Robust Solvers for Square Jigsaw Puzzles

Debajyoti Mondal, Yang Wang and Stephane Durocher
Department of Computer Science
University of Manitoba
Winnipeg, Manitoba, Canada
Email: {jyoti,ywang,durocher}@cs.umanitoba.ca

***Abstract*—A jigsaw puzzle solver reconstructs the original image from a given collection of non-overlapping image fragments using their color and shape information. In this paper we introduce new techniques for solving square jigsaw puzzles (with no prior knowledge of the initial image) that improves the accuracy of the state-of-the-art jigsaw puzzle solvers. While the current puzzle solving techniques are based on finding enhanced compatibility metrics across piece boundaries, we combine the existing techniques to achieve higher accuracy and robustness, i.e., our solver outperforms the known solvers even when the piece boundaries are imprecise. Unlike the most successful puzzle solvers that use greedy pairwise compatibility metrics among puzzle boundaries, we incorporate global information that enhances performance. As a step towards the future goal of developing an automated assembler for real-life corrupted image fragments or shredded documents, we examine puzzles that are corrupted by noise. Our proposed compatibility metrics shows robustness even in such scenarios.**

***Keywords*-jigsaw puzzles; image reconstruction**

Figure 1. Top row illustrates a puzzle under Gaussian noise, and the bottom row shows a puzzle where one pixel row/column has been cropped from piece boundaries. (left) original image (middle) constructed puzzle (right) solved image using our technique.

I. Introduction

Jigsaw puzzle solving has been one of the most popular fun activities among children and adults for centuries. Original jigsaw puzzles were created by painting a picture on a flat, rectangular piece of wood, and then cutting that picture into small pieces. Over time, the natural appeal of jigsaw puzzles and their applications in many scientific problems (e.g., reassembling fossil/archaeological relics [4], [5] or shredded documents [16], [13], molecular docking problem for drug design [10], DNA/RNA modeling [15], and image base CAPTCHA construction [9]) have motivated many efforts in solving jigsaw puzzles using computer vision techniques. We now have several variations of jigsaw puzzles, such as spherical jigsaw puzzles, 3-dimensional jigsaw puzzles, jigsaw puzzles without pictures, and so on. Recently, Yang et al. [24], Pomeranz et al. [20] and Gallagher [8] have examined jigsaw puzzles with square pieces that contain real-world images. Most of these approaches focus on computing some compatibility metrics between pairs of puzzle pieces, and then solving the jigsaw puzzle by reassembling the pieces together based on these compatibility metrics.

In this paper, we introduce new approaches for solving square jigsaw puzzles. We assume that the orientations of the puzzle pieces are known, but their positions are unknown (i.e., Type 1 jigsaw puzzles of [8]). Similar to Gallagher [8], we do not assume any prior knowledge of the ground truth image.

Our key contribution is to introduce new approaches for measuring the compatibility between two puzzle pieces. Our experimental results demonstrate that our proposed compatibility metrics outperform the current state-of-the-art approaches.

Previous work in solving jigsaw puzzles assumes computer-generated puzzle pieces. Unfortunately this assumption often does not hold in real-world scenarios. For real-world applications, some rows/columns along the boundaries of the puzzle pieces may be missing (e.g., when puzzle pieces are cut out of paper and subsequently scanned), or the image may be corrupted by noise (e.g., old, squeezed, torn-apart images). In this paper, we evaluate and compare different compatibility metrics when the puzzle pieces are imperfect (e.g. with cropped boundaries or noise).

The rest of the paper is organized as follows. Section II reviews some previous work related to solving jigsaw puzzles. In Section III, we briefly describe some of the most successful compatibility metrics for measuring the compatibility between puzzle pieces in the literature. In Section IV, we introduce our new approaches for measuring the compatibility between two puzzle pieces. We evaluate our new approaches in Section V and conclude in Section VI.

II. Previous Work

Early attempts on developing jigsaw puzzle solvers were based on using shape information for comparing affinity between pairs of interlocking puzzle pieces. In 1964, Freeman and Gardner [7] first examined the problem of solving jigsaw puzzles using the shapes of the pieces where the pieces were uniformly gray. Radack and Badler [21] used curvature maxima and minima of the piece boundaries, Altman [2] used a string matching technique depending on shape information, and Wolfson et al. [23] used combinatorial optimization techniques for curve matching.

Besides the shape information (e.g., convexity/curvature analysis of the piece boundaries), Kosiba et al. [12] compared color sampling windows at regular intervals along the contours of both pieces. Chung et al. [4] examined RGB values along with hue and saturation. Since then there have been several attempts to improve accuracy using various similarity measures based on shape and color [1], [14], [22], [25], and finally in 2008, Nielsen et al. [17] developed a solver using image features and shape information that could solve computer-generated jigsaw puzzles with around 320 puzzle pieces.

Color information of jigsaw pieces has been an amazingly useful tool for solving jigsaw puzzles with images, where all pieces are of the same square shape [1], [3], [8], [24], [20]. Some of these solvers can provide elegant solutions to puzzles with approximately one thousand pieces [8], [20]. Cho et al. [3] evaluated several patch compatibility metrics such as similarity of feature vectors, image statistics, or color information along boundaries. They found that the dissimilarity-based compatibility, i.e., measuring the sum-of-squared color difference along the boundaries, gives the best results. Yang et al. [24] used similar compatibility measure in LAB color space. Pomeranz et al. [20] examined several L_p norms that makes the dissimilarity measure most discriminative. Gallagher [8] proposed a new compatibility measure called Mahalanobis gradient compatibility based on the similarity in intensity gradients that outperformed the sum-of-squared color difference strategy by a large margin (even when the rotations of the puzzle pieces are unknown). Our puzzle solving technique combines the Mahalanobis gradient compatibility and the color-dissimilarity based measures.

A successful puzzle solver not only requires a good compatibility measure, but also an effective reassembly technique to put the pieces together based on the compatibility measure. It has been shown that the piece reassembly is an NP-hard problem when there is uncertainty in piece compatibility [6], so we can only expect an approximate or greedy solution. Most of the initial solvers formulated the reassembly process as a TSP-problem [4]. Alajlan [1] used Hungarian assignment to find neighboring pieces. The square jigsaw solvers of Cho et al. [3] and Yang et al. [24] used a loopy belief propagation and a particle filter inference technique, respectively. Both Pomeranz et al. [20] and Gallagher [8] used greedy reassembly. Pomeranz et al.'s reassembly was based on repeated relocation of partial solutions for finding the best fit, and Gallagher's technique was to first find a minimum spanning tree and then fill the unoccupied holes. In this paper, we use Gallagher's technique for puzzle reassembly.

III. Computing Pairwise Compatibility

In solving jigsaw, one of the most critical issues is measuring the compatibility of two puzzle pieces. In this section, we briefly describe the color difference based similarity measure [3] and the Mahalanobis gradient compatibility [8], two of the most successful approaches for measuring puzzle piece compatibility in the literature. In Section IV, we will introduce our new compatibility metrics based on these two approaches.

Sum of Squared Distance Scoring (SSD): Let x_i be a color square puzzle piece with $P \times P$ pixels. We use $x_i(h,w,c)$ to denote the value at position (h,w) of the color channel c, where $1 \leq h,w \leq P$, and $1 \leq c \leq 3$. The dissimilarity between x_i and x_j when x_i is placed on the left side of x_j is:

$$D_{LR}(x_i,x_j) = \sum_{c=1}^{3}\sum_{h=1}^{P}(x_i(h,P,c) - x_j(h,1,c))^2 \qquad (1)$$

The right-left, top-bottom and bottom-top dissimilarities are defined similarly. One can define different color based dissimilarity measures, e.g., RGB SSD, HSV SSD and LAB SSD, depending on the color space used. We have tried different color spaces and found their performances are similar. In the rest of the paper, we will use RGB SSD.

Mahalanobis Gradient Compatibility (MGC): Another very popular approach for measuring the compatibility between puzzle pieces is the Mahalanobis Gradient Compatibility (MGC) [3]. Instead of comparing the difference of RGB values like SSD, MGC compares the difference between color gradients. Let $\mu_i(c)$ be the average color difference at the cth color channel between the rightmost pair of columns of x_i, and let $G_{ijLR}(h,c)$ be the color difference at hth position of the cth color channel between the right side of x_i and the left side of x_j. Then

$$\mu_i(c) = \frac{1}{P}\sum_{h=1}^{P} x_i(h,P,c) - x_i(h,P-1,c), \qquad (2)$$

$$G_{ijLR}(h,c) = x_j(h,1,c) - x_i(h,P,c). \qquad (3)$$

Now the dissimilarity score from piece x_i to x_j is

$$G'_{ijLR} = \sum_{h=1}^{P} (G_{ijLR}(h) - \mu_i) S_i^{-1} (G_{ijLR}(h) - \mu_i)^T, \quad (4)$$

where S_i is a 3×3 covariance matrix computed from the gradient difference at the right side of x_i. The final MGC score is $C_{LR}(x_i, x_j) = G'_{ijLR} + G'_{jiRL}$. Interested readers are referred to [8] for details.

IV. Our Approach

In this section, we propose two new approaches for measuring the compatibility of two puzzle pieces. These approaches are based on SSD and MGC introduced in Section III. Our hypothesis is that SSD and MGC provide complementary information for measuring the compatibility. Our approaches aim to combine them in some sensible way.

A. Combining MGC and SSD (M+S)

Our first approach is to measure the compatibility of two puzzle pieces by combining MGC and SSD as follows:

$$C'_{LR}(x_i, x_j) = C_{LR}(x_i, x_j) \cdot (D_{LR}(x_i, x_j))^{1/q} \quad (5)$$

where q is a free parameter that can be set. We refer to this compatibility measure as M+S.

Similar to Gallagher [8], we normalize the compatibility matrix as follows. Let M be a compatibility matrix, such that $M(x_i, x_j)$ is the dissimilarity score between two distinct pieces x_i and x_j. Let X be the set of all pieces. Then the normalized matrix M' is defined as follows.

$$M'(x_i, x_j) = \frac{M(x_i, x_j)}{\min\left(\min_{\forall x \in X} M(x_i, x), \min_{\forall x \in X} M(x, x_j)\right) + \varepsilon} \quad (6)$$

Note that we do not need to define the diagonal entries of M in Eq. 6 since we never have to compare a piece with itself. We will show empirically in Section V that this normalization significantly improves the accuracy.

B. Selectively Weighted MGC (wMGC)

Our second approach is based on a further refinement of M+S. Instead of weighting every MGC score by the corresponding SSD score (i.e., M+S), here we weight the MGC scores selectively on those pairs that are likely to be misclassified by MGC score. One of our initial attempts is as follows. Given a jigsaw piece x_i, we first find the two best matches x_a, x_b (based on the normalized MGC scores in Eq. 6). If the difference between $C(x_i, x_a)$ and $C(x_i, x_b)$ is less than some fixed threshold, we assume that the MGC classification was not sufficiently confident in this scenario and hence we classified based on the weighted scores. However, this strategy could not improve the performance at all. In fact, selecting the pairs that are likely to be misclassified by MGC scores is a very challenging task.

Instead, our solution is based on the following observation. Any locally computed compatibility score is potentially misled by the fact that multiple jigsaw pieces may have the same nearest neighbor. Instead of breaking ties arbitrarily, we need some global agreement for selecting adjoining pieces. Therefore, we formulate the problem as a minimum weight bipartite matching problem as follows. Let X and Y be two copies of all puzzle pieces. Then the set X and Y correspond to the two partitions of a complete bipartite graph, and the edges are weighted by the normalized MGC scores. Our goal is to find a a bijective function ϕ that matches elements in X to elements in Y by selecting a subset of edges in the graph. Give the edge weights, the optimal matching ϕ can be efficiently found using the Hungarian algorithm [11].

Now we describe the details of how to define the edge weights in the graph. Let $n(x_i)$ be the nearest neighbor of x_i according to the MGC scores. Then for a pair of pieces x_i and x_j, we take either the MGC score or M+S score depending on whether $\phi(x_i) = n(x_i)$ or not. Hence the left-right wMGC score between x_i and x_j is

$$W_{LR}(x_i, x_j) = \begin{cases} C_{LR}(x_i, x_j) & \text{if } \phi(x_i) = n(x_i), \\ C'_{LR}(x_i, x_j) & \text{otherwise,} \end{cases} \quad (7)$$

where C and C' correspond to MGC and M+S, respectively. It is straightforward to modify Equation (7) to define compatibility measures for other arrangements of x_i and x_j, i.e., right-left, top-bottom and bottom-top.

Observe that if the matching piece $\phi(x_i)$ coincides with the nearest neighbor of x_i, then we assign the row $C_{LR}(x_i, \cdot)$ to $W_{LR}(x_i, \cdot)$; which is inspired by the higher success of MGC over SSD. Otherwise, if $\phi(x_i) \neq n(x_i)$, then we assume that the corresponding MGC scores are not confident enough. Hence the color gradient between x_i and its adjoining target piece may not be very similar. In such scenarios, we assign the row $C'_{LR}(x_i, \cdot)$ to $W_{LR}(x_i, \cdot)$.

Our experimental results in Section V show that this wMGC measure outperforms MGC or M+S, even when the puzzle pieces are corrupted, e.g., by removing rows/columns around the boundary, or by adding random noise.

V. Experimental Results

In this section, we evaluate different compatibility metrics on a benchmark dataset. We are particular interested in how these compatibility metrics perform when the puzzle pieces are corrupted, e.g., by cropping around the boundaries, or by adding random noise. We then analyze the impact of those compatibility measures on the final puzzle reassembly.

A. Classification Accuracy with Cropping

We use the classification accuracy defined in [3] to quantitatively compare different compatibility metrics. If a pair of puzzle pieces are adjacent in the original image, they should receive a high compatibility score. The classification accuracy is defined to capture this characteristic. For each image patch x_i, we find the most compatible patch x_j according to one of the compatibility metrics to be considered. We then calculate the percentage of the patches that the compatibility metric assigns the correct matches. We perform this on the same dataset used in [3] which contains 20 images.

There are several interesting conclusions we can draw from the experiments.

1) M+S outperforms MGC or SSD: Table I shows the classification accuracy for MGC, SSD and M+S, where t denotes the number of rows/columns cropped from the boundary from the piece boundaries (i.e., the cropped piece is of size $(P-2t)\times(P-2t)\times 3$), and K denotes the number of pieces per image.

Since MGC is based on the assumption that the adjoining puzzle pieces will have similar color gradients, its classification accuracy is likely to decrease more quickly than that of SSD as the boundary of the puzzle pieces become imprecise. Table I supports this assumption when the size of the pieces is small, i.e., $P=28$. For larger pieces, i.e., $P=56$, the rate of decrease in accuracy with the increase in t is smaller for MGC than for SSD. It is likely because larger boundaries help MGC to find similar gradient values in adjoining pieces. However, we observe that for higher values of t the decrease rate again become smaller for SSD.

Table I
CLASSIFICATION ACCURACY FOR VARIOUS AMOUNT OF CROPPING t AND NUMBER OF PUZZLE PIECES K ON CHO ET AL.'S DATABASE [3]. HERE M+S,i IS THE M+S SCORE WITH $q=i$. THE CELLS WHERE THE ACCURACY OF M+S IS BETTER THAN ANY OTHER MEASURE ARE SHOWN IN BOLD.

	$P=28$, $K=432$			$P=56$, $K=108$	
	$t=0$	$t=1$	$t=2$	$t=0$	$t=1$
MGC	0.9024	0.5245	0.3661	0.9556	0.7971
SSD	0.7895	0.4231	0.3004	0.9074	0.6655
Ours, M+S,4	0.9010	**0.5377**	**0.3805**	0.9532	**0.8014**
Ours, M+S,5	**0.9028**	**0.5385**	**0.3796**	0.9537	**0.8010**
Ours, M+S,6	**0.9035**	**0.5401**	**0.3786**	0.9546	**0.8022**
Ours, M+S,7	**0.9040**	**0.5399**	**0.3780**	0.9545	**0.8042**

Table I shows that M+S consistently outperforms MGC and SSD when $t\geq 1$, and also when $t=0, P=28$. A possible explanation is that SSD shows more robustness (with respect to t) than MGC. Since M+S is essentially the MGC weighed by SSD, its performance decrease with t is slower than that of MGC. Consequently, M+S outperforms MGC for larger values of t. An interesting observation that supports this explanation is the increase in t requires more weight (lower values of q) on SSD to attain higher accuracy, as shown in underline in Table I. From the perspective of error rate, M+S reduces the error rate (in comparison to MGC) by 1.6% when $t=0, P=28$, and by 3.2% when $t=1, P=28$.

2) Normalization matters: Table II shows the classification accuracy computed with normalized matrices. Although the accuracy reported is higher than that of Table I, the relative performance among different compatibility measures remains the same.

Table II
CLASSIFICATION ACCURACY WITH NORMALIZED COMPATIBILITY MATRICES, WHERE M+S,i DENOTES THE M+S SCORE WHEN $q=i$.

	$P=28$, $K=432$			$P=56$, $K=108$	
	$t=0$	$t=1$	$t=2$	$t=0$	$t=1$
MGC	0.9208	0.576	0.4082	0.9609	0.8363
SSD	0.8402	0.4714	0.3389	0.9338	0.7350
M+S,5	0.9203	0.5863	**0.4194**	0.9596	0.8341
M+S,7	**0.9209**	**0.5871**	**0.4185**	0.9601	**0.8349**
M+S,14	**0.9218**	**0.5851**	**0.4159**	0.9605	**0.8364**
M+S,16	**0.9220**	**0.5849**	**0.4151**	0.9605	**0.8368**

3) wMGC outperforms all the other metrics: Table III compares the classification accuracy of MGC, SSD, M+S and wMGC. From the perspective of error rate, wMGC reduces the error rate (in comparison to MGC) by 3.6% when $t=0, P=28$, and by 4.1% when $t=1, P=28$. In the following, we will use wMGC in our approach.

Table III
CLASSIFICATION ACCURACY FOR VARIOUS AMOUNT OF CROPPING t AND NUMBER OF PUZZLE PIECES K. $K=432$ FOR CHO ET AL.'S DATABASE [3] AND $K=540$ FOR MCGILL DATABASE [19]. HERE WMGC,i DENOTES THE WMGC SCORE WHEN $q=i$. THE CELLS WHERE THE ACCURACY OF WMGC IS BETTER THAN ANY OTHER MEASURE ARE SHOWN IN BOLD.

	$K=432$			$K=540$	
$P=28$	$t=0$	$t=1$	$t=2$	$t=0$	$t=1$
MGC	0.9208	0.5760	0.4082	0.9394	0.7483
SSD	0.8402	0.4714	0.3389	0.8159	0.4792
M+S	0.9220	0.5871	0.4194	0.9439	0.7581
wMGC,3	**0.9229**	**0.5930**	**0.4244**	**0.9440**	**0.7591**
wMGC,4	**0.9235**	**0.5935**	**0.4232**	0.9434	**0.7588**
wMGC,5	**0.9237**	**0.5924**	**0.4221**	0.9434	**0.7588**
wMGC,7	**0.9233**	**0.5905**	**0.4197**	0.9431	**0.7583**

B. Classification Accuracy with Noise

The importance of jigsaw puzzle solving is certainly not limited by the fun factor of the puzzle. It provides potential solutions for a wide range of practical applications such as reassembling archaeological relics [5] or shredded documents [16], [13]. In those applications, the ground truth images may have large amounts of noise. In the literature, previous work in solving jigsaw puzzles always assumes that the puzzle pieces are perfect. Little is known about the performance of current jigsaw puzzle solvers when the puzzle pieces are corrupted by noise. Here we provide the first experimental evaluation of popular jigsaw puzzle solving techniques with noisy puzzle pieces.

Table IV shows the classification accuracy of various compatibility measures, where the puzzle pieces are corrupted by various amounts of Gaussian noise or salt & pepper noise.

The results are very interesting. Salt & pepper noise drastically reduced the performance of SSD. On the other hand, SSD outperformed MGC under Gaussian noise. Surprisingly, wMGC shows robustness in such tough conditions, i.e., its accuracy remains close to the maximum.

Table IV
CLASSIFICATION ACCURACY UNDER GAUSSIAN NOISE OF MEAN ZERO AND VARIANCE v AND SALT & PEPPER NOISE OF NOISE DENSITY n (I.E., APPROXIMATELY n% PIXELS OF EACH IMAGE ARE AFFECTED). GAUSSIAN NOISE WAS APPLIED INDEPENDENTLY AT EACH COLOR CHANNEL.

$K = 432$	Gaussian Noise			Salt & Pepper	
$P = 28$	v=10^{-4}	v=10^{-3}	v=0.01	n=0.01	n=0.1
MGC	0.8215	0.6020	0.2927	0.8345	0.7309
SSD	0.7823	0.6210	0.3469	0.5936	0.0931
wMGC,3	**0.8322**	**0.6294**	0.3247	0.8284	0.7047
wMGC,4	**0.8315**	**0.6278**	0.3204	0.8301	0.7101
wMGC,5	**0.8316**	**0.6257**	0.3175	0.8318	0.7144
wMGC,7	**0.8302**	**0.6225**	0.3140	0.8328	0.7190

C. Results on Final Puzzle Reassembly

In this section, we apply different compatibility metrics and evaluate their performance in terms of the final puzzle reassembly. To make a fair comparison, it is important to use exactly the same puzzle reassembly method for all the compatibility metrics. We choose to use the technique in Gallagher [8] to reassemble the final puzzle. Given a compatibility metric and a set of puzzle pieces, this method first computes a pairwise scoring matrix representing the compatibility among each pair of pieces in some spatial arrange (e.g., left-right, top-bottom and so on). It then constructs a constrained spanning tree greedily from the pairwise scoring matrix while maintaining a non-overlapping planar embedding of the tree. This is followed by trimming the embedding to fix the image boundary, and filling the remaining holes greedily with the remaining pieces. Interested readers are referred to [8] for details about the puzzle reassembly technique. Here we use the source code provided by authors of [8]. The original reassembler code in [8] uses MGC as the compatibility measure. Although our natural approach could be to replace the MGC with wMGC scores, however, we found that the reassembly phase of [8] is very sensitive to consistent MGC scores. Therefore, instead of using wMGC scores directly, we use a modified MGC score based on wMGC with $q = 5$ as follows. Let $n(x_i)$ and $n'(x_i)$ be the nearest neighbor of x_i according to left-right MGC and wMGC scores, respectively. Then for every puzzle piece x_i, where $n(x_i) \neq n'(x_i)$, we swap the left-right MGC scores of $(x_i, n(x_i))$ and $(x_i, n'(x_i))$, i.e., we swap $C_{LR}(x_i, n(x_i))$ and $C_{LR}(x_i, n'(x_i))$. We update the right-left, top-bottom and bottom-top MGC scores analogously. Similar to [3], [8], [20], we consider three criteria for measuring the performance of the puzzle reassembly:

Direct comparison: for each puzzle piece, we compare its position in the assembled jigsaw with its position in the ground-truth image. The direction comparison measures the percentage of puzzle pieces that are assigned to the correct positions in a dataset.

Neighbor comparison: for each pair of puzzle pieces that are adjacent in the assembled jigsaw, we check how many of them are also adjacent in the ground-truth images. The neighbor comparison measures the percentage of such correct assigned pairs among all possible pairs in a dataset.

Perfect reconstruction: this measures the numbers of jigsaw puzzles that are perfectly reconstructed. We like to emphasize that this performance measure is very strict. A jigsaw puzzle is considered to be "correct" only when all the puzzle pieces are in the right locations.

The comparison is shown in Table V. We can see that our proposed method either outperforms or is comparable to other baseline approaches with respect to all three performance criteria.

Table V
EVALUATION OF REASSEMBLED PUZZLES ON CHO ET AL.'S DATABASE [3] OF 20 IMAGES (EACH CONTAINING 432 PIECES) AND MCGILL IMAGE DATABASE [19] OF 20 IMAGES (EACH CONTAINING 540 PIECES).

On Cho et al.'s database [3]	Direct	Neighbor	Perfect
Cho et al. [3]	0.100	0.550	0
Pomeranz et al. [20]	0.940	0.950	13
Gallagher [8]	0.953	0.951	12
Ours	**0.956**	**0.954**	13
On McGill image database [19]	**Direct**	**Neighbor**	**Perfect**
Pomeranz et al. [20]	0.830	0.910	9
Gallagher [8]	0.928	0.955	11
Ours	**0.935**	**0.967**	11

The benefit of our method becomes even more apparent for puzzles with pieces having imprecise boundaries or noise. Table VI reports the results on images from two categories (forest, city) of the MIT Scene database [18]. There are 328 images in the forest category and 308 images in the city category. Under small Gaussian noise, or with one row/column cropped from the piece boundary, our method outperforms MGC and SSD by about 5% in accuracy with respect to both Direct and Neighbor comparisons. Since large number of noisy or corrupted puzzle pieces makes the task extremely challenging for all the compatibility measures, their performance become difficult to compare. Therefore, we used MIT Scene database [18] that contains images with smaller size than that of Cho et al.'s database [3].

Figure 2 shows some examples of jigsaw puzzles with imprecise piece boundary and the solutions obtained using SSD, Gallagher [8], and our approach. Figure 3 shows examples of jigsaw puzzles with Gaussian noise and the

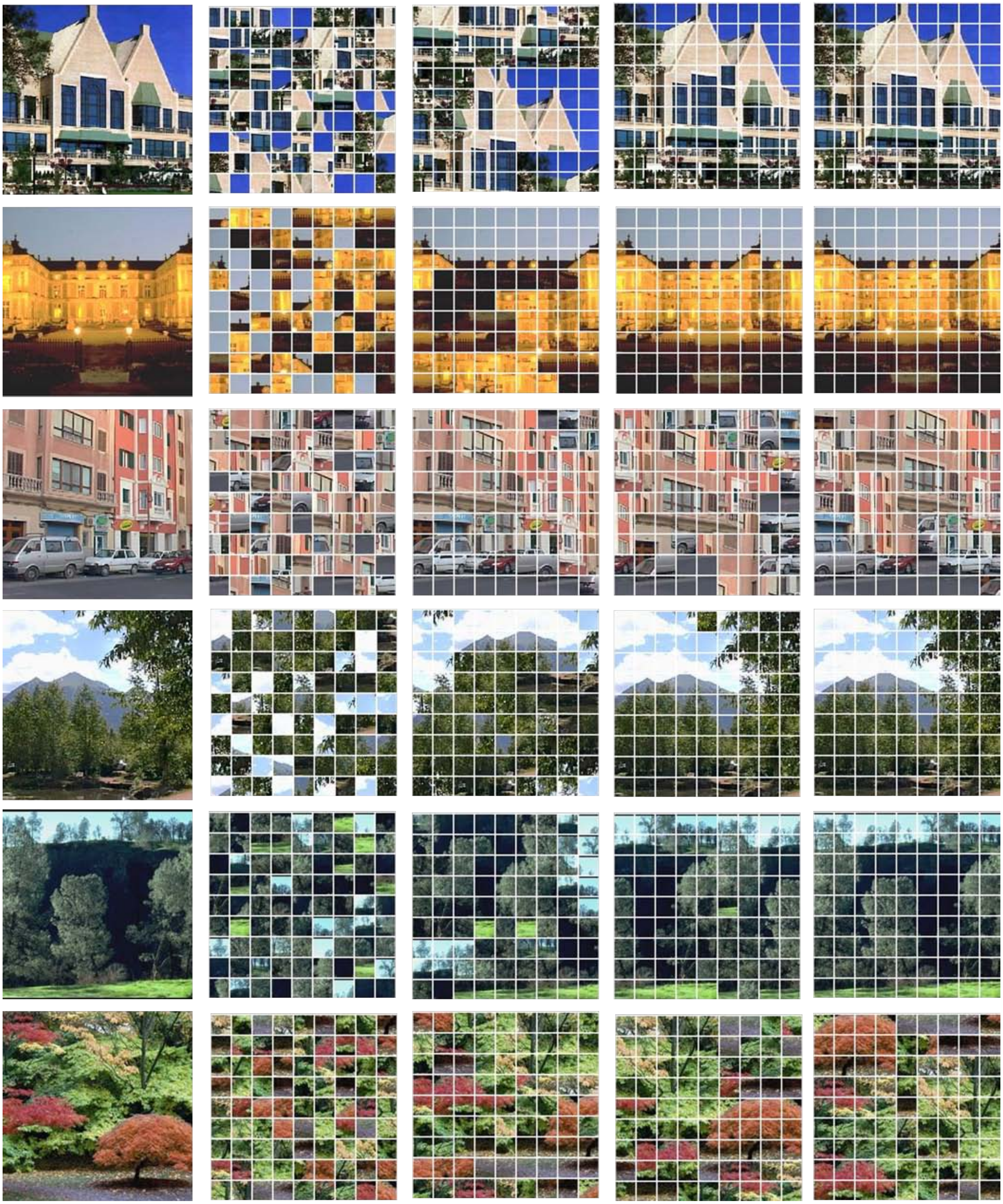

Figure 2. Examples of jigsaw puzzles with imprecise piece boundaries and the solutions obtained using different approaches. Each row corresponds to an example. The columns are: (1st) original image; (2nd) puzzle pieces; (3rd) solutions of SSD; (4th) solutions of Gallagher [8]; (5th) our solutions.

Table VI
EVALUATION OF REASSEMBLED PUZZLES, EACH CONTAINING 81 PIECES OF SIZE $28 \times 28 \times 3$. HERE v IS VARIANCE OF THE MEAN ZERO GAUSSIAN NOISE, AND t IS THE AMOUNT OF CROPPING.

	Dire.	Neig.	Perf.	Scene	v	t
SSD	.694	.741	83	forest	0	0
Gallagher [8]	.969	.977	304	forest	0	0
Ours	**.980**	**.989**	**311**	forest	0	0
SSD	.857	.900	141	city	0	0
Gallagher [8]	.988	.992	285	city	0	0
Ours	**.990**	**.994**	**286**	city	0	0
SSD	.656	.709	65	forest	0.001	0
Gallagher [8]	.732	.784	109	forest	0.001	0
Ours	**.783**	**.830**	**126**	forest	0.001	0
SSD	.758	.813	50	city	0.001	0
Gallagher [8]	.752	.803	63	city	0.001	0
Ours	**.783**	**.831**	**72**	city	0.001	0
SSD	.118	.176	0	forest	0	1
Gallagher [8]	.647	.734	54	forest	0	1
Ours	**.689**	**.772**	**74**	forest	0	1
SSD	.3440	.4597	0	city	0	1
Gallagher [8]	.684	.780	46	city	0	1
Ours	**.746**	**.829**	**76**	city	0	1

solutions obtained using these three methods. In both figures, wMGC perfectly solves the puzzles in the 1st, 2nd, 4th and 6th rows.

VI. CONCLUSION

We have proposed new approaches for measuring the compatibility of two jigsaw puzzle pieces. The proposed compatibility metrics can be used in combination with existing puzzle reassembly method to solve jigsaw puzzles. Our experimental results demonstrate that our proposed compatibility metrics outperform the state-of-the-art approaches, especially when the puzzle pieces have imprecise boundaries or noise.

ACKNOWLEDGMENT

We thank Andrew C. Gallagher for his source code to reproduce the results of his paper.

REFERENCES

[1] N. Alajlan. Solving square jigsaw puzzles using dynamic programming and the Hungarian procedure. *American Journal of Applied Sciences*, 6(11), 2009.

[2] T. Altman. Solving the jigsaw puzzle problem in linear time. *Applied Artificial Intelligence*, 3(4), 1990.

[3] T. S. Cho, S. Avidan, and W. T. Freeman. A probabilistic image jigsaw puzzle solver. In *Proc. of CVPR*, 2010.

[4] M. G. Chung, M. Fleck, and D. Forsyth. Jigsaw puzzle solver using shape and color. In *Proc. of ICSP*, 1998.

[5] M. Ş. Sağiroğlu and A. Erçil. A texture based approach to reconstruction of archaeological finds. In *Proc. of VAST*, 2005.

[6] E. D. Demaine and M. L. Demaine. Jigsaw puzzles, edge matching, and polyomino packing: Connections and complexity. *Graphs and Combinatorics*, 23(1), 2007.

[7] H. Freeman and L. Gardner. Apictorial jigsaw puzzles: A computer solution to a problem in pattern recognition. *IEEE Transactions on Electronic Computing*, EC-13, 1964.

[8] A. C. Gallagher. Jigsaw puzzles with pieces of unknown orientation. In *CVPR*, 2012.

[9] H. Gao, D. Yao, H. Liu, X. Liu, and L. Wang. A novel image based CAPTCHA using jigsaw puzzle. In *Proc. of CSE*, 2010.

[10] D. Ierardi and S. Park. Docking molecules by geometric surface complementarity at multiple resolutions. *Technical report, University of Southern California*, 1995.

[11] R. Jonker and A. Volgenant. A shortest augmenting path algorithm for dense and sparse linear assignment problems. *Computing*, 38(4), 1987.

[12] D. Kosiba, P. Devaux, S. Balasubramanian, T. Gandhi, and K. Kasturi. An automatic jigsaw puzzle solver. In *Proc. of IAPR*, 1994.

[13] H.-Y. Lin and W.-C. Fan-Chiang. Reconstruction of shredded document based on image feature matching. *Expert Systems with Applications*, 39(3), 2012.

[14] M. Makridis and N. Papamarkos. A new technique for solving puzzles. *IEEE Transactions on Systems, Man, and Cybernetics*, 40(3), 2010.

[15] W. Marande and G. Burger. Mitochondrial DNA as a genomic jigsaw puzzle. *Science*, 318(5849):415, 2007.

[16] T. R. Nielsen, P. Drewsen, and K. Hansen. Darpa shredder challenge solved. *Pattern Recognition Letters*, 29(14), 2008.

[17] T. R. Nielsen, P. Drewsen, and K. Hansen. Solving jigsaw puzzles using image features. *Pattern Recognition Letters*, 29(14), 2008.

[18] A. Oliva and A. Torralba. Modeling the shape of the scene: A holistic representation of the spatial envelope. *International Journal of Computer Vision*, 42(3):145–175, 2001.

[19] A. Olmos and F. A. A. Kingdom. McGill calibrated color image database. http://tabby.vision.mcgill.ca/.

[20] D. Pomeranz, M. Shemesh, and O. Ben-Shahar. A fully automated greedy square jigsaw puzzle solver. In *Proc. of CVPR*, 2011.

[21] G. M. Radack and N. I. Badler. Jigsaw puzzle matching using a boundary-centered polar encoding. *Computer Graphics and Image Processing*, 19(1), 1982.

[22] M. Sagiroglu and A. Ercil. A texture based matching approach for automated assembly of puzzles. In *Proc. of ICPR*, 2006.

[23] H. Wolfson, E. Schonberg, A. Kalvin, and Y. Lamdan. Solving jigsaw puzzles by computer. *Annals of Operations Research*, 12(1), 1988.

[24] X. Yang, N. Adluru, and L. J. Latecki. Particle filter with state permutations for solving image jigsaw puzzles. In *Proc. of CVPR*, 2011.

[25] F.-H. Yao and G.-F. Shao. A shape and image merging technique to solve jigsaw puzzles. *Pattern Recognition Letters*, 24(12), 2003.

Figure 3. Examples of jigsaw puzzles with Gaussian noise and the solutions obtained using different approaches. Each row corresponds to an example. The columns are: (1st) original image; (2nd) puzzle pieces; (3rd) solutions of SSD; (4th) solutions of Gallagher [8]; (5th) our solutions.