Fisher vector encoding of micro color features for (real world) jigsaw puzzles

Fabian Richter, Christian Eggert, Rainer Lienhart

Angaben zur Veröffentlichung / Publication details:

Richter, Fabian, Christian Eggert, and Rainer Lienhart. 2015. "Fisher vector encoding of micro color features for (real world) jigsaw puzzles." In *13th International Conference on Document Analysis and Recognition (ICDAR), 23 - 26 August 2015, Tunis, Tunisia*, 521–25. Piscataway, NJ: IEEE. https://doi.org/10.1109/icdar.2015.7333816.

Fisher Vector Encoding of Micro Color Features for (Real World) Jigsaw Puzzles

Fabian Richter, Christian Eggert, Rainer Lienhart
Multimedia Computing and Computer Vision Lab, University of Augsburg, Germany
{fabian.richter, christian.eggert, rainer.lienhart}@informatik.uni-augsburg.de

***Abstract*—In this work we propose a computationally efficient yet effective color encoding scheme for the purpose of jigsaw puzzle solving. Our contribution is twofold: (i) First of all, we show that a Fisher vector encoding gives superior performance over the commonly used compatibility descriptor of stacked color values. (ii) Furthermore, we show experimentally on synthetic and real-world data that our proposed color encoding is more robust in the presence of noise, as compared to three widely used features. Due to the robustness of our proposed descriptor we also anticipate its use to yield performance improvements in other applications, e.g., for the virtual reconstruction of hand-torn documents and archaeological findings.**

I. Introduction

Automated jigsaw puzzle solving is a traditional computer vision problem that has been studied extensively in many variants. While some problem definitions include the identification and matching of shapes [1], [2], others focus on matching image content [3], [4], [5]. As the number of puzzle pieces grows, so does the need for a more robust matching strategy. In recent years, the prevalent scenario was to focus on images that have been digitally cut into many square pieces.

Such synthetically generated jigsaw puzzles however oversimplify piece matching: Pixel-correspondences between adjacent pieces are assumed to be known, and due to the puzzle's virtual nature, the pieces' boundaries are not polluted by noise. Under these premises, simple similarity metrics such as the sum-of-squared color differences (across neighboring boundary pixels) yield satisfactory results. In real-world jigsaw puzzles however, these assumptions typically do not hold. After digitization, pieces suffer from artifacts along their boundaries; also, in case of irregularly shaped jigsaw pieces, it may not be straightforward to establish exact pixel-correspondences. These problems often lead to a sharp decline in the robustness of the matching.

In this work we propose to utilize the Fisher Vector (FV) framework, which has enjoyed tremendous success in several applications in image retrieval and image classification [6], [7], [8], as well as in 3D object retrieval [9]. Besides being computationally very efficient, we found our color encoding to be very robust in the presence of noise. Despite having a compact representation, our proposed micro color feature outperforms traditional representations, which is shown experimentally on both, synthetically generated as well as real-world data.

II. Related Work

The literature on approaches to jigsaw puzzles is vast. Due to the scope of our work, we briefly recap recent approaches that rely on the pieces' content rather than shape information.

For example, Alaijan in his work [3] considered square pieces, focusing exclusively on image content. Grey-level pixel values along the boundaries are used in conjunction with a dissimilarity measure based on Dynamic Time Warping. In [4] the authors propose a probabilistic approach for solving jigsaw puzzles. Both color- and gradient-based features are being evaluated, as well as a range of different dissimilarity measures. We note however that all of their examined measures rely on known pixel-correspondences. Yang et al. [10] adopt a different probabilistic approach for solving the puzzle, using color features and a particle filter.

Sholomon et al. [11] recently proposed a genetic algorithm-based approach for reconstruction. The authors use the sum-of-squared differences measure that was shown to be the most discriminative in [4]. For the sake of completeness we hence use this measure as our first baseline. Improved compatibility measures have been introduced by Pomeranz et al. [12], who utilize a Taylor expansion to extrapolate expected pixel values across patch boundaries. Finally, the authors of [5] propose the Mahalanobis Gradient Compatibility (MGC) measure, which we use as a second baseline in our experiments.

III. Fundamentals of the Fisher Vector Encoding

In the following we give a brief introduction to the Fisher Vector framework, as it forms the basis of our proposed micro feature. The Fisher Vector (FV) [13] provides a mechanism by which a variable number of features can be incorporated into a fixed-length signature, while retaining most of the discriminative power of the original features. Although the encoding incurs a loss of information – the original features cannot be recovered – it can be beneficial to encode features in this way. Normally, in order to successfully compare two feature sets directly, it is necessary to (a) establish feature correspondences, (b) define a sensible similarity measure on those features, and (c) incorporate the similarity of the individual feature pairs into a global measure of similarity.

The FV encoding provides a principled way to measure the similarity without steps (a) and (b). It is based on the assumption that the distribution of features can be modeled through a generative process, i.e., it relies on a probability density function $p(\mathbf{x}|\boldsymbol{\lambda})$ that is parametrized by $\boldsymbol{\lambda}$. Typically, a Gaussian mixture model (GMM) is used for this purpose, in which case $\boldsymbol{\lambda} = \{\pi_1, \dots \pi_K, \boldsymbol{\mu}_1, \dots \boldsymbol{\mu}_K, \Sigma_1, \dots, \Sigma_K\}$ are the parameters of K mixture components. Here the π_k's and $\boldsymbol{\mu}_k$'s denote the components' priors and means, respectively. A commonly used simplification is to only consider diagonal covariance matrices Σ_k. By applying this standard assumption

one can denote the diagonal of the matrix by $\boldsymbol{\sigma}_k^2$, which is the vector of variances.

In what follows, $\mathcal{X} = \{\boldsymbol{x}_1, \dots \boldsymbol{x}_n\}$ represents our *feature set*, which has an associated likelihood function $u(\boldsymbol{\lambda}|\mathcal{X})$ with respect to the GMM. The FV encodes the gradient $\nabla_{\boldsymbol{\lambda}'} \log u(\boldsymbol{\lambda}|\mathcal{X})$ regarding the model parameters of the log-likelihood of $\mathcal{X}$. Since the mixture priors have been shown [8] to add only little discriminative information, we exclude them from the gradient computation and only consider the gradients for $\boldsymbol{\lambda}' = \{\boldsymbol{\mu}_1, \dots \boldsymbol{\mu}_K, \boldsymbol{\sigma}_1, \dots, \boldsymbol{\sigma}_K\}$. Note that the dimensionality of the FV does not depend on the number of features in $\mathcal{X}$. That is, given that the $\boldsymbol{x}_i$'s are d-dimensional vectors, their FV encoding has a fixed dimensionality of $2K \times d$. The computational effort for encoding of the feature set $\mathcal{X}$ is $\mathcal{O}(nKd)$, since we only consider diagonal covariance matrices.

Although the FV allows to avoid the explicit definition of feature correspondences and similarity measures on individual features, one could have accomplished the same with a bag-of-words [14] (BoW) approach. However, as discussed in [8], the FV can be regarded as a generalization of BoW, and usually leads to better performances for a given feature dimensionality.

IV. Synthetic Jigsaw Puzzles

A. Piece Comparison

As the most commonly used compatibility measures we first recap the sum-of-squared color differences. We complement this first baseline with the gradient-based approach of [5]. A prerequisite for both features is that pixel-correspondences between the boundaries of two pieces are known. Finally we discuss two features without this requirement, the latter of which being our proposed micro color feature.

1) Color Compatibility: A very simple – yet widely used – approach to determine the compatibility of two square image patches is based on the sum-of-squared color differences across their adjacent boundaries. Denote by $\mathcal{P}_k$ and $\mathcal{P}_l$ two patches of size $H \times H$. Without loss of generality, we assume that piece $\mathcal{P}_k$ is positioned to the left of piece $\mathcal{P}_l$. To determine the pieces' compatibility we first extract two feature descriptors by stacking the color values along the rightmost column of $\mathcal{P}_k$ and the leftmost column of $\mathcal{P}_l$. Therefore we write:

$$\boldsymbol{\psi}(\mathcal{P}; x, y) = \big[\mathcal{P}(x, y, c)\big]_{c=1,\dots,3} \tag{1}$$

By c we refer to the 3 color channels of the CIELUV color space, x denotes a pixel column, and y is the pixel's vertical position within that column. According to the above definition, the stacking of color values along the pieces' adjacent boundaries can be formalized by $\boldsymbol{\psi}_L(\mathcal{P}_l) = \big[\boldsymbol{\psi}(\mathcal{P}_l; 1, y)\big]_{y=1,\dots,H}$, which comprises color values in the leftmost pixel column of the righthand piece[1]. Similarly, the boundary of the lefthand piece is represented by $\boldsymbol{\psi}_R(\mathcal{P}_k) = \big[\boldsymbol{\psi}(\mathcal{P}_k; H, y)\big]_{y=1,\dots,H}$. As in [4], [11] we determine the *color compatibility* of both pieces by computing the squared Euclidean distance between those two descriptors.

[1]We index both rows and columns from 1 to H.

Fig. 1. Three pages of the *bdw082010* test set [15] that were used to sample synthetic jigsaw pieces of size $H \times H$.

2) Mahalanobis Gradient Compatibility: The Mahalanobis gradient compatibility (MGC) has been proposed by Gallagher et al. [5]. The authors' key idea for the compatibility measure is to penalize changes in color gradients rather than penalizing changes in color directly. Thus the underlying assumption is that matching puzzle pieces continue the gradient across their adjacent boundaries. Besides, the authors propose to estimate the covariance between color channels to replace the Euclidean distance with the Mahalanobis distance.

3) Color Histogram: For our last baseline we extract the color histograms over $\boldsymbol{\psi}_R(\mathcal{P}_k)$ and $\boldsymbol{\psi}_L(\mathcal{P}_l)$. As compatibility measure we tried different kernel functions and number of bins, as will be evaluated in section IV-B.

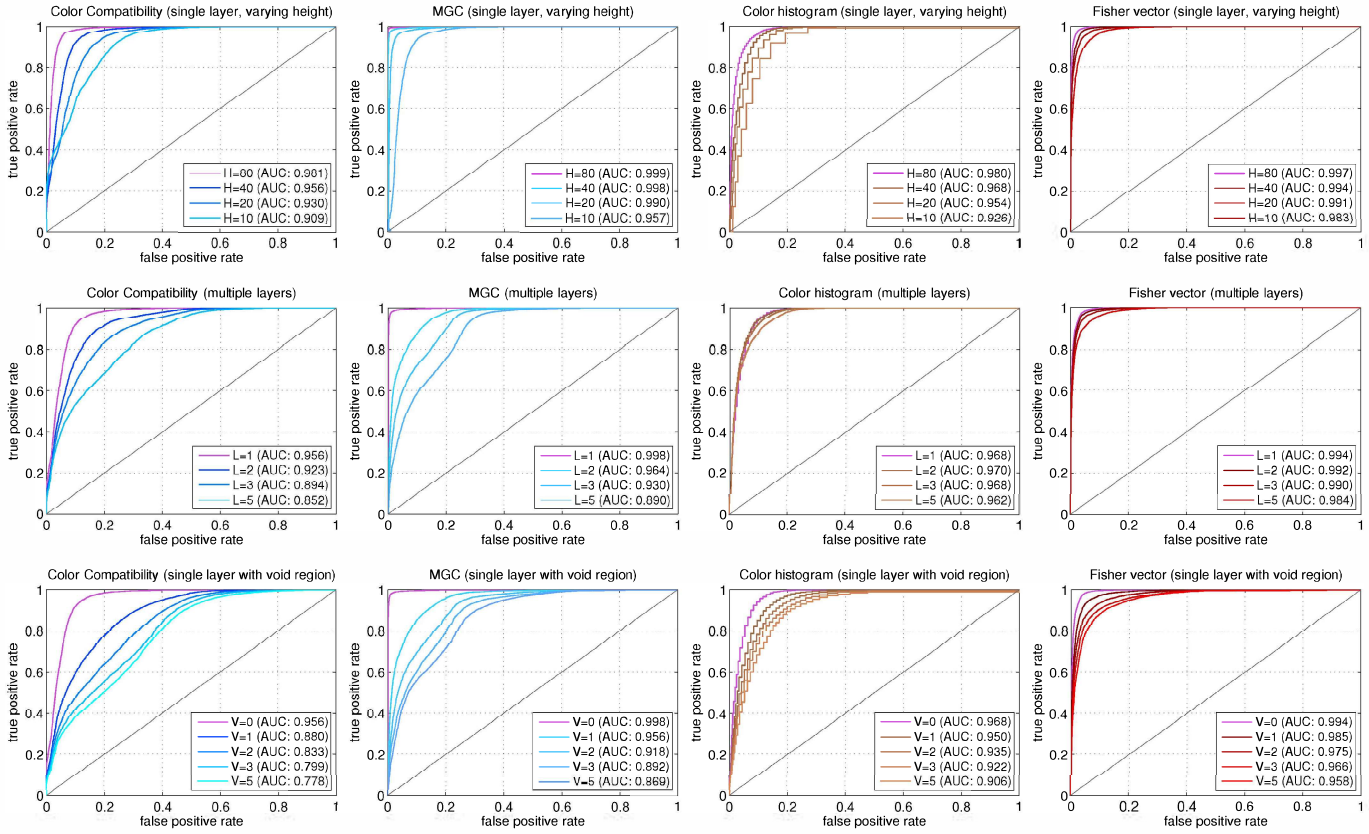
4) Fisher Vector Color Encoding: The main contribution of this work is a micro color feature that eliminates the need of establishing pixel-correspondences. Denote by $\boldsymbol{\psi}_{1\times M}(\mathcal{P}; x, y)$ a *micro patch descriptor* along a piece's boundary. Each such descriptor is the stacking of the color values on an image patch $\mathcal{P}$, at position (x, y), having a spatial extent of M pixels in height[2]. We perform a dense extraction of overlapping micro patches with 1 pixel stride, resulting in feature set $\mathcal{X}(\mathcal{P}; x) = \{\boldsymbol{\psi}_{1\times M}(\mathcal{P}; x, y) \in \mathbb{R}^{3M} \mid y = 1, \dots, H-M+1\}$. We then use the FV encoding to represent this set of local color descriptors as the concatenation of gradients relative to the log-likelihood of a Gaussian mixture model. With $\mathcal{X} = \mathcal{X}(\mathcal{P}; x)$ we write

$$\boldsymbol{\psi}_{\boldsymbol{\lambda}'}(\mathcal{P}; x) = (\mathcal{G}^{\mathcal{X}}_{\boldsymbol{\mu}_1}, \dots, \mathcal{G}^{\mathcal{X}}_{\boldsymbol{\mu}_K}, \mathcal{G}^{\mathcal{X}}_{\boldsymbol{\sigma}_1}, \dots, \mathcal{G}^{\mathcal{X}}_{\boldsymbol{\sigma}_K}), \tag{2}$$

where $\mathcal{G}^{\mathcal{X}}_{\boldsymbol{\mu}_k}$ and $\mathcal{G}^{\mathcal{X}}_{\boldsymbol{\sigma}_k}$ are the d-dimensional gradient vectors with respect to a GMM with K mixture components (see [8] for details). We use VLFeat [16] for computation of the *improved FV*, for which a power normalization is first applied in each dimension, before finally, the descriptor is ℓ_2-normalized.

By using the FV encoding, we represent an entire pixel column of each piece through statistics of the deviation of its micro patch descriptors from a generative Gaussian mixture model. Similar as for the color compatibility we extract two descriptors $\boldsymbol{\psi}_R(\mathcal{P}_k) = \boldsymbol{\psi}_{\boldsymbol{\lambda}'}(\mathcal{P}_k; H)$ and $\boldsymbol{\psi}_L(\mathcal{P}_l) = \boldsymbol{\psi}_{\boldsymbol{\lambda}'}(\mathcal{P}_l; 1)$. Typically, the similarity measure of choice between two FVs is their dot product. However, since both vectors have unit length, there is no functional difference between their dot product and their Euclidean distance, because both measures yield equivalent results up to an additive and multiplicative constant.

[2]We assume pieces to be matched along their vertical boundaries.



(a) Color Compatibility (b) Mahalanobis Gradient Compatibility (c) Color histogram (d) Fisher vector

Fig. 2. Feature comparison on the test set, organized in three groups of experiments (rows), in which we evaluate the impact of the patch height (i.e., the boundary length), a multilayer representation, and how void regions influence performance. Columns from left to right: (a) *Color compatibility*, (b) *Mahalanobis gradient compatibility* (MGC), (c) *color histogram* ($B = 32$ bins per channel), (d) *Fisher vector encoding on micro patches*, ($M = 1$, and $K = 4$ Gaussian mixture components). The feature parameters for the FV (M, K) and the color histogram (B) were determined on the validation set (see text for details).

B. Evaluation

We now quantitatively compare the feature performances on synthetically generated jigsaw pieces.

1) Dataset: We conduct experiments on the *bdw082010* dataset [15], which consists of 96 magazine pages that have been digitized using an off-the-shelf scanner. All pages have been partitioned into three disjoint sets: {train} (32 pages), {val} (16 pages), and {test} (48 pages). The document pages feature plain text, natural images, and concept art among other contents, and the datasplits were chosen deliberately to contain all the different contents (see figure 1 for examples).

We then extract positive and negative examples from 10 randomly rotated versions of each page so as to avoid any bias for predominant horizontal or vertical image features. Positive examples are composed of two adjacent patches, e.g., $(\mathcal{P}_k, \mathcal{P}_l)$, which are sampled randomly from individual pages. Negative examples on the other hand are composed of two patches that are sampled from different pages. For each example, patch $\mathcal{P}_k$ is always positioned to the left of $\mathcal{P}_l$.

2) Methodology: We compute the feature compatibilites for all positive and negative examples, and plot their ROC curve from the resulting list. The area-under-curve (AUC) is used as the measure for our quantitative comparison.

3) GMM Training and Feature Parameters: The GMM was trained using 10^6 micropatches that were randomly sampled from {train}. We used the VLFeat [16] library for training, and the covariance matrices were restricted to diagonal form.

Parameters for the color histogram (number of bins, B) and the FV (size of micro patches M, number of components in the GMM, K) were determined in separate experiments on {val}. To decide on a compatibility measure for the color histograms, we conducted experiments using a linear kernel, normalized- and unnormalized histogram intersection, and Chi-square. The number of bins (per color channel) was chosen from $\{2, 4, 8, 16, 32, 64\}$. We found that $B = 32$, in conjunction with the unnormalized histogram intersection, worked the best and is hence used for all experiments on {test}.

For the FV we evaluated micro patches of varying size, i.e., $M \in \{1, 2, 3, 5, 10\}$, in conjunction with different numbers of mixture components K for the GMM. We chose K to range from 1 to 16 in powers of 2. While most of the 25 experiments provided comparable AUC values, $M = 1$, $K = 4$ produced the best results. Notice that for this parameter configuration, micro patches correspond to individual pixels, hence the micro patch descriptors $\psi_{1\times M}$ are only $d = 3$ dimensional. We want to emphasize that the FV, with $M = 1, K = 4$, thus yields a compact 24-dimensional color descriptor.

4) Experiments: We organized experiments on the test set in the following three groups, in which we compare the feature performances in different settings:

(1) Varying Patch Height (H): In our first set of experiments we evaluate the features' robustness depending on patch height H. As one would expect, longer pixel boundaries lead to a higher discriminativeness, as is also reflected by the plots in the first row of figure 2. Despite its more compact representation, the FV performs on a par with the MGC, and both features outperform the color compatibility and color histograms. Since the number of available pixels in practice is often limited (e.g., due to an increasing number of puzzle pieces), we assume a fixed height of $H = 40$ for the remainder experiments.

(2) Mutliple Layers (L): Next we make use of a multilayer pixel band closest to the piece's outer boundary. As dictated by L, we now consider up to 5 layers for pixelwise comparison, as well as for feature extraction (for color histogram, MGC, and FV), respectively.[3] As can be seen in figure 2, this change has a detrimental effect on all features but the color histogram. However, we also note that the performance degradation for the FV is not as pronounced as for the pixelwise comparison and the MGC.

(3) Void Region (V): With the plots in the last row we strive to evaluate the "robustness" of each individual layer. Therefore we plot ROC curves for single layers that are offset from the outer boundary by V pixels. As can be seen in this series of plots, the features' performances generally decline for pixel layers that are more distant from the piece boundary. One can conclude from the plots however that the FV is the most reliable feature when close-by pixel layers (e.g., immediate neighbors across the boundaries) are not readily available.

In summary, for real-world jigsaw puzzles one would anticipate the FV to outperform the other three features, because in this scenario "void regions" become inevitable. Considering the tearing process, be it by hand or a shredding machine, it becomes apparent that pieces are likely to suffer from material loss along their boundaries. Less obvious yet very important is that the tearing not only proceeds along a paper's surface, but also through the thickness of the document. Consequently, some of the outmost boundary pixels may be non-informative or even detrimental for matching and feature extraction.

We want to emphasize that our 24-dimensional FV offers a more memory efficient representation than the other features: For $H = 40$, the color compatibility and the MGC respectively store 120 color values and color gradients. Note that even a sparse color histogram requires to encode at most 40 non-empty bins.

V. Real-World Jigsaw Puzzles

In this section we discuss our approach to feature extraction on real-world jigsaw puzzles. We also give a quantitative comparison of the features' performances regarding the experiment on the synthetic examples.

[3]For the MGC we use gradients of all L layers for the computation of the covariance matrices.

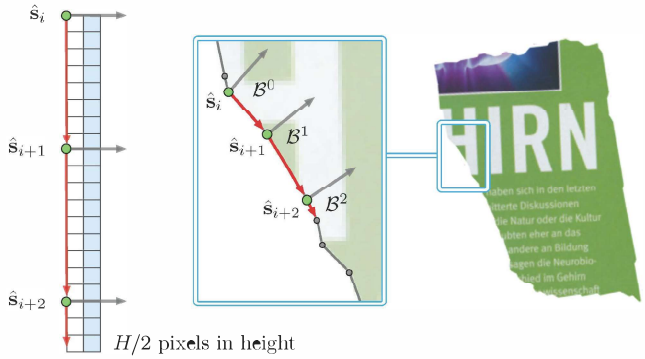


Fig. 3. LCE is used to extract a rectified pixel layer ($L = 1$) as representation of point $\hat{\boldsymbol{s}}_i$ (light blue column). In this example, due to the insufficient length of the first line segment, additional support points are needed to obtain $H/2$ many rectified pixels. Source pixels stem from the local coordinate systems spanned by $\mathcal{B}^0$, $\mathcal{B}^1$ and $\mathcal{B}^2$. Embedded pixels in white color stem from a void region of size $V = 1$ and are omitted from the representation. Also note that the remaining $H/2$ pixels in clockwise direction are not shown here.

A. Hand-Torn Document Pieces

Same as for the experiments on square jigsaw pieces we use to the *bdw082010* dataset [15]. The dataset provides a collection of hand-torn pieces for each document page. Those pieces vary in size and can have arbitrary shape. For this reason, the color compatibility and the MGC are not directly applicable for a piece-wise comparison, as both require known pixel-correspondences. In order to provide a fair comparison between all four features we thus resort to the use of contour subsegments of fixed length.

A common representation of a piece's outer boundary is a closed polygonal curve that separates its foreground from its background. A polygonal curve is defined by a sequence of n support points $\hat{S} = \{\hat{\boldsymbol{s}}_i\}_{i=1}^{n}$. Each pair of consecutive points along the polygonal curve encloses one line segment that provides a locally linear approximation of the piece's exact boundary. Since those line segments typically vary in length, we next explain our approach to extract a fixed-size rectangular image patch comprising the outer pixel layers around a given support point.

B. Local Coordinate Embedding (LCE)

To obtain a rectified contour representation of fixed length we introduce a method dubbed *Local Coordinate Embedding* (LCE). The purpose of the LCE is to align pixels within close proximity to the boundary into a rectangular image patch. In analogy with synthetic pieces we further want the pixels to be laid out orthogonally to their boundary, i.e., its piecewise linear approximation. Consider the illustration in figure 3. To embed the piece's pixels in the vicinity of support point $\hat{\boldsymbol{s}}_i$, we always consider two subsequent points $\left\{(\hat{\boldsymbol{s}}_{i+k}, \hat{\boldsymbol{s}}_{i+k+1})\right\}_{k\geq 0}$ at a time.

Denote by $\boldsymbol{v}_1^k = \hat{\boldsymbol{s}}_{i+k+1} - \hat{\boldsymbol{s}}_{i+k}$ the direction vector which is associated with the line segment enclosed by the k-th tuple. Based on $\boldsymbol{v}_1^k$ we then choose $\boldsymbol{v}_2^k$ as the orthogonal vector being directed towards the inside of the piece. Intuitively, those two vectors define an orthogonal basis $\mathcal{B}^k$ and span a

local coordinate system. We embed boundary pixels from that coordinate system into a straight pixel column (highlighted in light blue in the figure). By aligning $\boldsymbol{v}_1^k$ and $\boldsymbol{v}_2^k$ with the image axes, we implicitly change the basis from $\mathcal{B}^k$ to an orthogonal basis of the local image coordinate system. Pixel intensities in the resulting rectified layer are computed from the source image using bilinear interpolation. Note that in the illustration in figure 3, a single layer is embedded ($L = 1$), which is offset by a one pixel void region ($V = 1$) from the boundary, in direction of $\boldsymbol{v}_2^k$.

Since line segments vary in length, so does the number of pixels that can be embedded along direction $\boldsymbol{v}_1^k$. In order to obtain a rectified patch of fixed height H, we thus adaptively adjust k (i.e., the number of local coordinate embeddings), until at least $H/2$ pixels have been stacked. Starting from $\hat{\boldsymbol{s}}_i$, the procedure is once applied for clockwise- and counterclockwise direction. The two resulting patches are each truncated to $H/2$ pixels, and finally become stacked to define the rectified image patch of size $H \times L$.

C. Evaluation

1) Ground Truth: After all pieces have been digitized and preprocessed, individual document pages have been reassembled manually using an annotation tool. Only adjacent point-pairs between neighboring pieces are considered to be positive examples throughout our experiments. To obtain a collection of negative examples we once again use random piece-pairs from different pages. For each such pair we sample a pair of support points that then defines a negative example.

2) Experiments: In our real-world experiments we use the same feature parameters that were previously determined on {val} (see section IV-B3), using synthetic jigsaw pieces. Also we re-use the Gaussian mixture model that was obtained from the synthetic training examples.

Parameters	AUC on the test set			
(L,V)	Color compat.	MGC [5]	Color hist.	Fisher vector (FV)
(1,0)	0.778	0.799	0.692	0.805
(1,1)	**0.806**	0.882	0.820	0.919
(1,2)	0.796	**0.900**	0.873	0.950
(2,2)	0.790	0.898	0.887	**0.951**
(3,2)	0.785	0.896	0.893	0.950
(5,2)	0.782	0.894	**0.898**	0.949

TABLE I. COMPARISON OF FEATURE PERFORMANCES, DEPENDING ON THE NUMBER OF LAYERS (L), AND VOID REGION SIZE (V) IN PIXELS. THE BEST PERFORMANCE FOR EACH FEATURE IS MARKED BOLD.

In table I we summarize the results of our experiments on {test}. Two important observations can be made: First of all, despite having an "implicit void region" due to material loss, adding an explicit 2 pixel void region ($V = 2$) improves the classification performance (except for the color compatibility, where $V = 1$ works best). We attribute this to the fact that the outmost pixel layer is often contaminated by noise, e.g., by pixels that stem from the scan background. Second, we observe that the pixelwise comparison is outperformed by color histograms and the MGC, yet the latter requires only a single instead of five pixel layers for equivalent performance. Finally, in direct comparison to the MGC with $L = 1$ and $V = 2$, we observe that the Fisher vector achieves a flat 5% increase in AUC. This is quite remarkable, because its representation is more compact compared to the two features based on pixel-correspondences.

VI. CONCLUSION

In this work we proposed a color-based Fisher vector encoding for solving jigsaw puzzles. We chose three common baselines for comparison and conducted experiments on both synthetic- and real-world data. While on synthetic data our proposed feature offers essentially the same performance as the state-of-the art, it provides a substantial performance gain on real-world pieces. Due to its low memory requirements and its robustness, we deem our encoding scheme to be particularly useful in practical applications.

REFERENCES

[1] H. Freeman and L. Garder, "Apictorial jigsaw puzzles: The computer solution of a problem in pattern recognition," *IEEE Trans. on Electronic Computers*, vol. EC-13, no. 2, pp. 118–127, 1964.

[2] R. W. Webster, P. S. LaFollette, and R. L. Stafford, "Isthmus critical points for solving jigsaw puzzles in computer vision," *IEEE Trans. on Systems, Man and Cybernetics*, vol. 21, no. 5, pp. 1271–1278, 1991.

[3] N. Alajlan, "Solving square jigsaw puzzles using dynamic programming and the hungarian procedure," *American Journal of Applied Sciences*, no. 6, pp. 1941–1947, 2009.

[4] T. S. Cho, S. Avidan, and W. T. Freeman, "A probabilistic image jigsaw puzzle solver," in *IEEE Conf. on Computer Vision and Pattern Recognition*, 2010, pp. 183–190.

[5] A. Gallagher, "Jigsaw puzzles with pieces of unknown orientation," in *IEEE Conf. on Computer Vision and Pattern Recognition*, 2012, pp. 382–389.

[6] G. Csurka and F. Perronnin, "Fisher vectors: Beyond bag-of-visual-words image representations," in *Computer Vision, Imaging and Computer Graphics. Theory and Applications*, vol. 229, 2011, pp. 28–42.

[7] F. Perronnin, L. Yan, J. Sanchez, and H. Poirier, "Large-scale image retrieval with compressed fisher vectors," in *IEEE Conf. on Computer Vision and Pattern Recognition*, 2010, pp. 3384–3391.

[8] J. Sanchez, F. Perronnin, T. Mensink, and J. Verbeek, "Image classification with the fisher vector: Theory and practice," *Int. Journal of Computer Vision*, vol. 105, no. 3, pp. 222–245, 2013.

[9] M. A. Savelonas, I. Pratikakis, and K. Sfikas, "Fisher Encoding of Adaptive Fast Persistent Feature Histograms for Partial Retrieval of 3D Pottery Objects," in *Eurographics Workshop on 3D Object Retrieval*. The Eurographics Association, 2014.

[10] X. Yang, N. Adluru, and L. J. Latecki, "Particle filter with state permutations for solving image jigsaw puzzles," in *IEEE Conf. on Computer Vision and Pattern Recognition*, 2011, pp. 2873–2880.

[11] D. Sholomon, O. David, and N. S. Netanyahu, "A genetic algorithm-based solver for very large jigsaw puzzles," in *IEEE Conf. on Computer Vision and Pattern Recognition*, 2013, pp. 1767–1774.

[12] D. Pomeranz, M. Shemesh, and O. Ben-Shahar, "A fully automated greedy square jigsaw puzzle solver," in *IEEE Conf. on Computer Vision and Pattern Recognition*, 2011, pp. 9–16.

[13] T. Jaakkola and D. Haussler, "Exploiting generative models in discriminative classifiers," in *Advances in Neural Information Processing Systems II*, 1998, pp. 487–493.

[14] J. Sivic and A. Zisserman, "Video Google: A text retrieval approach to object matching in videos," in *Int. Conf. on Computer Vision*, vol. 2, 2003, pp. 1470–1477.

[15] F. Richter, C. X. Ries, N. Cebron, and R. Lienhart, "Learning to reassemble shredded documents," *IEEE Trans. on Multimedia*, vol. 15, no. 3, pp. 582–593, 2012.

[16] A. Vedaldi and B. Fulkerson, "VLFeat: An open and portable library of computer vision algorithms," "http://www.vlfeat.org/", 2008.