \hat{\mathbf{e}}_1, \hat{\mathbf{e}}_2)$, so that:

$$\begin{pmatrix} \lambda_0 & 0 & 0 \\ 0 & \lambda_1 & 0 \\ 0 & 0 & \lambda_2 \end{pmatrix} = (\lambda_0 - \lambda_1)\mathbb{T}_{stick} + (\lambda_1 - \lambda_2)\mathbb{T}_{plate} + \lambda_2\mathbb{T}_{ball}, \quad (8)$$

where $\mathbb{T}_{stick}$, $\mathbb{T}_{plate}$ and $\mathbb{T}_{ball}$ are respectively the stick tensor, the plane one and the ball one, named according to their representations as ellipsoids (see figure in [29]), and each of them represents a different type of structure: The stick component encodes the saliency of surfaces that are normal to $\hat{\mathbf{e}}_0$, the plate component is encoding some curves with tangent direction $\hat{\mathbf{e}}_2$, and the ball component is encoding points, e.g. corresponding to thin structure junctions.

The stick kernel that allows for the vote cast by a stick tensor, $\mathbb{T}_{stick} \in \mathbb{R}^{3\times3}$, involves a multiplication of $\mathbb{T}_{stick}$ by a decay function DF, and a rotation by a vector $\mathbf{\Omega}$. Specifically, DF is as follows:

$$DF(r, \phi, \sigma_T) = \exp\left(-\frac{r^2 + v\phi^2}{\sigma_T^2}\right),$$

where σ_T is the scale parameter, v is a constant that controls the decay with curvature, $r \in \mathbb{R}_{>0}$ is the length of the circle arc between O and P on the osculating circle joining O and P with normal $\hat{\mathbf{e}}_0$ at point O and $\phi \in]-\pi, \pi]$ the angle between the tangent to the same osculating circle in O and $\overrightarrow{OP}$. The decay function allows for a smooth voting kernel whose support can be bounded to a sphere of radius $3\sigma_T$. Along with the term $v\phi^2$ used for increasing the decay with curvature, [17] proposes also to restrict vote to the area where $\phi < \frac{\pi}{4}$ and consider that the term $DF(r, \phi, \sigma_T)$ is null otherwise.

The rotation $\mathbf{R}(\mathbf{\Omega}) \in \mathbb{R}^{3\times3}$ is defined by the rotation vector $\mathbf{\Omega} \in \mathbb{R}^3$, that transforms the vector $\hat{\mathbf{e}}_0$ into the vector $\hat{\mathbf{e}}'_0$ with $\hat{\mathbf{e}}'_0$ and $\hat{\mathbf{e}}_0$ symmetrical with respect to the mediator of the segment OP. This allows for computing the cast tensor $\mathbb{T}'_{stick} \in \mathbb{R}^{3\times3}$ as follows:

$$\mathbb{T}'_{stick} = DF(r, \phi, \sigma_T)\mathbf{R}(\mathbf{\Omega})\mathbb{T}_{stick}\mathbf{R}^T(\mathbf{\Omega}).$$

where $\cdot^T$ is the transposition operation.

Plate tensor can be written $\mathbb{T}_{plate} = \hat{\mathbf{e}}_0\hat{\mathbf{e}}_0^T + \hat{\mathbf{e}}_1\hat{\mathbf{e}}_1^T$, while ball tensor is written $\mathbb{T}_{ball} = \hat{\mathbf{e}}_0\hat{\mathbf{e}}_0^T + \hat{\mathbf{e}}_1\hat{\mathbf{e}}_1^T + \hat{\mathbf{e}}_2\hat{\mathbf{e}}_2^T$. The plate and ball kernels are derived from the stick kernel by integration of stick tensors. Approximating these integrals as sums of tensors,

$$\mathbb{T}'_{plate} \approx \sum_{i=0}^{I} DF(r, \phi, \sigma_T)\mathbf{R}(\mathbf{\Omega})\mathbb{T}_{stick}(i\Delta_\rho)\mathbf{R}^T(\mathbf{\Omega})\Delta_\rho,$$

$$\mathbb{T}'_{ball} \approx \sum_{i=0}^{I}\sum_{j=-J/2}^{J/2} DF(r, \phi, \sigma_T)\mathbf{R}(\mathbf{\Omega})\mathbb{T}_{stick}(i\Delta_\rho, j\Delta_\psi) \mathbf{R}^T(\mathbf{\Omega})\sin(j\Delta_\psi)\Delta_\psi\Delta_\rho,$$

where $\Delta_\rho = \frac{\Pi}{I}$ and $\Delta_\psi = \frac{\Pi}{J}$, and $I, J \in \mathbb{N}$ are arbitrary constants. Note that these kernels are usually precomputed for computational efficiency.

Then, any tensor $\mathbb{T}_s$ at location $s \in \mathbb{R}^3$ can be decomposed from Equation (8) in a basis $(\hat{\mathbf{e}}_0, \hat{\mathbf{e}}_1, \hat{\mathbf{e}}_2)$ as $\mathbb{T}(s) = (\lambda_0 - \lambda_1)\hat{\mathbf{e}}_0\hat{\mathbf{e}}_0^T + (\lambda_1 - \lambda_2)\hat{\mathbf{e}}_1\hat{\mathbf{e}}_1^T + \lambda_2\hat{\mathbf{e}}_2\hat{\mathbf{e}}_2^T$, and the vote cast at location $t \in \mathbb{R}^3$ is written:

$$\begin{aligned} VF(\mathbb{T}, \overrightarrow{st}) &= (\lambda_0 - \lambda_1)VF(\mathbb{T}_{stick}(t), \overrightarrow{st}) \\ &+ (\lambda_1 - \lambda_2)VF(\mathbb{T}_{plate}(t), \overrightarrow{st}) \\ &+ \lambda_2 VF(\mathbb{T}_{ball}(t), \overrightarrow{st}) \end{aligned}$$

Having introduced the voting operation for one tensor, let us specify the global voting process.

From $\mathcal{S}_0, \mathcal{S}_1 \subset \mathcal{S}$ the sets of voters and the cast locations respectively, $\forall s \in \mathcal{S}$,

$$\begin{cases} \forall p \notin \mathcal{S}_1, & \mathbb{T}'(p) = \mathbb{T}(p), \\ \forall p \in \mathcal{S}_1, & \mathbb{T}'(p) = \mathbb{T}(p) + \sum_{s\in\mathcal{S}_0} VF(\mathbb{T}(s), \overrightarrow{sp}), \end{cases}$$

where $\mathbb{T}'(s)$ is the tensor at location s after vote and $\mathbb{T}(s)$ before.

References

[1] S. Geman, D. Geman, Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images, IEEE Transactions on Pattern Analysis and Machine Intelligence 6 (1984) 721–741.

[2] R. Szeliski, R. Zabih, D. Scharstein, O. Veksler, V. Kolmogorov, A. Agarwala, M. Tappen, C. Rother, A comparative study of energy minimization methods for Markov random fields with smoothness-based priors, IEEE Transactions on Pattern Analysis and Machine Intelligence 30 (2008) 1068–1080.

[3] C. Ribal, N. Lermé, S. Le Hégarat-Mascle, Efficient graph cut optimization for shape from focus, Journal of Visual Communication and Image Representation 55 (2018) 529–539.

[4] Y. Boykov, M.-P. Jolly, Interactive graph cuts for optimal boundary & region segmentation of objects in N-D images, in: Proceedings of International Conference on Computer Vision, volume 1, 2001, pp. 105–112.

[5] P. Favaro, Recovering thin structures via nonlocal-means regularization with application to depth from defocus, in: Proceedings of International Conference on Computer Vision and Pattern Recognition, 2010, pp. 1133–1140.

[6] D. Stutz, A. Hermans, B. Leibe, Superpixels: An evaluation of the state-of-the-art, Computer Vision and Image Understanding 166 (2018) 1–27.

[7] S. Gould, R. Fulton, D. Koller, Decomposing a scene into geometric and semantically consistent regions, in: Proceedings of International Conference on Computer Vision, 2009, pp. 1–8.

[8] P. Arbeláez, M. Maire, C. Fowlkes, J. Malik, Contour detection and hierarchical image segmentation, IEEE Transactions on Pattern Analysis and Machine Intelligence 33 (2011) 898–916.

[9] J. Stawiaski, E. Decencière, Region merging via graph-cuts, Image Analysis & Stereology 27 (2011) 39.

[10] Y.-J. Liu, C.-C. Yu, M.-J. Yu, Y. He, Manifold SLIC: A fast method to compute content-sensitive superpixels, in: Proceedings of International Conference on Computer Vision and Pattern Recognition, 2016, pp. 651–659.

[11] B. Fulkerson, A. Vedaldi, S. Soatto, Class segmentation and object localization with superpixel neighborhoods, in: Proceedings of International Conference on Computer Vision, 2009, pp. 670–677.
[12] B. Cui, X. Xie, X. Ma, G. Ren, Y. Ma, Superpixel-based extended random walker for hyperspectral image classification, IEEE Transactions on Geoscience and Remote Sensing 56 (2018) 1–11.
[13] S.-C. Pei, W.-W. Chang, C.-T. Shen, Saliency detection using superpixel belief propagation, in: Proceedings of International Conference on Image Processing, 2014, pp. 1135–1139.
[14] R. Giraud, V.-T. Ta, A. Bugeau, P. Coupe, N. Papadakis, SuperPatchMatch: An algorithm for robust correspondences using superpixel patches, IEEE Transactions on Image Processing 26 (2017) 4068–4078.
[15] Y. Yu, H. Guan, Z. Ji, Rotation-invariant object detection in high-resolution satellite imagery using superpixel-based deep Hough forests, IEEE Geoscience and Remote Sensing Letters 12 (2015) 2183–2187.
[16] S. Nayar, Y. Nakagawa, Shape from focus, IEEE Transactions on Pattern Analysis and Machine Intelligence 16 (1994) 824–831.
[17] G. Medioni, C.-K. Tang, M.-S. Lee, Tensor Voting: Theory and applications, in: Proceedings of RFIA, 2000.
[18] O. Merveille, H. Talbot, L. Najman, N. Passat, Curvilinear structure analysis by ranking the orientation responses of path operators, IEEE Transactions on Pattern Analysis and Machine Intelligence 40 (2018) 304–317.
[19] M. Moeller, M. Benning, C. Schonlieb, D. Cremers, Variational depth from focus reconstruction, IEEE Transactions on Image Processing 24 (2015) 5369–5378.
[20] V. Gaganov, A. Ignatenko, Robust shape from focus via Markov random fields, Conference "GraphiCon'2009" (2009) 74–80.
[21] S. Pertuz, D. Puig, M. Garcia, Analysis of focus measure operators for shape-from-focus, Pattern Recognition 46 (2013) 1415–1432.
[22] Y. Boykov, V. Kolmogorov, An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision, IEEE Transactions on Pattern Analysis and Machine Intelligence 26 (2004) 1124–1137.
[23] Q. Zou, Y. Cao, Q. Li, Q. Mao, S. Wang, Cracktree: Automatic crack detection from pavement images, Pattern Recognition Letters 33 (2012) 227–238.
[24] D. Scharstein, C. Pal, Learning conditional random fields for stereo, in: Proceedings of International Conference on Computer Vision and Pattern Recognition, 2007, pp. 1–8.
[25] J. Yao, M. Boben, S. Fidler, R. Urtasun, Real-time coarse-to-fine topologically preserving segmentation, in: Proceedings of International Conference on Computer Vision and Pattern Recognition, 2015, pp. 2947–2955.
[26] Z. Wang, H. R. Sheikh, Image quality assessment: From error visibility to structural similarity, IEEE Transactions on Image Processing 13 (2004) 14.
[27] V. Machairas, M. Faessel, S. Cárdenas-Peña, T. Chabardes, T. Walter, E. Decencière, Waterpixels, IEEE Transactions on Image Processing 24 (2015) 3707–3716.
[28] C. Ribal, N. Lermé, S. Le Hégarat-Mascle, Thin structures segmentation using anisotropic neighborhoods, in: Information Processing and Management of Uncertainty in Knowledge-Based Systems, volume 1237, 2020, pp. 601–612.
[29] G. Medioni, P. Mordohai, M. Nicolescu, The Tensor Voting Framework, in: Handbook of Geometric Computing, Springer-Verlag, Berlin/Heidelberg, 2005, pp. 535–568.