Leveraging Mutual Information in Local Descriptions: From Local Binary Patterns to the Image

Tahir Q. Syed(✉), Sadaf I. Behlim, Alishan K. Merchant, Alexis Thomas, and Furqan M. Khan

Visual Analytics Lab, National University of Computer and Emerging Sciences, Karachi, Pakistan
{tahir.syed,sadaf.iqbal,k112214,k112026,furqan.khan}@nu.edu.pk

Abstract. Local image descriptors provide robust descriptions of image localities. Their geometric arrangement provides additional information about the image they describe, a fact often ignored when employing them to that wide slew of tasks from image registration to scene classification. On the premise that descriptor quality could be assessed in terms of its expressiveness of image content, we investigate the use of the described as well as that additional geometric information to the task of recovering the image from its local descriptors. This paper uses Local Binary Patterns, an operator nested in a dense geometry, to study how this additional information in the form of constraints among pixels dictates the intensity estimated for a pixel. We determine that constraints propagate from regional extrema to regions around them that observe the same constraint class, and that the intensity for any of the region's pixels influences that for all others. We build a directed constraint graph of pixel nodes such that the arcs on the graph are strongly k-consistent, and propagate intensity estimates from extremum nodes. Evaluations are run on the SIPI texture and the BSD500 datasets. The estimates preserve the local structure of the image, as shown by the Mean Absolute Error of about 15% and 18% respectively and Structural Texture SIMilarity of about 92% for both datasets, in addition to observing 100% constraint satisfaction.

1 Introduction

Local image descriptors describe image localities in compact yet rich ways, which is why they have been successfully employed in numerous computer vision applications such as image retrieval [1], action recognition [2,3], object detection and recognition [4]. The success of local descriptors is due to their expressiveness of image content, which has been quantified by seminal work [5] via precision-recall criteria on the image retrieval task. We propose to measure this expressiveness directly by investigating the fundamental question: *could we get an intensity image back given its local description*?

T.Q. Syed and S.I. Behlim—Equal contribution.

V. Murino and E. Puppo (Eds.): ICIAP 2015, Part II, LNCS 9280, pp. 239–251, 2015.
DOI: 10.1007/978-3-319-23234-8_23

Concretely, this *inverse problem* amounts to the estimation of an image's pixel intensities given its local description which preserves not only the perceptual semantics but is also consistent with the structure arising from image content. The manner that the descriptor patches are arranged within the image has lead to two threads of investigation: 1) where the macroscopic information (arrangement) is kept as meta-data separate from the descriptor [6,7], and 2) where densely computed local descriptors allow this information to be inferred [8,9]. The work presented here lies in the second because we find it more interesting to see whether at all the inversion problem is addressable in the absence of meta information.

In this thread of investigation, Kruse et al. [10] show that the local-structure of an image could be revealed by exploring a pixel's relationship with its 8-connected neighborhood, solving for a system of inequalities, one for each pixel. This simultaneous solution is computationally prohibitive for any meaningful image size. After the advent of local image descriptors, Lindahl et al. [11] reproduce Kruse's results by estimating pixel intensities beginning with an image of uniform intensity and then running a gradient descent procedure to arrive at a similar local description as the one supplied for the original image. Wu et al. [8] also perform reconstruction by taking a permuted version of an image as input and tweaking the pixel values until the reconstructed image produce the same description as was provided for the image before permutation. These works illustrate that the encoding mechanism of descriptors maintains the local structure of the described patch, even though they do not use this information for the purpose of reconstruction. More recent work, Waller et al. [9], takes into account the information encoded by a descriptor patch. They utilize inter-pixel intensity relationship (*greater-than* or *less-than*) information provided by the LBP descriptor to estimate the pixel intensity values in a way that is consistent with these relationships, by following the longest path uphill from image minima using a recursive procedure. Since their method does not conserve the path from the minimum to any given pixel, there is no way to update the values of the pixels along the path, including the minima, which are universally estimated as 0.

In this paper we are investigating the inversion problem in a manner that takes into account both the within-patch structure information provided by the descriptor but *also* the macroscopic information between pixels of different patches interlocked in a dense geometrical relationship. We work with a dense binary image descriptor, Local Binary Patterns (LBPs) [12] which in its original and extended form is widely used in applications ranging from segmentation [13] to gait analysis [3]. This descriptors is simple and easy to compute and encodes not only the local structural information of an image but also spatial information of patch pixels w.r.t the center pixel of the patch. Our contributions in this paper are:

1. posing image inversion from binary descriptors as a constraint satisfaction problem, i.e., reformulating the problem of estimating pixel intensities as one of domain shrinkage of constrained variables.

2. proposing a constraint graph sub-graphs of which take tree forms that encode constraints (either superiority or inferiority, defined in Sec. 2) possibly between any pair of pixels in the image. For the purpose of inversion, we work only with the constraints occurring through the longest successions of constraints.
3. proposing a method that inverts an image given its LBP codes that account for all encoded constraints being satisfied, and quantifiably recovers local image contrast.

2 Local Binary Pattern and Constraint Propagation

The Local Binary Pattern (LBP) is a dense local binary descriptor that describes the relationship between pair of pixels using a two-way comparison within a given window size. It encodes the local structure of an image within a window in terms of constraints [8]. This view at the operator level would generalize to the fact that an LBP code at any given pixel constrains the intensity of its neighbors to be either greater than (*superior to*) or less than (*inferior to*) its intensity.

The LBP code for a particular pixel is the weighted sum of a Heaviside[1] function of oriented differences around each pixel of a patch (Fig. 1(a)). These codes encode only superiority and inferiority constraints present between neighborhood pixels but do not contain any information about *how much* a pixel is superior or inferior to the center pixel, that is, they have lost all contrast information. A

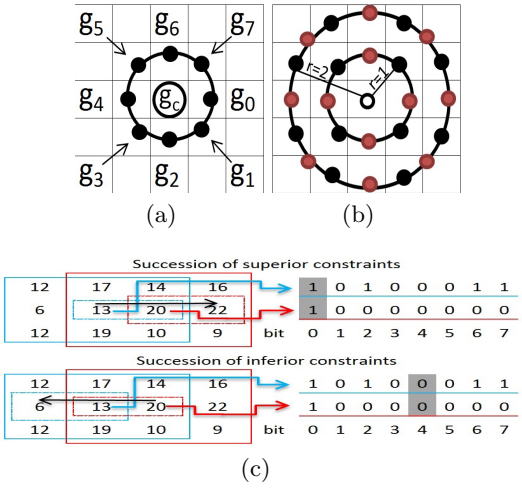


Fig. 1. (a) Neighbors of center pixel g_c participating in $LBP_{8,1}$ (b) different neighbor samplings (e.g. black pixels might be discounted) in neighborhoods of different sizes (c) succession of superiority or inferiority constraints via LBP codes

[1] $H(z) = \begin{cases} 0 & z \text{ ¡ } 0 \\ 1 & z \geq 0 \end{cases}$

pixel P_2 observes the superiority constraint, denoted by $superior(P_1, P_2)$ w.r.t pixel P_1 when the Heaviside function of difference between P_2 and P_1 results in 1, else observes the inferiority constraint denoted by $inferior(P_1, P_2)$.

$$superior(P_1, P_2) = H(P_2 - P_1) \tag{1a}$$

$$inferior(P_1, P_2) = 1 - superior(P_1, P_2) \tag{1b}$$

The LBP code could be mathematically defined in terms of the superiority constraint as:

$$LBP_{n,r}(g_c, g_i^r) = \sum_{i=0}^{n-1} superior(g_c, g_i^r) * 2^i. \tag{2}$$

where r is the radius of a circle defined on the L_∞ norm, n is the number of sampling points in the neighborhood (Fig. 1(b)), g_c is the intensity of the pixel for which the LBP code is being calculated, and g^r is a pixel at a radius r on a circle in the L_∞ space.

The LBP code could also provide information about a larger neighborhood than defined by the local patch on which it was computed, i.e. the LBP of a larger radius, $LBP_{16,2}$ could be recursively defined in terms of the $LBP_{8,1}$ primitive. The relationship between any two non-neighbor pixels can be estimated through the LBP codes if there is a succession of either superiority or inferiority constraints through any possible path i.e. relationship between pixelindex (linear index) PI-5 and PI-11 could be defined by evaluating the LBP codes of PI-5 and PI-8 as shown in Fig. 1c. Analyzing the value at bit 0 of PI-5 and PI-8 codes, which is 1, reveals that PI-8 is superior to PI-5 and PI-11 is superior to PI-8 respectively. This implies that PI-11 is superior to PI-5 by transitivity. Mathematically,

$$superior(P_i, P_j) = superior(P_i, P_k) \wedge superior(P_k, P_j) \tag{3}$$

Thus, a succession of superiority constraints, $Superior$, could be defined as:

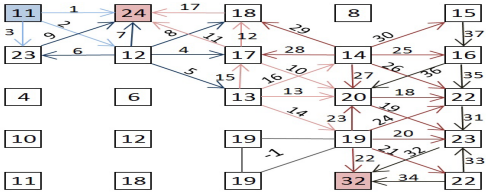
$$Superior(g_c, g_j^r) = superior(g_c, g_i^1) \wedge superior(g_i, g_k^{1 \vee \cdots \vee r}) \wedge \ldots \wedge superior(g_k^{(r-1) \vee r}, g_j^r) \tag{4}$$

where $\wedge$ is a composition and $\vee$ is a selection operator. The succession of inferiority constraints, $Inferior$, is also defined in a dual fashion (Fig. 1(c)) where the relationship to PI-2 from PI-8 is found to be inferior.

Sometimes, it is not possible to identify the relationship between two pixels using any shared pixel i.e. In Fig. 2(a), a relationship between PI-1 and PI-16 could not be determined using any neighborhood pixels. Thus, constraint propagation comes to a halt when an inferiority constraint follows a succession of superiority constraints and vice versa i.e.

$$Superior(g_c, g_j^r) \wedge inferior(g_j^r, g_p^r) \text{ or } Inferior(g_c, g_j^r) \wedge superior(g_j^r, g_p^r)$$

These opposite constraints themselves either have g_p^r as an image extremum or propagate towards an extremum. The set P of all such successions emanating from a given extremum demarcated by pixels p defines the *propagation extent* of that extremum. For instance, for minimum min_n:



(a)

Propagation extent of regional maxima				
[2,24]	2	2	[2,24]	24
2	[2,24]	[2,24]	[2,24]	24
[2,24]	[2,24]	[2,24]	24	24
[2,24]	[2,24]	24	24	24
[2,24]	24	24	24	24

Propagation extent of regional minima				
1	[1,4,11]	[1,4,11]	4	[1,4,11]
[1,11]	[1,11]	[1,4,11]	[1,4,11]	[1,4,11]
11	11	[1,11]	[1,4,11]	[1,4,11]
11	11	[1,11]	[1,11]	[1,4,11]
11	11	[1,11]	[1,4,11]	[1,11]

(b)

Fig. 2. (a) Constraint graph formed due to the arboresence expanding from the regional minimum lying at PI-1. Hotness of color encodes increasing distance from the minimum. (b) pixel memberships in multiple propagation extents of regional maxima (pink) at $PI - 6$ and $PI - 20$ and of regional minima (blue) at $PI - 1$, $PI - 3$ and $PI - 16$

$$P = \bigcup_{p} \{Superior(min_1, g_j^r) \wedge inferior(g_j^r, p^r)\} \tag{5}$$

Therefore, it is of interest to determine the positions of image extrema as well as their propagation extents.

3 The Directed Constraint Graph

Given any pixel g_c, we can enumerate all the neighborhood pixels that observe the superiority or the inferiority constraints, and for all of these neighborhood pixels a similar enumeration could be performed. This naturally lends itself to two kinds of constraint arborescence, one that encodes superiority constraints having regional minimum as the starting node and other one encode inferiority constraints having regional maximum as the starting node. Fig. 2(a) shows 3 regional minima and 2 regional maxima which are identified by analyzing the LBP codes of a particular image - it is observed that the LBP code for regional maxima evaluates to 0, and for regional minima to 255, discounting border effects.

Algorithm 1. Minima Trees

Input: A list of minima $minList = \{min_1, min_2, \ldots, min_n\}$ which contains each minimum linear index in an image.

Output: A set of tree $treeList = \{tree_1, tree_2, \ldots, tree_n\}$. Each tree corresponds to a minimum contains array of cell each $tree_{no} = \{node_1, node_2, \ldots, node_n\}$ holds information of nodes $node_n = \{nodeId, nodeValue, parNodeId, nodeLevel\}$ that observe superiority constraints where $nodeId$ is the temporary Id and $nodeValue$ is a value that is either a pixelindex or -1

```
1  for i ← 1 to size(minList) do
2      initialize expandNodesQueue, tree_i ← ∅ and nodesCount ← 1
3      push minList(i) in expandNodesQueue
4      insert root of tree_i as node_1 = {nodesCount, minList(i), NULL, 1}
5      increment nodesCount
6      while not empty(expandNodesQueue) do
7          parNodeId ← pop(expandNodesQueue)
8          superiorNodes contains parNodeId neighbors that observe
           superiority constraints
9          if not empty(superiorNodes) then
10             for each supNode in superiorNodes do
11                 supNodeLevel is parNodeLevel + 1
12                 check whether the tree_i contains supNode
13                 if false then
14                     insert supNode in tree_i as
                       {nodesCount, supNode, parNodeId, supNodeLevel}
15                     increment nodesCount
16                     push supNode in expandNodesQueue
17                 else
18                     get nodeLevel of supNode from tree_i
19                     if nodeLevel < supNodeLevel then
20                         update supNode parNodeId and supNodeLevel in
                           tree_i
21                         update nodeLevel of subtree associated with
                           supNode if exists
```

However, sometimes it is possible that a certain pixel observes the same succession of constraints from either pixel g_c or from one of its children, e.g. in Fig. 2(a) PI-2 could be reached from PI-1 and PI-7 where PI-7 and PI-2 are both children of PI-1. Therefore, a directed constraint graph emerges from every extremum which could be lead to a number of arborescences, one for each extremum. Fig. 2(a) shows the directed constraint graph emerges from minimum lying at PI-1 and Fig. 2(b) indicates for each pixel the extremum with which the pixel is associated.

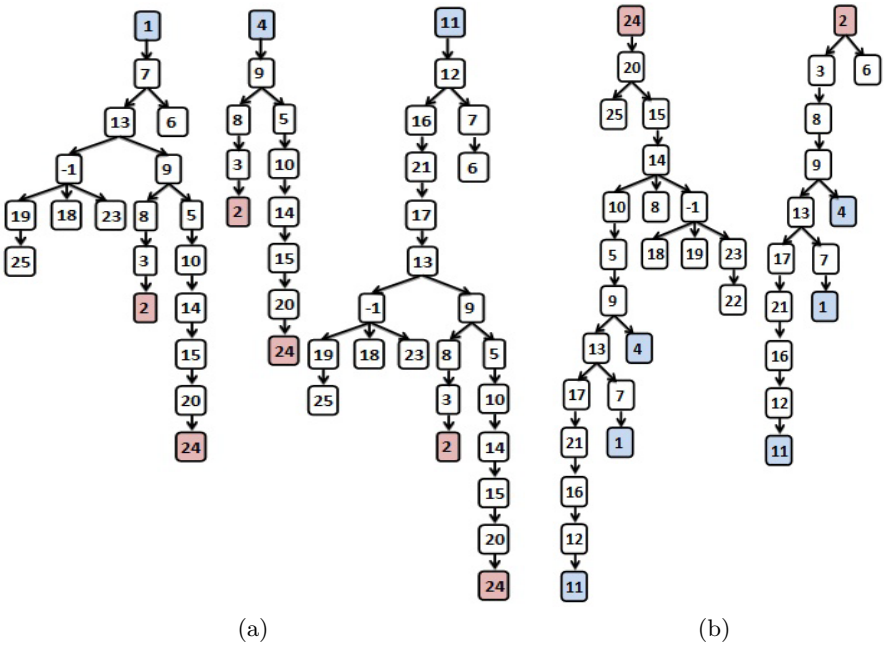


Fig. 3. Overlapping trees formed (a) regional minima; (b) regional maxima (Each node(box) of the tree shows pixelindex PI, node with value -1 is used to connect all equal nodes)

It is well known that for large constraint graph with high branching factors, 8 in our case, the complexity of constraints satisfaction significantly reduces if the graph could be reducible to a tree based on information about the problem domain [14]. Since multiple directed paths could connect a pair of nodes therefore multiple spanning trees are possible. Each of these branches enforces constraint propagation in a strongly-k-consistent manner [15,16], k being the length of the branch. A large value of k enables the encoding of a large contrast interval, and also enables the exclusion of a subset of inconsistent domain values. Therefore, the path with the largest ultrametric value of k is retained. This hierarchical representation thus formed helps in approximating contrast not onlylarge in the local 3×3 LBP window but also some contrast information in the propagation extent of that extremum. Algorithm 1 describes the formation of trees based on superiority constraints emanating from regional minima. Fig. 3(a) and 3(b) show trees formed from regional minima and maxima.

4 The Inversion Algorithm

The proposed data structure presents a new representation of an image in terms of propagation extents of regional extrema. Algorithm 1 ensures constraint propagation in a strongly k-consistent manner from the minimum to pixels defining

the contour of its propagation extent. This translates to domain shrinkage of pixels' to values consistent with the constraints. Every combination of those consistent values generates a different variations of the image with same local structure, but different appearance in terms of luminance and contrast.

Since no constraints are applied at the root of all minima trees, their possible values could lie in the interval $[0, 255 - (treeDepth \times step)]$. As we move down the tree hierarchy, the set of possible values shrinks by increasing the lower limit of the interval. Algorithm 2 explains how pixel intensities are estimated and how the assignment of estimates proceeds, done in two passes.

In $Pass - 1$, all minima trees are filled with monotonically increasing values. The increase in the intensity estimate at each level (the *step*) lies in the interval $[1 - maxStep]$ where $maxStep$ is determined by dividing the range of intensity value i.e. 256 with the depth of the deeptest tree for a particular image. The $Pass - 1$ reconstructed image thus has all minima assigned 0. The LBP codes at this stage are satisfied to 100% but it is possible to have a better estimate in terms of local contrast by noting that nodes that form part of multiple propagation extents are assigned values propagating from each of their sources, finally retaining the largest one. Any successor down the branch is therefore

Algorithm 2. IMAGE RECONSTRUCTION

Input: A set of trees $treeList = tree_1, tree_2, \ldots, tree_n$ that observes superiority constraints.
Output: Reconstructed image $reconsImg[m, n]$

initialize $reconsImg[m, n] \leftarrow -1$ and $step \leftarrow$ floor(256/depth of the deepest tree in $treeList$)
for $k \leftarrow 1$ ***to*** $size(treeList)$ **do**
  for ***each*** *node in* $tree[k]$ **do**
    i, j are 2D indices of pixelindex $nodeValue$
    if $(reconsImg[i, j] < ((nodeLevel - 1) * step))$ **then**
      $reconsimg[i, j]$ = ($nodeLevel$ - 1) * $step$
for $k \leftarrow 1$ ***to*** $size(treeList)$ **do**
  $nodeId \leftarrow$ nodeId of deepest node of tree[k]
  $pixelInd \leftarrow$ nodeValue of deepest node of tree[k]
  i, j are 2D indices of pixelindex $pixelInd$
  $maxValue \leftarrow$ reconsImg[i,j]
  $treeDepth \leftarrow$ nodeLevel of corresponding nodeId
  $updRootValue \leftarrow$ maxValue - ((treeDepth-1) * step)
  $localStep \leftarrow$ (255 - maxValue)/treeDepth
  for ***each*** *node in* $tree[k]$ **do**
    i, j are 2D indices of pixelindex $nodeValue$
    if $(reconsImg[i, j] < (updRootValue + (nodeLevel - 1) * localStep))$ **then**
      $reconsImg[i, j] = updRootValue + (nodeLevel - 1) * localStep$

consistent with our longest ultrametric requirement but the ancestors need updating. Therefore, a second pass of node intensity estimation is performed.

$Pass-2$ of the reconstruction updates the estimates back to the roots of the trees. In order to make the estimates consistent with local contrast, we adjust the values of all nodes using a 'local' step size that is determined using the depth of a tree. This removes the loss in local-contrast-preservation induced by the longest-ultrametric update having only been made on parts of the tree in $Pass-1$.

5 Results and Discussions

The algorithms described above ensure that the descriptor-to-image inversion is consistent with the constraints emerging from the mutual arrangement of the descriptor patches. In this section, we qualtify the goodness-of-inversion. Results are separately reported for both kind of propagation sources, minima and maxima. However, since the average of two constraints-consistent values lies in the interval of consistent values, we also report the measure on the averaged image. To show that $Pass-2$ of the inversion algorithm improves results over $Pass-1$, measures are shown on both passes.

Since the LBP is classically considered a descriptor of image texture, we work with the SIPI texture dataset, which includes images from the earlier Brodatz dataset as well. However, since our method is not limited to texture description because superiority or inferiority constraints occur due to the nature of the descriptor not the images containing significant texture, we also use the popular BSD500 dataset, containing images in the wild.

We can observe from Table 1 that the estimated images make visual sense. Edges and object silhouettes, where present, are discernable, the image content structure is respected i.e. the estimated image is dark or light where the original image was dark or light. In the Lenna image, one can observe the gradual shading on the hat, consistent with the original. But the quality of estimates need to be quantified to answer two questions: 1. whether the intensity estimated at a particular pixel matches the one in the original image, and 2. whether the local appearance of the image i.e. the contrast and luminance along the two axis is respected.

Competing work [9] uses the **Mean Absolute Error** between the original and the estimated pixel intensities to explain the efficacy of their method of inversion, so we use the MAE too for benchmarking. The MAE is given by:

$$\mathrm{MAE}(\bar{R},\bar{O}) = \frac{\sum_{i=1}^{m}\sum_{i=1}^{n}\bar{R}-\bar{O}}{m\times n} \tag{6}$$

where $\bar{X} = \frac{X-min(X)}{max(X)-min(X)}$ for an image with size $m \times n$, normalizes image X to lie in the interval $[0,1]$.

Table 1. Reconstructed images using LBP codes - first 3 rows show *Pass*-1 reconstruction (*minima* , *maxima* and *average* respectively), next 3 rows show *Pass*-2 reconstruction and last row contains sample original images.

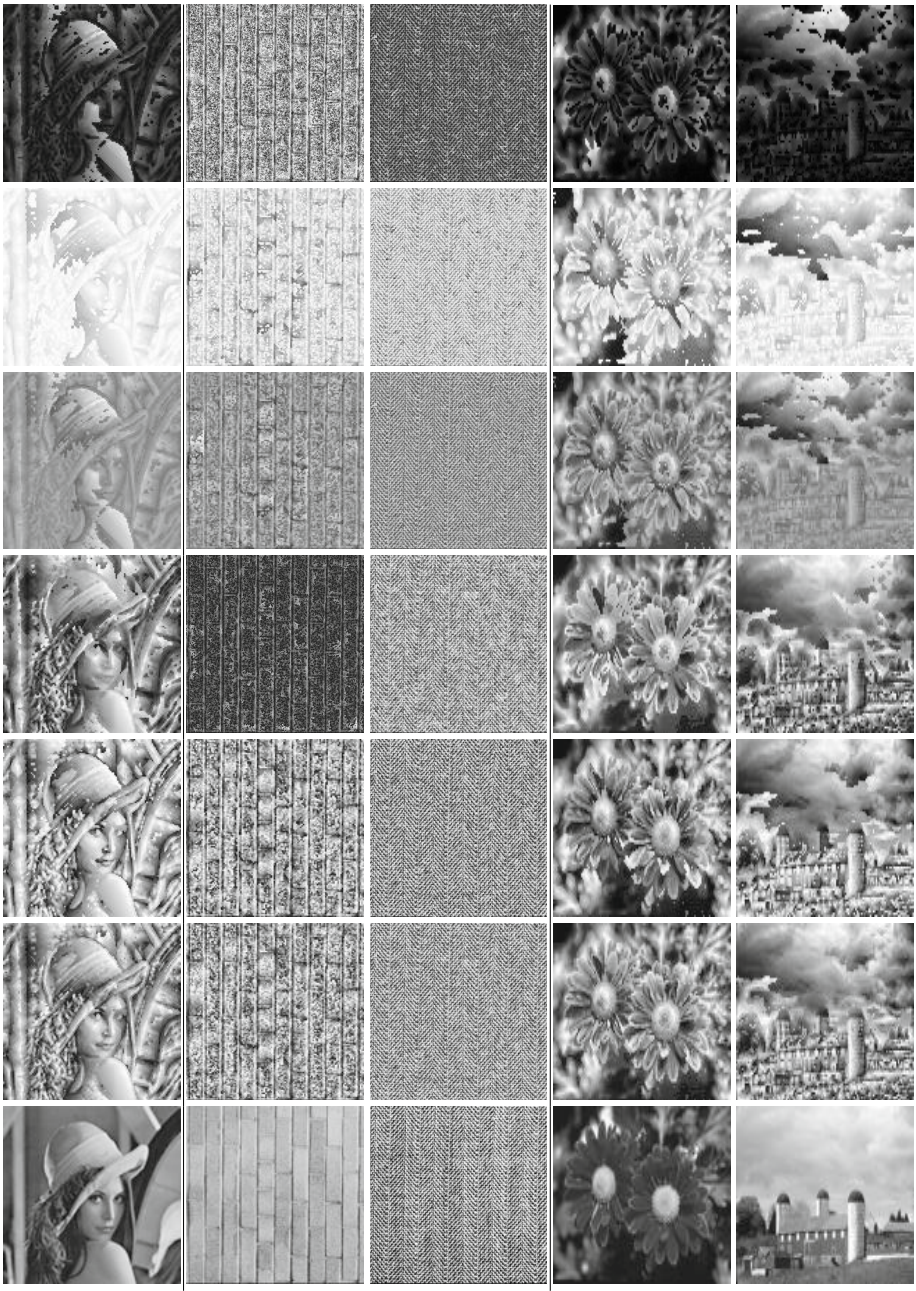


Table 2 shows the MAE computed on the SIPI and BSD500 dataset. The results computed on the two passes of the inversion algorithm show that the MAE calculated on images from $Pass-1$ is higher than the $Pass-2$ estimates. Averaging is likely to improve a measure by spreading the image intensity histogram and improving contrast. Also, the average image MAE diminishes because a generic image is not likely to have its histogram naturally skewed to either toward 0 or toward 255, under the application of the central limit theorem. MAE on average images is found to be 15% and 18% on the SIPI and BSD500 dataset respectively.

A truer measure for inversion quality would take into account the contrast of image localities, partly because the MAE is very sensitive to appearance changes while local descriptors are quasi-invariant to them, but mostly because we would be interested to know whether any of the contrast, which the LBP operator lost, has been recovered. Structural similarity measures give access to that information, and we use the **Structural Texture SIMilarity measure (STSIM)**, proposed by [17]. This measure incorporates two appearance terms, *luminance* and *contrast*, which measure the local mean and standard deviation of an image using Eq. 7, where R is the estimated image, O is the original image, l is the luminance and c is the contrast comparison term and $C_{R,O}$ $(0,1)$ and $(1,0)$ are the first order auto correlation terms computed in the x and y directions.

$$STSIM(R,O) = (l_{R,O})^{1/4} \times (c_{R,O})^{1/4} \times (C_{R,O}(0,1))^{1/4} \times (C_{R,O}(1,0))^{1/4} \quad (7)$$

Table 2 shows the $STSIM$, *luminance* and *contrast* computed for both datasets. Results of $STSIM$ show that the reconstructed image retains 92% structural similarity with the original image. Although, *contrast* doesnot show remarkable change between the two passes, however, *luminance* is improved by approximately 10% in $Pass-2$ particularly in the image reconstructed through the *maxima* tree generation, because intensity estimates are now closer to the original intensities.

Table 2. Results calculated on $SIPITexture$ and $BSD500$ dataset

	$SIPI$			$BSD500$		
	minima	*maxima*	*average*	*minima*	*maxima*	*average*
Pass-1						
Luminance	0.929	0.785	0.971	0.816	0.768	0.952
Contrast	0.886	0.886	0.886	0.941	0.932	0.902
STSIM	0.909	0.869	0.919	0.891	0.874	0.912
MAE	0.217	0.305	0.167	0.374	0.227	0.179
Pass-2						
Luminance	0.976	0.974	0.976	0.952	0.954	0.952
Contrast	0.854	0.868	0.867	0.928	0.928	0.931
STSIM	0.913	0.919	0.919	0.918	0.916	0.917
MAE	0.155	0.156	0.149	0.197	0.179	0.180

Another deduction which could be made from Table 2 is that both MAE and STSIM for inverted images after $Pass-2$ through either minima or maxima trees converge to those of the average case. Therefore, it would be computationally interesting to perform the inversion using either of the two kind of trees.

6 Conclusions

It is instructive to understand what information is lost when local descriptions for instance LBP codes are generated, and whether and how that information could be estimated. Descriptors such as FREAK [18] or BRIEF [19] explicitly encode geometric information, while we derive it by observing the constraints emerging from the LBP's sliding window architecture and use it to answer the inversion problem posed in the introduction.

The underlying mechanism of overlapping sliding windows provides the leverage to argue about global image structure. It also poses the question of generalizing the inference of implied information to any sliding-window operation.

References

1. Yao, C.-H., Chen, S.-Y.: Retrieval of translated, rotated and scaled color textures. Pattern Recognition **36**(4), 913–929 (2003)
2. Wang, H., Ullah, M.M., Klaser, A., Laptev, I., Schmid, C: Evaluation of local spatio-temporal features for action recognition. In: Procedings of the British Machine Vision Conference 2009, pp. 124.1–124.11 (2009)
3. Kellokumpu, V., Zhao, G., Li, S.Z., Pietikäinen, M.: Dynamic Texture Based Gait Recognition, pp. 1000–1009 (2009)
4. Zhao, G., Chen, J., Pietikäinen, M.: An improved local descriptor and threshold learning for unsupervised dynamic texture segmentation. In: Proc. 2nd IEEE International Workshop on Machine Learning for Vision-based Motion Analysis (MLVMA 2009), Kyoto, Japan, pp. 460–467 (2009)
5. Mikolajczyk, K., Schmid, C.: A Performance Evaluation of Local Descriptors. IEEE Transaction on Pattern Analysis and Machine Intelligence **27**, 1615–1630 (2005)
6. Weinzaepfel, P., Jégou, H., Pérez, P.: Reconstructing an image from its local descriptors. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 337–344 (2011)
7. Daneshi, M., Guo, J.: Image reconstruction based on local feature descriptors (2011)
8. Wu, J., Rehg, J.M.: CENTRIST: A Visual Descriptor for Scene Categorization. IEEE Transactions on Pattern Analysis and Machine Intelligence, 1–14, December 2010
9. Waller, B.M., Nixon, M.S., Carter, J.N.: Image reconstruction from local binary patterns. In: 2013 International Conference on Signal-Image Technology & Internet-Based Systems, pp. 118–123. IEEE, December 2013
10. Gudmundsson, B., Kruse, B., Antonsson, D.: PICAP and Relational Neighborhood Processing in FIP
11. Lindahl, T:. Study of Local Binary Patterns Study of Local Binary Patterns Examensarbete utfört i medieteknik Tobias Lindahl (2007)

12. Timo, O., Pietikätinen, M., Harwood, D.: A comparative study of texture measures with classification based on featured distributions. Pattern Recognition **29**(1), 51–59 (1996)
13. Qing, X., Yang, J., Ding, S.: Texture segmentation using lbp embedded region competition. Electronic Letters on Computer Vision and Image Analysis **5**(1), 41–47 (2005)
14. Mackworth, A.: Consistency in networks of relations. Artificial Intelligence **8**(1), 99–118 (1977)
15. Freuder, C.E.: Synthesizing constraint expressions. Commun. ACM **21**(11), 958–966 (1978)
16. Freuder, E.C.: A sufficient condition for backtrack-free search. J. ACM **29**(1), 24–32 (1982)
17. Zujovic, J., Pappas, T.N., Neuhoff, D.L.: Structural similarity metrics for texture analysis and retrieval. In: 2009 16th IEEE International Conference on Image Processing (ICIP), vol. 22(7), pp. 2545–2558 (2009)
18. Ortiz, R.: Freak: Fast retina keypoint. In: Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2012, pp. 510–517. IEEE Computer Society, Washington, DC (2012)
19. Calonder, M., Lepetit, V., Strecha, C., Fua, P.: BRIEF: Binary robust independent elementary features. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV 2010, Part IV. LNCS, vol. 6314, pp. 778–792. Springer, Heidelberg (2010)