Solving Square Jigsaw Puzzle by Hierarchical Loop Constraints

Kilho Son, *Member, IEEE*, James Hays, *Member, IEEE*, and David B. Cooper, *Fellow, IEEE*

Abstract—We present a novel computational puzzle solver for square-piece image jigsaw puzzles with no prior information such as piece orientation or anchor pieces. By "piece" we mean a square d x d block of pixels, where we investigate pieces as small as 7×7 pixels. To reconstruct such challenging puzzles, we propose to find maximum geometric consensus between pieces, specifically hierarchical piece loops. The proposed algorithm seeks out loops of four pieces and aggregates the smaller loops into higher order "loops of loops" in a bottom-up fashion. In contrast to previous puzzle solvers which aim to maximize compatibility measures between all pairs of pieces and thus depend heavily on the pairwise compatibility measures used, our approach reduces the dependency on the pairwise compatibility measures which become increasingly uninformative for small scales and instead exploits geometric agreement among pieces. Our contribution also includes an improved pairwise compatibility measure which exploits directional derivative information along adjoining boundaries of the pieces. We verify the proposed algorithm as well as its individual components with mathematical analysis and reconstruction experiments.

Index Terms—Computational jigsaw puzzle, discrete optimization algorithm, hierarchical loop constraints, maximizing consensus

1 Introduction

A puzzle by definition [1] is a game, toy or problem designed to test ingenuity or knowledge. Various types of puzzles such as jigsaw puzzles, crossword puzzles, Rubik's cube puzzles and Sudoku (number puzzles) are enjoyed by people. The puzzles initially "puzzle" players with the huge number of possible answers. The partial answers of the puzzle, however, provide clues (or constraints) to extend solutions to the neighboring parts of the puzzle. The solution to the entire puzzle can often be uncovered from the agreement of all the partial answers. Due to this interesting property, the puzzles are also viewed as a serious scientific problem and their solutions significantly contribute to many applications such as encryption and decryption [2].

Introduced many centuries ago, a jigsaw puzzle requires the assembly of oddly shaped pieces where each piece displays a part of a picture or a pattern. When completely assembled, the jigsaw puzzle presents a natural and whole image or pattern. Although a jigsaw puzzle is generally enjoyed as an educational pastime, in other fields such as archeology and medicine many researchers work on assembling ancient artifacts, priceless but shredded documents, and even fractured and displaced bone fragments. Recent advances in computer science and engineering enable researchers to develop computational algorithms and systems to solve digitized jigsaw puzzles. Computational puzzle solvers have been important tools to organize clues to assemble time-consuming and sometimes infeasible jigsaw puzzle problems [3], [4], [5], [6], [7], [8].

The computational puzzle problems have also been related with fundamental problems in various fields such as machine learning, computer vision and bio-informatics, and algorithms of computational puzzles have been applied to solve problems in these fields. The goal of computational puzzle solving could be understood as constructing seamless images or 3D structures given partial and unorganized observations. Many problems in computer vision and graphics such as panorama assembly, image completion, and even object recognition share the same core problem and existing puzzle solvers have been successfully applied to those problems [9], [10]. In addition, the puzzle assembly problem, with some modifications, could be understood as inferring seamless latent variables given local observations which is a fundamental problem in computer vision such as stereo matching and optical flow. Thus, the main ideas and technologies in computational puzzle solvers could be applied to improving tools for capturing relational information such as Graphical Models. Furthermore, computational Jigsaw puzzle problem could be also viewed as modeling relational information among objects [11], [12]. If we consider the puzzle pieces as objects, a goal of puzzle problem is to construct a structure of a network between the objects. This problem corresponds to constructing a sensor network or a social network which is a fundamental problem in computer vision such as SLAM, SFM, tracking and even template matching as well as Bio-informatics such as modeling interaction network between proteins [13]. Some techniques in computational puzzle solvers are analogous to strategies used in these domains [14], [15], [16].

- *K. Son and D. B. Cooper were with the School of Engineering, Brown University, Providence, RI 02912. E-mail: kilho.son@gmail.com, david_cooper@brown.edu.*
- *J. Hays is with the School of Interactive Computing, Georgia Institute of Technology, Atlanta, GA 30332. E-mail: hays@gatech.edu.*

Manuscript received 29 June 2017; revised 6 June 2018; accepted 3 July 2018. Date of publication 18 July 2018; date of current version 13 Aug. 2019. (Corresponding author: Kilho Son.) Recommended for acceptance by S. Roth. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below. Digital Object Identifier no. 10.1109/TPAMI.2018.2857776

Introduced by Freeman et al. [17], the task of solving *square* jigsaw puzzles has been a challenge to many researchers. The pieces of the puzzles are all square-shape and their sizes are the same, thus a clue for assembling these pieces is only texture information on the pieces without geometric information which would be available if the shape of each of piece were different. Some prior information such as piece orientation, anchor pieces (ground truth configurations), and dimension of resulting puzzles (number of pieces in the horizontal and vertical directions of resulting assemblies) is often given to alleviate the difficulties of the problem. Recently, many works [3], [18], [19], [20], [21], [22], [23], [24], [25], [26] tackled non-overlapping square piece image jigsaw puzzles and notable progress has been made.

In this paper, we propose a solver for challenging square piece image jigsaw puzzles where no prior information—piece orientation, anchor pieces, and dimension of resulting puzzles are all unknown. Furthermore, we also attempt to assemble puzzles with small pieces, 14 by 14, 10 by 10 and 7 by 7 pixels. When the size of a piece becomes smaller, the information in the pieces is reduced so that the predefined compatibility measure is no longer reliable enough to configure the puzzles correctly. Thus, we must develop smarter and more robust strategies to assemble small pieces successfully.

For assembling these demanding puzzles, we choose a framework that estimates pairwise matching and subsequently aligns these matched pieces by making use of all the pairwise costs involved in the aligned sets. Because we are using this framework, our puzzle solver is capable of simultaneously reconstructing multiple puzzles, whose pieces are mixed together, with no prior information. Our solver also handles missing fragments, which is a common situation in archaeological applications [27]. The most challenging aspect of this puzzle reconstruction framework is that the consecutive pairwise matches should not include false pairwise matches. Even a single initial false pairwise match may lead to an entire incorrect configuration of many pieces. And, missing a true pairwise match also could cause serious problem. That is, if we miss true pairwise matches, some pieces can not be added to the existing configurations.

To overcome these issues, we contribute two crucial steps in the matching-based assembly algorithm: an assembly strategy and a pairwise compatibility measure. For assembling pieces, we consider that the pairwise compatibility measure could be noisy when the textures on the pieces are noisy, the pieces are small, the pieces are from textureless regions, or the boundaries of pieces are aligned with boundaries of image structures (e.g., the edge of a building). Thus, we exploit an idea that an assembly by multiple modest-bonds is more reliable than an assembly by a strong single bond which is found by the pairwise compatibility measure. That is, configurations built from *consensus* or agreement from the neighbor pieces are reliable and robust to noise. From this idea, we propose a new objective for assembling puzzles—maximizing consensus configurations—specifically hierarchical loops. In order to search for configurations which maximize consensus, the proposed algorithm explicitly finds all small loops or cycles of four pieces and in turn groups these small loops into higher order "loops of loops" in a bottom-up fashion. Our method uses these puzzle loops, specifically 4-cycles, as a form of outlier rejection whereas previous methods, e.g., Gallagher [21], treat cycles as a nuisance and avoid them by constructing cycle-free trees of puzzle pieces. Then, the algorithm grows the size of the loops of loops, the hierarchical loops, by adding a new piece to the existing configurations if the addition increases the size of hierarchical loop configurations. This procedure operates regardless of pairwise compatibility measure used. This growing process plays an important role to find configurations which are missed by misleading local pairwise compatibility measures. During this assembly, some of the small loops discovered may be spurious. Our algorithm then proceeds top-down to merge unused loop assemblies onto the dominant structures if there is no geometric conflict. Otherwise, the loop assemblies are broken into sub-loops and the merging is attempted again with smaller loops. If the loops of 4 pieces still geometrically conflict with the dominant structures, we remove them as an another form of outlier rejection. In addition, we also improve the pairwise compatibility measure by adding derivative information along the boundary in the piece to the pairwise compatibility measure.

1.1 Related Works

Although Demaine et al. [28] discovered that puzzle assembly is an NP-hard problem, many published algorithms incrementally improved the performance with novel ideas. For non-overlapping square piece jigsaw puzzles, many researchers have studied solvers to enhance reconstruction accuracy and reduce prior information needed. Earlier works [3], [18], [19], [20], [22], [23] solved image puzzles where the orientation of the puzzle pieces is known, and additional information is given such as the dimensions of resulting puzzles. The positions of the puzzle pieces are the only unknowns. If the dimensions of the resulting puzzle are known, the puzzle problem can be formulated as recovering the optimal 2D permutation of labels where the labels are the IDs of the pieces. Implicitly or explicitly, their objectives are to maximize a predefined pairwise compatibility measure between neighbor pieces and various optimization methods were applied such as a belief propagation [3], particle filtering [19], greedy algorithms [20], constrained quadratic function minimization [22] and genetic algorithms [23]. Over the years these methods substantially enhanced reconstruction accuracy, increased the number of puzzle pieces that the solver can perfectly assemble, and reduced computational time. However, when their compatibility measures are not reliable enough their objective functions often lead to false configurations.

Gallagher [21] introduced a new type of image puzzle where the orientation of the piece is also unknown with no prior information such as a resulting puzzle dimension. Since the dimension of the puzzle is unknown, Gallagher formulated the puzzle problem as optimally linking the pieces without geometric conflicts such as piece overlaps and solved it with Kruskal's algorithm [29]. Gallagher also proposed a powerful pairwise compatibility measure, Mahalanobis Gradient Compatibility (MGC) which penalizes gradient changes across the pieces when the changes are more than expected and normalizes them with sample covariance estimated by pixels around the edge of the piece. Even though we argue for a puzzle assembly strategy which

is effectively opposite to Gallagher, because we try to leverage puzzle cycles early and Gallagher tries to avoid them, the proposed MGC pairwise compatibility measure is a significant improvement on the previous works and their method performs well. Paikin et al. [25] also proposed a similar greedy assembly strategy with a more careful initial configuration. These assembly strategies [21], [25] are also based on a strong belief on the possibly incorrect pairwise compatibility measure. They accept a puzzle configuration as long as there is a single strong bond even though the other sides of these pieces could be incompatible with their neighbors. Thus, when false pair matches return high compatibility values or true pair matching returns low compatibility value due to insufficient information or unexpected noise, their assembly strategies fail to build true assemblies. Yu et al. [30] formulated solving jigsaw puzzle as minimizing a non-convex function with integer constraints. The objective function was relaxed to form a linear objective function with linear constraints and solved by linear programming. They iteratively solve the relaxed objective with new constants acquired by the previous solution. The reformulated objective, however, is limited because it does not precisely correspond to the original objective function due to the relaxation.

Distinct from the previous algorithms, our assembly strategy depends less on the pairwise compatibility measure used. When a pairwise compatibility measure returns high values on false pairs, we do not accept these pairs unless these pairs agree with the other neighbor pieces. Conversely, when true pairs output low compatibility measure values, we accept these configurations if these pairs receive agreements from the other neighbors pieces. By exploiting consensus information from the neighbors, our proposed algorithm is robust to noisy compatibility measure values which occur in many challenging puzzles with small size pieces and/or non-informative pieces.

"Consensus" information has been utilized in many fields. Robust estimation, exploiting consensus of the measurements such as RANSAC has presented impressive improvement on many computer vision applications [31], [32]. Robust estimation has proven useful for camera geometry estimation [33]. Consensus information was used in low-level vision. Chakrabarti et al. [34] introduced a multi-scale framework to accumulate information from all the scales and found a consensus to reject the outlier information. The "Best Buddies" strategy [20] that prefers pair matches when each piece in the pair independently believes that the other is its most likely match can be considered as a type of loop consensus. Our algorithm using hierarchical loop constraints extends the basic idea of "Best Buddies" [20]. In stead of using 2 pieces that agree on 1 boundary, we further exploit a higher-order fundamental unit of 4 pieces that agree on 4 boundaries. This higher ratio of boundaries to pieces gives us more information to constrain our matching decisions. Our algorithm also proceeds from coarse to fine, so that higher-order loops can reject spurious smaller loops.

1.2 Outline

We first present the proposed assembly algorithm, given a pairwise compatibility measure, in Section 2, and an enhanced pairwise compatibility measure which exploits directional derivative information along the boundaries of the adjoining pieces in Section 3. We attempt to investigate the validity of the proposed assembly algorithm analytically and empirically in Section 4. Extensive experiments including component analysis and performance comparison with state of the art algorithms with both the known and unknown orientation standard puzzles are presented in Section 5.

2 Assembly Algorithm

In this section we describe the proposed assembly algorithm built on hierarchical loop constraints given a predefined compatibility measure between a pair of pieces. Our proposed compatibility measures are presented in Section 3.

2.1 Overview

We present a computational puzzle solver for image jigsaw puzzles where the each piece is a square and pieces do not overlap. Our puzzle problem excludes prior knowledge such as orientation of the pieces, position of "anchor" pieces, and the resulting dimension of the puzzles. The puzzle pieces may be from a single image or multiple images and some pieces may be missing. That is, our puzzle solver computationally outputs assemblies solely from unorganized piles of squared-pieces.

Similar with the previous algorithms [21], we assemble the puzzles by estimating pairwise matches and consecutively adding pieces based on pairwise matching. Although this approach does not require any prior information, which is suitable for the demanding puzzles defined in this article, assembly results are free form so that the accurate assemblies are more challenging.

Prior works [3], [19], [20], [21], [22], [23] perform accurately on puzzles where the pairwise compatibility measure correctly identifies which pieces are true neighbors. If the pairwise compatibility measure is inaccurate the algorithms often fail. These challenging cases happen frequently when the size of the pieces is small, the textures on the pieces have no distinguishing characteristics, or the boundaries of the pieces are aligned with edges of man-made structures. Here, we propose a new objective for puzzle assembly—maximizing consensus configurations, specifically hierarchical loops. In contrast to previous algorithms which aim for assemblies maximizing compatibility measures between all pairs of pieces and thus depend heavily on the pairwise compatibility measure used, we attempt to find configurations which maximize a geometric consensus between pieces. That is, we believe harmonious (non-conflicting and supporting each other) piece configurations are more important than a single bond between two pieces defined by the pairwise compatibility measure (validations of our new objective are presented in Section 4). With the proposed objective, we reduce the dependency on the possibly erroneous pairwise compatibility measure and instead exploit the geometric information of existing piece configurations. In the case where the compatibility measure fails to distinguish the true neighbor pieces, we successfully assemble the puzzle with help of geometric consensus from neighbor pieces.

Our proposed algorithm begins by searching for pairwise matching candidates using a predefined pairwise compatibility measure (Section 2.2). Then, outliers in the pairwise matching candidates are rejected by constructing

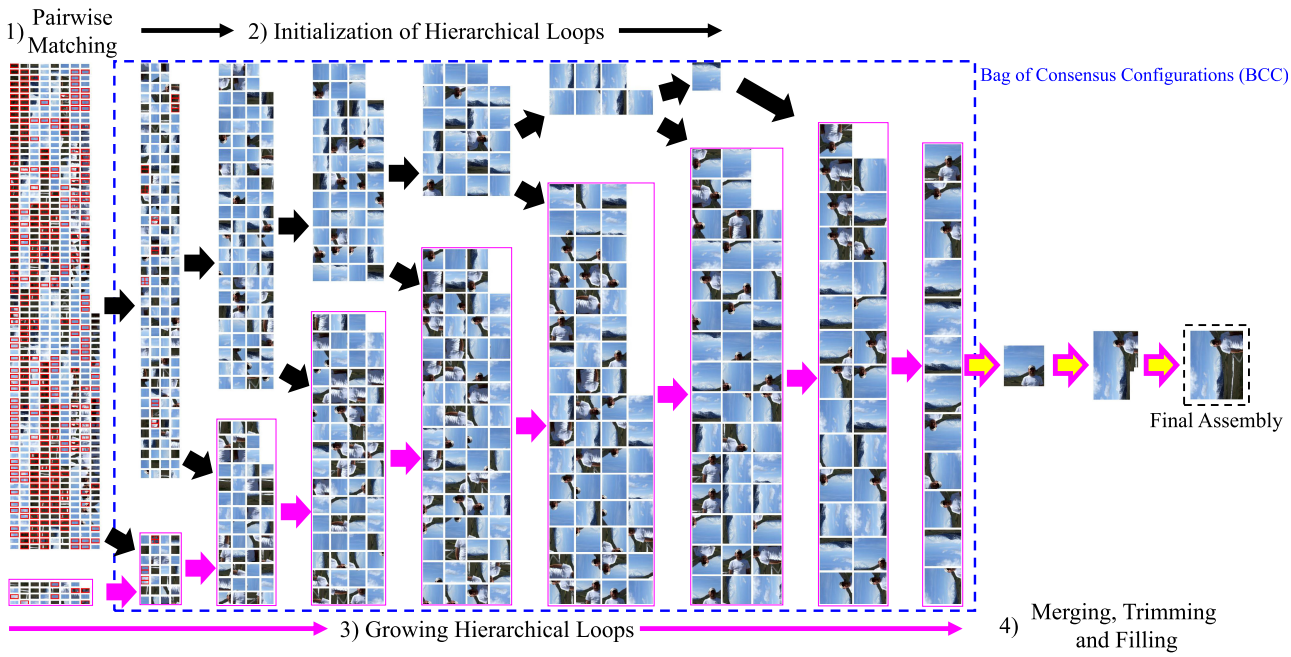


Fig. 1. An example of puzzle assembly process with hierarchical loop constraints. 1) The pairwise matches are selected as candidates when the pairwise compatibility measures are greater than a threshold. 2) We discover small loops of increasing order, hierarchical loops (consensus configurations), from the collection of the pairwise matching candidates. The first small loops discovered, made from four matched pairs, are of order 2. We then iteratively build higher order loops from the lower order loops, e.g., 4 loops of order 2 are assembled into loops of order 3, i.e., loops of 3 by 3 pieces, and so on. We continue assembling loops of loops in a bottom-up fashion until the algorithm finds maximum loop(s). 3) Independent of the pairwise compatibility measure, we add more pairwise matches iteratively if they are harmonious with existing hierarchical loops and, as a result, increase the order of existing hierarchical loops. 4) The algorithm then proceeds top-down to merge unused loop assemblies onto the dominant structures. This top-down merging is more permissive than the bottom-up assembly and has no "loop" constraint, as long as two hierarchical loops share at least two pieces in consistent position they are merged. At a particular level, remaining loops are considered for merging in order of priority, where priority is determined by the mean of pixel variances on boundaries of the pieces in that loops. After all possible merges are done for loops of a particular order, the unused loops are broken into their sub-loops and the merging is attempted again with smaller loops. If the reconstruction is not rectangular, trimming and filling steps are required. The puzzle in this example is very challenging because all the pieces are only 14 by 14 pixels. For illustration purpose, false configurations are highlighted in red. Note that when the order of hierarchical loops is equal to or greater than 3, there are no false configurations in this example.

hierarchical loops in a "bottom-up" fashion (Section 2.3). When pairwise matches return low compatibility so that they are not found in the pairwise matching step but they are harmonious with current configurations and as a result increase the size of the hierarchical loops, the pairwise matches are added to the current configurations independent of the compatibility measure (Section 2.4). As a final step, we process "top-down" to merge unused loop assemblies to the dominant structure and refine the assemblies by trimming and filling the pieces so as to make the configurations rectangular (Section 2.5). For more overview of our algorithm, refer to Fig. 1.

2.2 Local Pairwise Matching

Before assembling loops we must consider the pairwise similarity metric which defines piece compatibility and a strategy for finding candidate pair matches. In unknown orientation puzzles, two square puzzle pieces can be aligned in 16 different ways (in known orientation puzzles, two pieces can be assembled in 4 different ways). We calculate dissimilarities between all pairs of pieces for all 16 configurations using the Sum of Squared Distances (SSD) in LAB color space from Cho et al. [3], Mahalanobis Gradient Compatibility (MGC) from Gallagher [21], or our proposed compatibility measure explained in Section 3. Absolute dissimilarities between potential piece matches are not useful (e.g., sky pieces will always produce smaller dissimilarities), so we follow the lead of Gallagher [21] and use *dissimilarity ratios* instead. For each edge of each piece, we divide all dissimilarities by the smallest matching dissimilarity. Unless otherwise stated, edge matches with a dissimilarity ratio less than $\theta = 1.07$ are considered "candidate" matches for further consideration. Using dissimilarity ratios makes edge dissimilarities more comparable but it has the downside that the smallest dissimilarity ratio is exactly 1 for every edge and thus they are no longer comparable. To alleviate this, the smallest dissimilarity ratio is substituted with the reciprocal of the second smallest dissimilarity ratio. We limit the maximum number of candidate matches that one side of a piece can have to $\zeta = 10$ for computational efficiency.

2.3 Initialization of Hierarchical Loops

In this step we initialize hierarchical loops, referred to "a Bag of Consensus Configurations (BCC)", from the pairwise matching candidates (See Figs. 1 and 2). In the first iteration all cycles (loops) of four pairwise matches in the candidates found in Section 2.2 are discovered resulting in a set of assemblies of 2 by 2 pieces (order 2). For example, if there are four pairwise matches right side of x_1 - left side of x_2, bottom of x_2 - top of x_3, left side of x_3 - right side of x_4 and top of x_4 - bottom of x_1, the four pieces (x_1, x_2, x_3, x_4), with four pairwise matching, make a loop and generate a 2 by 2 assembly whose elements are clockwise x_1, x_2, x_3 and x_4.

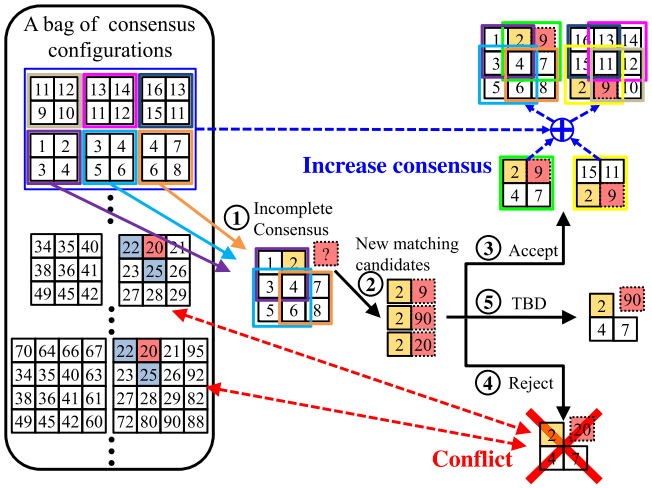


Fig. 2. An example of single iteration of increasing consensus configurations. The squares and the numbers within represent pieces and their unique IDs respectively. From the Bag of Consensus Configurations (BCC), ① we search for order 3 incomplete consensus from three loops of order 2. Note that the assemblies in the BCC are always square, and order N consensus configuration means that the dimension of the assembly is N by N pieces. ② we find pairwise matching candidates to complete this order 3 incomplete consensus. Since pairwise dissimilarity measures between top of piece 7 and bottom of pieces 9, 90 and 20 are lower than the threshold (θ), the right side of piece 2 to the left side of those pieces (each of ID $9, 90, 20$) are considered as new pairwise matching candidates, independent of the pairwise compatibility measure. Likewise, pieces returning lower dissimilarity measures with the right side of piece ID 2 could propose more pairwise matching candidates with piece ID 7. ③ If the new pairwise match completes at least two consensus configurations of order 3, we accept the pairwise matches. If the newly accepted pairwise match is true, this addition sometimes generates 2 or more additional larger sized consensus configurations. In this example, two consensus configurations of order 3 are generated additionally. ④ If the candidate match conflicts with existing larger consensus configurations (in this example, order 3 or larger configurations), we reject the pairwise match. Otherwise, ⑤ we hold off the decision to the next iteration. We note that the decision is made only by existing consensus configurations in the BCC, independent of the pairwise compatibility strength.

Once all consistent loops are discovered, these loops are assembled into higher-order, loops of 4 cycles. For example loops of order 2 are assembled to construct loops of order 3. In detail, we consider the assemblies of order 2 as new super pieces of order 2 and discover pairwise matching between the super pieces. For matching between super pieces, instead of comparing boundary pixels, we use configurations of two super pieces. For left and right matching between two super pieces of order 2, we overlap the last column and the first column of the two super pieces respectively. If 1) the overlapped region presents the same IDs and rotations in both super pieces and 2) the combined configuration does not present the same ID repeatedly, we consider them matched.[1] Once we find pairwise matches of super pieces of order 2, we perform the same process as before to generate super pieces of order 3 from four super pieces of order 2. More generally, with this process we generate super pieces of order $N+1$ from four super pieces of order N. We iteratively increase the size of super pieces, one row and one column at a time, until we find the maximum size super pieces. We refer to the entire set of generated super pieces from order 2 to maximum as a Bag of Consensus Configurations (BCC).

Different consensus configurations may include the same ID and some of consensus configurations are possibly false configurations. The consensus configurations are always square. We note that as the consensus configurations become larger, the correctness of the assemblies grows which is discussed in Section 4.

2.4 Growing Hierarchical Loops

In this step our algorithm attempts to increase a size of hierarchical loops, refer to as consensus configurations in the BCC, initialized in the Section 2.3. The algorithm iterates two processes: 1) Proposing New Pair Matching Candidates Step and 2) Rejecting or Accepting Step until there are no incomplete consensuses in the BCC. When 3 consensuses of order N are matched well but only 1 consensus of order N is missing so that they cannot form order $N+1$ consensus, we call them an order $N+1$ incomplete consensus (See Fig. 2).

Iteratively, we consider new pairwise matching candidates which possibly complete existing incomplete consensuses in the BCC and increase the order and the number of the existing consensus configurations in the BCC. If new pairwise matching candidates agree with existing configurations and, as a result, grow the consensus configurations in the BCC, we accept to add them to the BCC. We reject to add the new pairwise matching candidates to the BCC if they are inconsistent with dominant configurations in the BCC. We note that those decisions are made independent of the predefined compatibility measure. More details are explained below.

1) Proposing New Pair Matching Candidates Step: We first discover new pairwise matching candidates which are not found in the first step (Section 2.2) due to their low pairwise compatibility measures but possibly increase the size of the consensus configurations in the BCC. For this purpose, the algorithm examines from the largest to the smallest consensus configurations in the BCC, and searches for incomplete consensuses made from 3 consensuses of the same order in the BCC. If a new pairwise match possibly completes the incomplete consensus as exemplified in Fig. 2, we count it as a new pairwise matching candidate, independent of the pairwise compatibility measure.

In detail, we search for the largest and unvisited incomplete consensuses in the BCC. An order $N+1$ incomplete consensus made from three consensuses of order N can be completed by adding 1 piece to the corner of the incomplete consensus and forms order $N+1$ consensus (see Fig. 2). In order to add 1 piece to the corner of the incomplete consensus and complete the incomplete consensus, two adjacent sides of the piece should be compatible to the corner of the incomplete consensus. Thus, the algorithm searches for a piece whose dissimilarity measure of one side to one corner of the incomplete consensus is lower than the threshold (θ) but dissimilarity measure of the adjacent side to the other corner of the incomplete consensus is higher than the threshold which is tentative missing true pairwise matching. The algorithm, thus, considers the pairwise matching between the new piece to the corner of the incomplete consensus whose dissimilarity measure is higher than the threshold (θ) as a new pairwise matching candidate. We do

1. For generalization of matching between super pieces, when two super pieces of order N are given and we consider left and right match between the super pieces, we overlap column 2 to N of the left piece with 1 to $N-1$ of the right piece.

not accept the new pairwise matching candidate now since the new pairwise matching received insufficient agreement from its neighbor pieces.

2) Rejecting or Accepting Step: For each new pairwise matching candidate, we take action of rejection, acceptance, or To Be Determined (TBD). A new pairwise matching candidate is rejected if the new pairwise matching configuration is inconsistent with existing consensus configurations whose order is greater than the order of consensus configurations that are used for generating the new pairwise matching candidate. That is, when a new pairwise matching candidate is considered to form order N consensus configurations, the new pairwise matching candidate is rejected if the configuration of the new pairwise matching conflicts with the existing consensus configurations whose order are equal to or greater than N. We accept a new pairwise matching candidate if this new pairwise matching generates 2 or more new consensus configurations of order $N$and those new configurations do not conflict with existing consensus configurations whose size are equal to or greater than order N. Otherwise, we consider a pairwise matching candidate as TBD which will be determined in the next iteration. That is, if a new pairwise matching candidate neither conflicts with current configurations nor contributes to increase the size of consensuses, we leave the decision to the next iteration. We note that these decisions are made only by the existing consensus configurations in the BCC, independent of the pairwise compatibility measure.

2.5 Merging, Trimming and Filling

At this point the algorithm proceeds in a top-down fashion and merges successively smaller remaining consensus configurations to the dominant consensus configurations without enforcing loop constraints. Consider a pair of consensus configurations of the largest order in BCC, and whether we merge them or not when the two consensuses share at least two of the same ID pieces.[2] The two consensuses are aligned by one of the shared pieces, and they are merged if there is no geometric conflict. We define geometric conflict as 1) a overlap with different ID or rotation, or 2) an existence of same IDs two or more times in a non-shared region. If there is conflict, the lower priority consensus configuration is broken into sub-loops and the merging is attempted again with smaller loops. The priority of consensus configurations are determined by the size of the configuration. If the sizes are the same, the consensus with the larger mean of pixel variance along the boundaries of the member pieces (more texture) has higher priority. After all possible merges are done for all consensus configurations of a particular order, the dominant structures merged as well as unused consensuses broken into sub-consensuses are attempted to be merged with consensus configurations of the next lower order.

The resulting assemblies from these steps are not guaranteed to be rectangular, because the proposed algorithm is a matching-based algorithm whose results are free-form. In this case we estimate the most probable dimension of the resulting puzzle given the current main configuration and the total number of pieces. With the estimated dimension of the puzzle, the algorithm trims the main configuration so that it cuts minimum order of loops in the main configuration. This is because higher order hierarchical loops are more reliable than are smaller ones (Section 4). We fill the holes with the remaining pieces in the order of minimum total dissimilarity score across all neighbors.

2. A single piece correspondence is enough to establish the geometric relationship between two groups of pieces, but we choose not merge on the basis of singe piece correspondences. Such correspondences are more likely to be spurious. However, if they are true correspondences it is likely that as the pieces grow they will eventually have two or more shared pieces and thus become merge-able by our criteria.

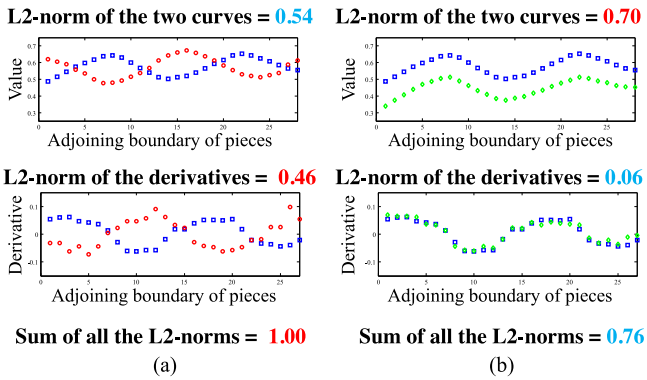


Fig. 3. Importance of derivative information for defining dissimilarity between pieces. Defining dissimilarity between two pieces could be seen as establishing a distance metric between two discretized curves where the two curves represent the values along the boundaries of the two pieces. Prior works define $L2$ norm as dissimilarity of two curves. In (a), two curves are more dissimilar than two curves in (b). But, the $L2$ norm of the two curves in (b) is greater than that in (a) which is not desired. The $L2$ norm of the two curves does not exactly represent the dissimilarity between two curves. By considering the derivatives in a direction along the boundary, we can compensate the erroneous measure. The value of the $L2$ norm of the derivative curve error in (a) is greater than that in (b) which is correct. For this reason, we incorporate derivative information along the adjoining boundaries for defining pairwise compatibility measure.

3 Pairwise Compatibility Measure

For measuring compatibility between two pieces, we calculate the dissimilarity between adjoining boundary pixels of the two pieces. Our pairwise compatibility measure has two components. First, consider pixel intensity changes crossing the border between two pieces. If the changes are greater or smaller than expected, we penalize it. This is similar to Mahalanobis Gradient Compatibility (MGC) by Gallagher [21] but we estimate the expected changes more accurately. Second, we incorporate directional derivative information along the adjoining boundaries of the pieces. If the directional derivative changes across the pieces are greater or smaller than expected, we also penalize it. This proposed pairwise compatibility measure is robust to the illumination changes on the boundary pixels that result in different offset levels because the derivative information is invariant to offset of pixel values as explained in Fig. 3.

Our proposed pairwise compatibility measure between the right side of the piece x_i and the left side of the piece x_j consists of four terms,

$$\begin{aligned}\Gamma_{LR}(x_i, x_j) = D_{LR}(x_i, x_j) + D_{RL}(x_i, x_j) \\ + D'_{LR}(x_i, x_j) + D'_{RL}(x_i, x_j).\end{aligned} \tag{1}$$

The first two terms penalize larger or smaller changes of the pixel value across the two pieces than expected. When the

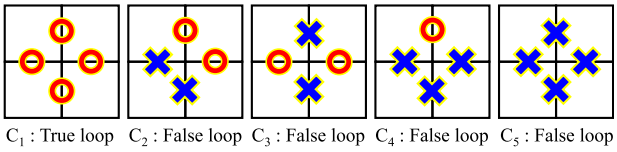


Fig. 4. All cases of loop of four pieces. The circle and the cross represent a true positive and a false positive pairwise matching respectively.

size of piece is S pixels,

$$D_{LR}(x_i, x_j) = \sum_{s=1}^{S} (\Lambda_{LR}^{ij}(s) - E_{LR}^{ij}(s)) V_{iL}^{-1} (\Lambda_{LR}^{ij}(s) - E_{LR}^{ij}(s))^T, \quad (2)$$

where,

$$\Lambda_{LR}^{ij}(s) = x_j(s,1) - x_i(s,S), \quad (3)$$

$$E_{LR}^{ij}(s) = \frac{1}{2}(x_i(s,S) - x_i(s,S-1) + x_j(s,2) - x_j(s,1)). \quad (4)$$

$x_i(u,v)$ is a 3 dimensional vector representing red, green and blue pixel intensities at the position (u,v) in the puzzle piece i. $\Lambda_{LR}^{ij}(s)$ is a pixel intensity change across the boundary of the two pieces and $E_{LR}^{ij}(s)$ is an expected change across the boundary of the two pieces. We normalize the changes with the sample covariance V_{iL} calculated from samples, $\{x_i(s,S) - x_i(s,S-1))|s = 1,2,\dots S\}$. $D_{RL}(x_i, x_j)$ is calculated in the same way.

$D'_{LR}(x_i, x_j)$ is calculated in a similar manner. The only difference is that we replace the pixel values $x_i(u,v)$ in the Equations (2), (3) and (4) with the directional derivatives of pixel values along the boundary of the piece,

$$\delta_i(u,v) = x_i(u,v) - x_i(u-1,v). \quad (5)$$

The sample variance V_{iL} for $D'_{LR}(x_i, x_j)$ is calculated from samples, $\{\delta_i(s,S) - \delta_i(s,S-1))|s = 2,\dots S\}$.

MGC assumes that the expected change across the piece boundaries is an average difference between the last two columns in x_i whereas we estimate the expected changes as an average of local pixel changes in both pieces. Our expected changes are more accurate than those used in MGC especially when the size of the piece is large.

4 Analysis of Algorithm

The main contribution of our work is a novel, hierarchical loop-based strategy to reconstruct puzzles from the local matching candidates. Using loops of four pieces as a fundamental unit for reconstruction is advantageous over alternative methods in which candidates are chained together without considering cycles (or explicitly avoiding cycles) because the loop constraint is a form of outlier rejection. A loop of potential matches indicates a consensus among the pairwise distances. While it is easy to find 4 pieces that chain together across 3 of their edges even if one or more of the three matches are spurious, it is unlikely that the fourth edge completing the cycle would also be among the candidate matches by coincidence. In fact, to build a small loop which is not made of true positive matches, *at least* two of the edge matches must be wrong (Fig. 4). While some loops of 4 pieces will contain incorrect matches that none-the-less lead to a consistent loop, the likelihood of this decreases as hierarchical loops are assembled. A loop of order 3, built from 4 loops of order 2, reflects the consensus among many pairwise dissimilarity measurements. This is also analyzed further in Sections 4.1, 4.3 and 4.2.

Our method focuses on the shortest possible cycles of pieces at each stage – loops of length 4. Longer loops could be considered, but in this study we use only loops of length 4, because (1) longer loops are less likely to be made of entirely true positive pairwise matches and (2) the space of possible cycles increases exponentially with the length of the cycle. While it is possible for us to enumerate all 4-cycles built from candidate pairwise matches in a puzzle, this becomes intractable with longer cycles. (3) Our algorithm is coarse-to-fine, so larger scale cycles are already discovered by finding loops of higher order.

4.1 Analysis of a Loop of Four

In this section, we analyze the fundamental unit of our proposed algorithm, loops of four pieces in the unknown orientation square jigsaw puzzle. Specifically, we show that with pairwise matching algorithms of high enough precision (ratio of true positives to positives) and recall (proportion of true positives retrieved), a loop of four can be more correct than a single pairwise matching is.

For the unknown orientation jigsaw puzzle, let the performance of a pairwise matching method be summarized with precision β at recall α. For this analysis, we assume each matching event is independent with an identical probability distribution. When the number of pieces in row and column of the resulting puzzle is $m_1 \times m_2$, we can calculate an upper and a lower bound of expected number of true or false loops of four based on the pairwise matching performance as,

$$C_1 \geq \max((4\alpha - 3)m_1 m_2 - m_1 - m_2 + 1, 0) = C_1' \quad (6)$$

$$C_2 < \frac{2}{3}\frac{\alpha^3(1-\beta)^2}{\beta^2} = C_2' \quad (7)$$

$$C_3 < \frac{8}{9}\frac{\alpha^4(1-\beta)^2}{\beta^2} = C_3' \quad (8)$$

$$C_4 < \frac{4}{3}\frac{\alpha^4(1-\beta)^3}{\beta^3} = C_4' \quad (9)$$

$$C_5 < \frac{4}{3}\frac{\alpha^4(1-\beta)^4}{\beta^4} = C_5', \quad (10)$$

where C_1 is an expected number of true positive loops and the others (C_2, C_3, C_4, C_5) are expected numbers of false positive loops in different cases (see Fig. 4). Derivations of these inequalities are discussed in Appendix 7. From these inequalities, we can calculate a lower bound for the ratio of the number of all the true positive loops to the number of all the positive loops we discover, i.e., precision of loop of four below,[3]

3. A function $f(x,y) = \frac{x}{x+y}$ where x and y are non-negative is monotonically increasing along the x direction and decreasing along the y direction. Thus, $\frac{C_1'}{C_1'+C_2'+C_3'+C_4'+C_5'} \leq \frac{C_1}{C_1+C_2+C_3+C_4+C_5}$.

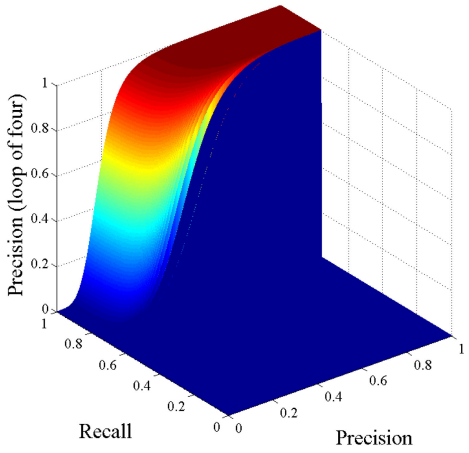


Fig. 5. Precision lower bound of loop of four pairwise matching according to precision and recall of pairwise matching, $m_1 = 18, m_2 = 24$.

$$\beta_{loop}^{lb} = \frac{C_1'}{C_1' + C_2' + C_3' + C_4' + C_5'}. \tag{11}$$

Fig. 5 shows the lower bound of the precision of loop of four β_{loop}^{lb} according to precision β and recall α of pairwise matching when the puzzle dimension is $m_1 = 18, m_2 = 24$. Interestingly, there exist considerable precision and recall region where lower bound of precision of loop of four pairwise matching (β_{loop}^{lb}) is greater than precision of a single pairwise matching (β). It means that once we find loops of four out of combinations of the pair matching candidates with decent precision and recall, the loops of four are shown to be more reliable than even a single pairwise match.

4.2 Importance of Consensus from Neighbors

In this section, we show that a number of positive neighbor matches (consensus from neighbor pieces) is more important than a single strong positive match for a correct piece configuration. When dissimilarity measures between a piece of interest and adjacent pieces are smaller than a threshold (θ), we decide that the neighbors as positives and otherwise are negatives. We first calculate a probability that a piece configuration is correct where ϵ number of the neighbors are negatives and the others are positives. Let us define events,

M_p: A pairwise piece configuration is decided as positive because the piece dissimilarity is below θ.
M_n: A pairwise piece configuration is decided as negative because the piece dissimilarity is above θ.
M_t: A pairwise piece configuration is true.

A true pairwise piece configuration means a ground truth configuration that we aim for. Let us assume that 1) all the pairwise piece configurations are independent events and 2) share the same conditional probabilities, $P(M_t|M_n)$, $P(M_t|M_p)$. When we find a piece configuration which has ϵ number of negative neighbors and $4-\epsilon$ number of positive neighbors, the probability that this configuration is true is,

$$P(M_t|M_n)^{\epsilon} P(M_t|M_p)^{(4-\epsilon)}. \tag{12}$$

Now, we summarize the error of pairwise matching candidates from the initial step of our algorithm by two metrics, precision β (ratio of true positives to positives) and recall α (proportion of true positives retrieved). Let's set ϕ as a number of positive pairs and ν as a number of negative pairs. Then, conditional probabilities are estimated by counting numbers of false negative matches and true positive matches,

$$P(M_t|M_n) = \beta \frac{\phi}{\nu} \frac{1-\alpha}{\alpha} \tag{13}$$

$$P(M_t|M_p) = \beta. \tag{14}$$

Thus, the probability in Equation (12) becomes

$$P(M_t|M_n)^{\epsilon} P(M_t|M_p)^{(4-\epsilon)} = \beta^4 \left(\frac{\phi}{\nu}\right)^{\epsilon} \left(\frac{1-\alpha}{\alpha}\right)^{\epsilon}. \tag{15}$$

We find $\zeta = 10$ positive pairwise matches maximally for each side of the pieces so that the number of positive pairs are $\phi = O(K)$ where K represents a number of puzzle pieces. And, the other possible pairwise piece configurations are all negative so that the number of negative pairs are $\nu = O(K^2)$. Thus,

$$\nu >> \phi. \tag{16}$$

With the fixed and moderate α and β each ranging from $0.1 \sim 0.9$, as a piece configuration includes more negative neighbors, the piece configuration drastically reduces its probability to be a correct configuration due to the term $(\frac{\phi}{\nu})^{\epsilon}$ in the Equation (15).

If we think that one of the 4 neighbors is a strong positive i.e., $P(M_t|M_p) \approx 1$,and the others are negatives, the probability in the Equation (12) becomes

$$P(M_t|M_n)^3 P(M_t|M_p) = \beta^3 \left(\frac{\phi}{\nu}\right)^3 \left(\frac{1-\alpha}{\alpha}\right)^3 \beta', \tag{17}$$

where β' is almost 1. Because this probability is already very small due to the term $(\frac{\phi}{\nu})^3$, the β' does not strongly affect the probability. This implies a single neighbor bond does not heavily increase the probability to be a correct configuration.

4.3 Analysis of Hierarchical Loops

We also show more empirical analysis to support the effectiveness of hierarchical loop constraints. Specifically, we empirically show that the precision rate of pairwise matching tends to rise as we assemble higher-order loops. We observe in Fig. 6 that as higher-order loops are built from smaller order loops the precision of pairwise matches increases significantly even in the presence of noise with dataset from Cho et al. [3] (MIT dataset). In Fig. 6a, for our dissimilarity measure, MGC and SSD+LAB, the precisions are 0.645, 0.627 and 0.5, respectively for local, pairwise piece matches (see loop of order 1) and jump dramatically to 0.951, 0.947 and 0.929 (see loop of order 2) with our assembly method at the lowest order loop (order 2). The precision keeps increasing as order of loops grows eventually reaching 1 for both measures. This tendency of the precision to reach 1 as higher order loops are assembled persists even when the threshold for pairwise matching candidates (θ) is varied (Fig. 6b) and when noise (pixel-wise Gaussian) is added to the puzzle pieces (Fig. 6c). With the various datasets, the precision also reaches 1 as the order of small loops increases (Fig. 6d). For most of the experiments, precision approaches 1 when the order of small loops is above 4 or 5.

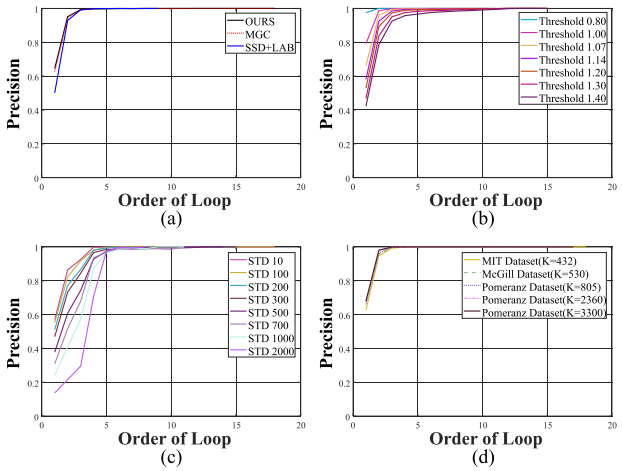


Fig. 6. The precision of pair matches changes as a function of order of hierarchical loops (a) for different dissimilarity measures, (b) at different local matching thresholds, (c) at different noise levels with the MIT dataset, and (d) with different datasets (with the MGC measure). The leftmost point represents the performance of the local matching by itself with no loop constraints. The noise levels in (c) are high magnitude because the input images are 16 bit. The piece size is $S = 28$ pixels.

Notably, although the precision of pair matches is below 15 percent under severe noise (2000 STD), it increases significantly and reaches to 1 as the order of small loops grows.

5 Experiments

We verified our square jigsaw puzzle assembly algorithm with sets of images from Cho et al. [3] (MIT dataset) and Olmos et al. [35] (McGill dataset). Each group of authors provides a set of 20 images. Pomeranz et al. [20] presents 1 set of 20 images (Pomeranz805 dataset) and 2 sets of 3 large images (Pomeranz2360 and Pomeranz3300 dataset). All the data sets are widely used as a benchmark to measure the performance of square jigsaw puzzle solvers. For some images, the camera is perfectly aligned with the horizon line and image edges (e.g., building boundaries) align exactly with puzzle edges. Some patches contain insufficient information (homogeneous region such as sky, water and snow) and others present repetitive texture (man-made textures and windows). As a result, the pairwise dissimilarity measures return many false positives and false negatives on these data sets.

We measured performance using metrics from Cho et al. [3] and Gallagher [21]. *"Direct Comparison"* measures a percentage of pieces that are positioned absolutely correctly. *"Neighbor Comparison"* is a percentage of correct neighbor pairs and *"Largest Component"* is a percentage of image area occupied by the largest group of correctly configured pieces. *"Perfect Reconstruction"* is a binary indicator of whether or not all pieces are correctly positioned with correct rotation.

5.1 Compatibility Measure

For unknown orientation puzzles, a strict accuracy metric is inappropriate to compare the performance of pairwise compatibility measures because the number of true negatives is far larger than the number of positives and false negatives. Chance is vanishingly small and thus accuracy tends to be small in absolute terms. Thus, the accuracy metric does not distinctly show the performance difference between the algorithms. One approach to evaluate the performance of pairwise compatibility measure is precision and recall [18].

We used the MIT, McGill and Pomeranz805 datasets and generated challenging unknown orientation puzzles where the piece size is $S = 28$ pixels and the numbers of pieces are 432, 540 and 805 respectively to evaluate performance of compatibility measures. In Fig. 7, precision and recall curves for the proposed pairwise compatibility measure, MGC [21] and Sum of Squared Distance(SSD) [3] are presented. We also calculate precision and recall values when each side of a piece matches with a single piece with lowest dissimilarity value. Our method reduced precision and recall error by 20 percent compared to MGC. Furthermore, we performed component analysis of our proposed pairwise compatibility measure with the same setup. Fig. 8 shows that our method performs better than the MGC [21] without derivative information. And, with the derivative information, the performance is even more enhanced.

5.2 Known Orientation Puzzles

We compared our algorithm including proposed algorithm without Growing Hierarchical Loops step with the previous

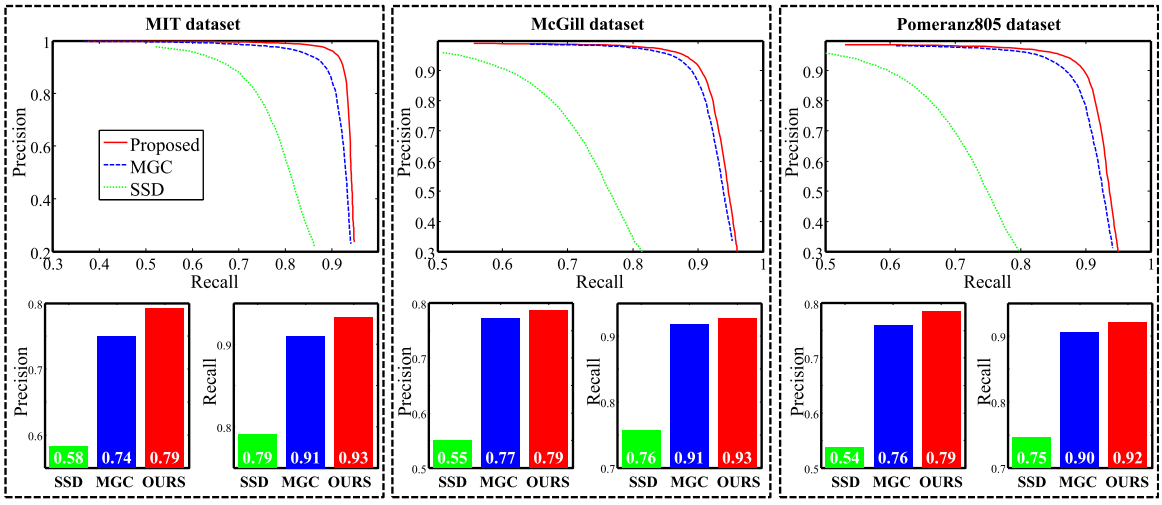


Fig. 7. Pairwise compatibility measure performance comparison. The figures represent precision and recall curves for Sum of Squared Distance (SSD) [3], MGC [21] and proposed measure on unknown orientation piece puzzle, and precision and recall values when each side of the piece possesses a single neighbor which returns lowest dissimilarity for various datasets.

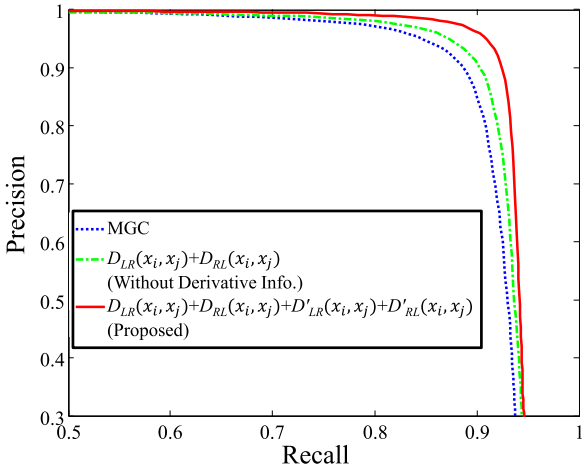


Fig. 8. Component analysis of the proposed compatibility measure. We analyzed the components of our proposed pairwise compatibility measure with unknown orientation piece puzzles from the MIT dataset (the size of piece is $S = 28$, the number of pieces is $K = 432$). Our method performs better than the MGC [21] without derivative information. And, with the derivative information, the performance is enhanced more.

works [3], [19], [20], [21], [22], [23], [25], [30] when orientation of each piece is known and only position of piece is unknown in Table 1. The performance of our algorithm with or without Growing Hierarchical Loops (GL) step on the known orientation piece puzzle is comparable with those of previous works.

5.3 Unknown Orientation Puzzles

Unknown orientation puzzles are a challenging extension of known orientation puzzles. With K pieces, unknown orientation puzzles have 4^K times as many possible configurations as known orientation puzzles. For small puzzles with $K = 432$ this means that unknown orientation puzzles have $4^{432} \approx 1.23 \times 10^{26}$ times as many possible assemblies.

MIT Dataset. Fig. 9 qualitatively compares assembly results on unknown orientation piece puzzles from previous works and our algorithm. The size of each piece is $S = 28$ pixels and the number of pieces in a puzzle is $K = 432$. These images in Fig. 9 are very challenging images from the MIT dataset due to lack of texture or man-made structures aligned with boundary of puzzle pieces. The Gallgher's algorithm [21] and our algorithm without Growing Hierarchical Loops (GL) step struggled to correctly assemble the puzzles (See the red rectangles in Fig. 9) whereas our algorithm successfully assembled the demanding image puzzles. We note that our result of the second image puzzle in Fig. 9 is visually perfect but the direct comparison is not 100 percent due to

TABLE 1
Reconstruction Performance on Known Orientation Piece Puzzles from the MIT, McGill and Pomeranz Datasets

	MIT (432 Pieces)		McGill (540 Pieces)		Pomeranz (805 Pieces)		Pomeranz (2360 Pieces)		Pomeranz (3300 Pieces)	
	Direct	Neighbor	Direct	Neighbor	Direct	Neighbor	Direct	Neighbor	Direct	Neighbor
Cho et al. [3]	10%	55%	-	0%	-	-	-	-	-	-
Yang et al. [19]	69.2%	86.2%	-	-	-	-	-	-	-	-
Pomeranz et al. [20]	94.0%	95.0%	83.5%	90.9%	80.3%	89.7%	33.4%	84.7%	80.7%	85.0%
Andalo et al. [22]	91.7%	94.3%	90.6%	95.3%	82.5%	93.4%	-	-	-	-
Gallagher et al. [21]	95.3%	95.1%	-	-	-	-	-	-	-	-
Sholomon et al. [23]	86.2%	**96.2%**	92.8%	96.0%	94.7%	96.3%	85.7%	88.9%	89.9%	92.8%
Paikin et al. [25]	**96.2%**	95.8%	93.2%	96.1%	92.5%	95.1%	94.0%	96.3%	90.6%	95.3%
Yu et al. [30]	95.8%	95.7%	94.8%	**97.3%**	**95.2%**	**96.6%**	**94.9%**	**97.6%**	94.1%	97.5%
Proposed without GL	95.6%	95.5%	92.2%	95.2%	93.1%	94.9%	94.4%	96.4%	92.0%	96.4%
Proposed	95.8%	95.6%	**95.4%**	97.0%	93.9%	95.5%	94.5%	96.0%	**94.9%**	**97.7%**

The piece size is $S = 28$ pixels.

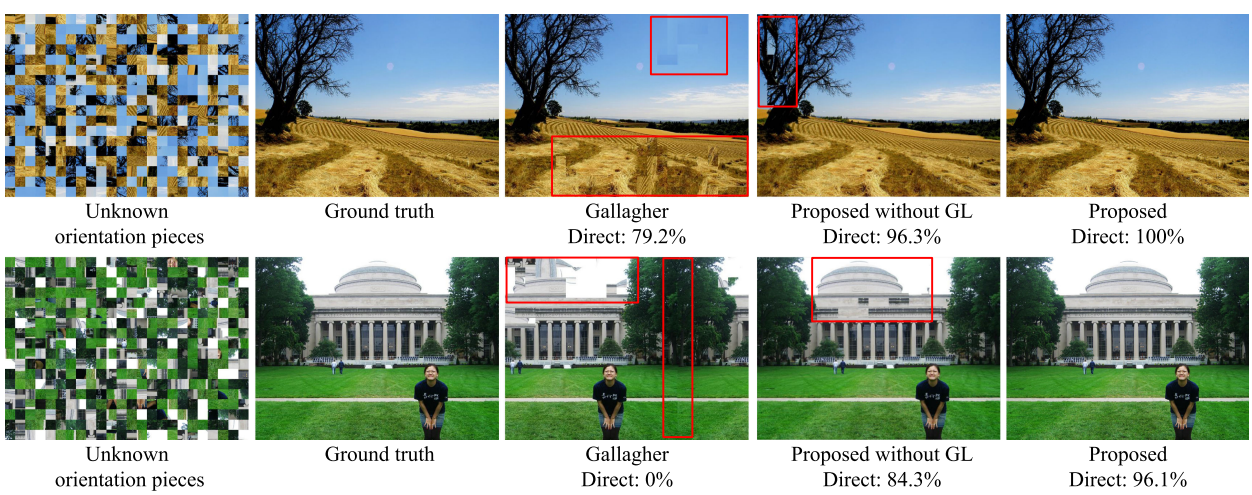


Fig. 9. Qualitative reconstruction performance on unknown orientation piece puzzles. This figure shows assembly results by Gallagher [21], proposed algorithm without Growing Hierarchical Loops step and proposed algorithm. The prior algorithms return false configurations in the red rectangle regions whereas our proposed algorithm correctly assembles those challenging pieces.

TABLE 2
Reconstruction Performance on Unknown Orientation Piece puzzles from the MIT Dataset

	Direct	Neighbor	Largest	Perfect
Gallagher [21]	82.2%	90.4%	88.9%	9
Paikin et al. [25]	95.4%	95.4%	-	-
Yu et al. [30]	95.6%	95.3%	95.6%	**14**
Proposed without GL	94.7%	94.9%	94.6%	12
Proposed	**95.8**%	**95.6**%	**95.8**%	12

$K = 432$, $S = 28$.

the saturated pieces where all the pixel values are maximum so that no information is included in the pieces. We performed quantitative analysis with the same setup (Table 2). Our method reduced reconstruction error from the prior algorithms. We note that the MIT dataset includes 5 images where parts of the images are saturated such as the second image in the Fig. 9. Due to these pieces, it is not possible to reach to 100 percent reconstruction accuracy.

McGill and Pomeranz Datasets. We also tested our algorithm with McGill dataset [35] and Pomeranz datasets [20]. For images from the McGill, Pomeranz805, Pomeranz2360 and Pomeranz3300, we generated $K = 540, 805, 2360, 3300$ numbers of puzzle pieces, respectively, where the orientation of pieces is unknown and the size of the pieces is $S = 28$. Our algorithm outperformed the previous works (Table 3).

Evaluation on various Size and Number of Pieces. As proposed by Gallagher [21], we performed more experiments by varying the size of pieces $S = 7, 10, 14, 28$ pixels and the number of puzzle pieces $K = 221, 432, 1064$ using the MIT dataset. When the size of a piece is small, each piece includes less information so that the compatibility measure is unreliable. Thus, we can evaluate the robustness of the assembly algorithms with weaker guidance from the pairwise compatibility measure. The Fig. 10 shows the reconstruction performance comparisons for each case. It is notable that our proposed method reduced the reconstruction error by up to 75 percent from Gallagher et al. [21] when the size of a piece is $S = 14$. In fact, our proposed algorithm almost maintained the reconstruction performance although the size of a piece was reduced to $S = 14$ whereas the previous works [21] reduced their reconstruction performance.

Algorithm Component Analysis. We first verified that the proposed assembly algorithm increases the size of consensus configurations. Fig. 11 shows largest size (order) of consensus assemblies for each image puzzle (unknown orientation piece puzzle, $S = 28$, $K = 432$) from the MIT dataset before and after Growing Hierarchical Loops step. We observed that our assembly algorithm increased the size of largest consensus assemblies and reached the maximum size of consensus assembly for most of the image puzzles.

We analyzed the contribution of each step in our algorithm. We used two pairwise compatibility measures, MGC [21] and our pairwise compatibility measure (refer to DC), and two assembly strategies, tree-based [21] and our assembly algorithm (refer to consensus-based). From the methods, we formed 4 different assembly algorithms from all combinations of the algorithms. Fig. 12 shows the reconstruction performance comparison on an unknown orientation piece puzzle from images from the MIT dataset ($S = 28$, $K = 432$). Each step of our method contributes the performance improvement.

TABLE 3
Reconstruction Performance on Unknown Orientation Piece Puzzles from the McGill and the Pomeranz Datasets

	McGill dataset ($K = 540$)			
	Direct	Neighbor	Largest	Perfect
Gallagher [21]	72.0%	73.3%	72.8%	7
Yu et al. [30]	**92.8**%	93.3%	-	-
Proposed without GL	89.1%	92.5%	89.0%	10
Proposed	92.3%	**94.5**%	**92.3**%	**11**
	Pomeranz805 dataset ($K = 805$)			
	Direct	Neighbor	Largest	Perfect
Gallagher [21]	83.2%	85.5%	83.5%	5
Yu et al. [30]	91.7%	92.9%	-	-
Proposed without GL	86.4%	88.8%	86.3%	10
Proposed	**92.7**%	**94.4**%	**92.6**%	**11**
	Pomeranz2360 dataset ($K = 2360$)			
	Direct	Neighbor	Largest	Perfect
Gallagher [21]	60.5%	62.5%	60.8%	0
Yu et al. [30]	**94.4**%	95.5%	-	-
Proposed without GL	94.0%	95.3%	94.0%	0
Proposed	**94.4**%	**96.8**%	**94.3**%	**1**
	Pomeranz3300 dataset ($K = 3300$)			
	Direct	Neighbor	Largest	Perfect
Gallagher [21]	80.3%	81.9%	80.2%	**1**
Yu et al. [30]	89.7%	90.2%	-	-
Proposed without GL	89.9%	93.4%	89.9%	**1**
Proposed	**92.1**%	**95.2**%	**92.1**%	**1**

Piece sizes are all $S = 28$.

Various Types of Puzzles. As opposed to prior works [3], [19], [20], [22], [23], our method does not require the dimension of resulting puzzles as an input. This allows us to solve puzzles with no information except the pieces themselves so that our algorithm can solve multiple images with missing pieces (Fig. 13). For larger puzzles, our solver perfectly assembles puzzles where the number of pieces are very large, 1900, 4108 and 9801 pieces from data [36] (Fig. 14).

Computational Complexity. The complexity for searching all loops of four is $O(\zeta^3\phi)$, where ζ is the maximum number of positive pair matches that one side of a piece can have and ϕ is a number of pair matching candidates. ζ is normally from 1 to 3 and maximally 10 in our experiments and each operation is just an indexation of a binary matrix. The average time for finding all loops of four is 0.308 second with the MIT dataset (432-piece unknown orientation puzzle) in Matlab. Most of the time is spent in pairwise matching, unoptimized merging, trimming and filling steps. Using MGC, our algorithm spends 3 minute for 432 pieces and 25.6 hours for 9801 pieces of unknown orientation on a modern PC.

6 Conclusion

We present a novel computational puzzle solver for square piece image jigsaw puzzles with no prior information such as piece orientation, anchor pieces, or resulting dimension of the puzzles. When the size of pieces becomes smaller so that pairwise compatibility measures become unreliable,

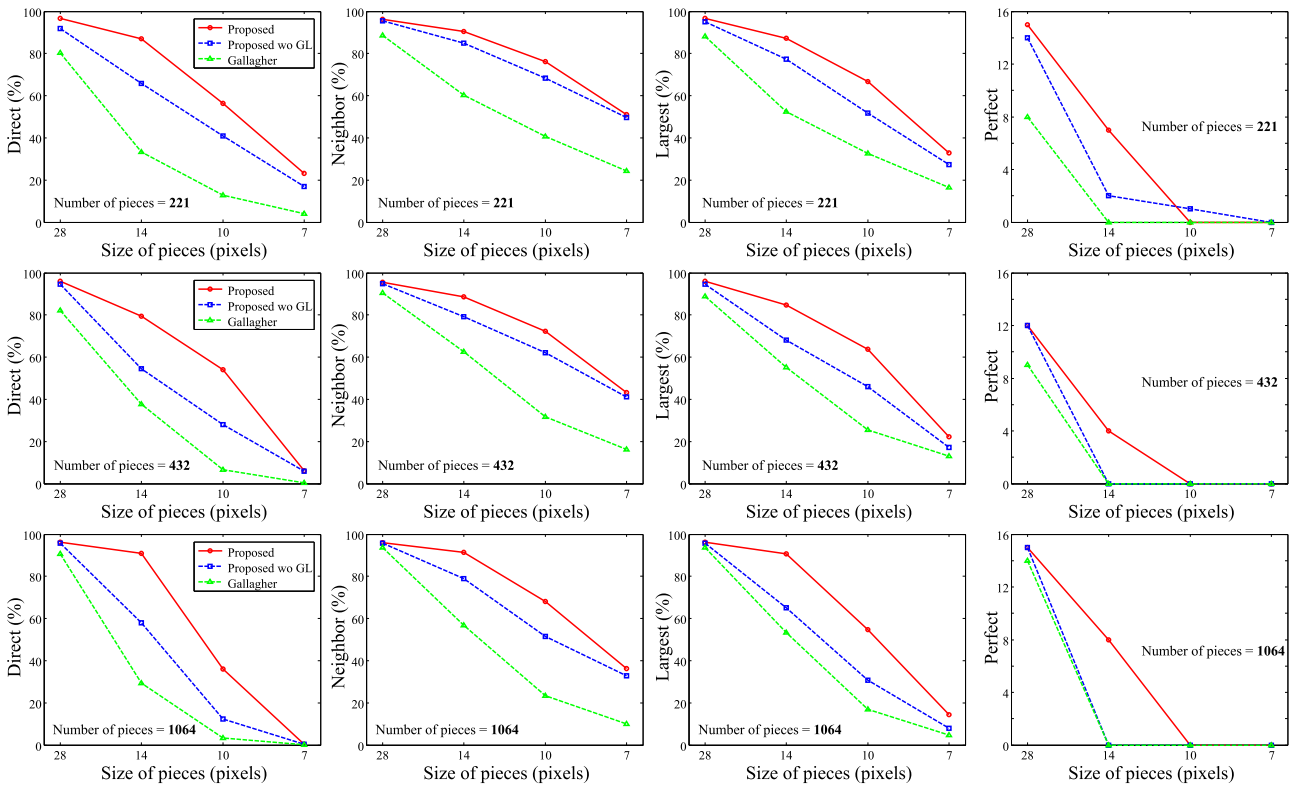


Fig. 10. Reconstruction performance of unknown orientation piece puzzle with various piece size and number of pieces. The size of pieces are $S = 28, 14, 10, 7$ pixels. The number of puzzle pieces are $K = 221, 432, 1064$ from the MIT dataset. When the size of a piece is small $S = 14$ pixels, our algorithm reduces reconstruction error up to 75 percent.

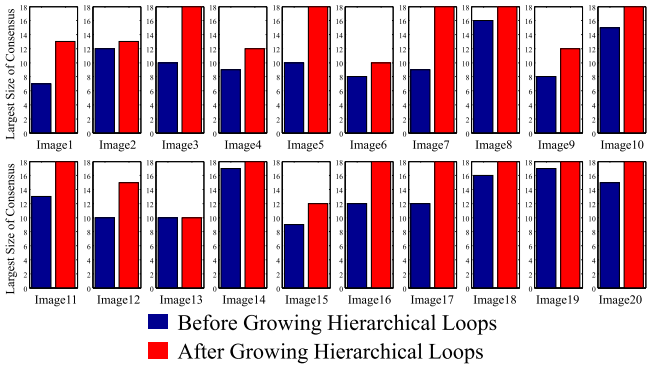


Fig. 11. Analysis of proposed assembly algorithm. We analyzed our proposed assembly algorithm with unknown orientation piece puzzles from the MIT dataset ($S = 28$, $K = 432$) [3]. This figure shows the largest size (order) of consensus assemblies before and after Growing Hierarchical Loops step. Since the resulting dimension of the puzzle is 24 by 18 pieces, the possible maximum size of hierarchical loops is 18 by 18 pieces. Our assembly algorithm configures the possible maximum size of consensus assembly on most of the puzzles from the MIT dataset.

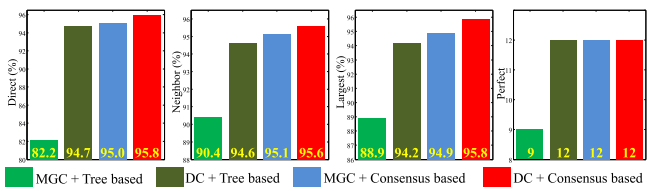


Fig. 12. Algorithm component analysis. MGC and tree-based are a pairwise compatibility measure and an assembly strategy proposed by Gallagher [21]. DC and consensus-based refer to our proposed pairwise compatibility measure and assembly strategy respectively. The experiments were performed on unknown orientation piece puzzles from the MIT dataset with $S = 28$ and $K = 432$.

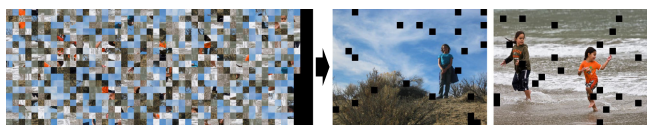
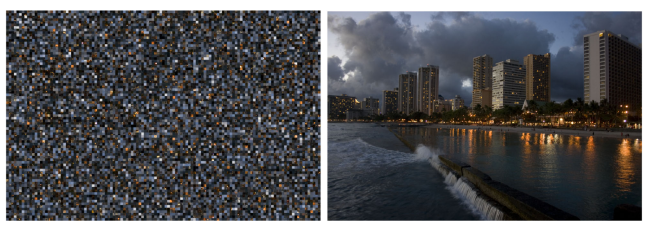


Fig. 13. Mixed, missed and unknown orientation piece puzzle. Puzzles are from two images and 5 percent of pieces are missing.

(a) 1900 mixed unknown orientation pieces and outputs

(b) 4108 unknown orientation pieces and an output



(c) 9801 unknown orientation pieces and an output

Fig. 14. Reconstructions on mixed unknown orientation puzzles and very large unknown orientation puzzle ($S = 28$).

our algorithm achieves higher reconstruction accuracy than do state of the art methods. The strength of our algorithm derives from its reduced dependency on a pairwise compatibility measure, which is often erroneous, and instead exploits geometric information from reliable neighbor configurations for puzzle assembly. In addition, a more accurate pairwise compatibility measure which utilizes directional derivative information along adjoining boundaries of the pieces is proposed and used.

Appendix
Derivation of Expected Numbers of Loops

In this Appendix we derive the expected numbers of true (C_1) or false (C_1, C_2, C_3, C_4) loops of four pieces out of pairwise matching candidates. When the true resulting dimension of unknown orientation of jigsaw puzzle is $m_1 \times m_2$, and the pairwise matching results are summarized with precision β and recall α, the numbers of true $|T|$, true positive $|T_p|$, false negative $|F_n|$, and false positive $|F_p|$ pairwise matches are calculated below,

$$|T| = m_1(m_2 - 1) + m_2(m_1 - 1) \; < \; 2m_1m_2 \tag{18}$$

$$|T_p| = \alpha|T| \; < \; 2\alpha m_1 m_2 \tag{19}$$

$$|F_n| = (1-\alpha)|T| \; < \; 2(1-\alpha)m_1m_2 \tag{20}$$

$$|F_p| = \frac{\alpha(1-\beta)}{\beta}|T| \; < \; 2\frac{\alpha(1-\beta)}{\beta}m_1m_2. \tag{21}$$

The expected number of true loops is

$$C_1 \geq \max((m_1 - 1)(m_2 - 1) - 2|F_n|, 0). \tag{22}$$

Initially, there exist $(m_1 - 1)(m_2 - 1)$ true loops which are made up of true pair matches without missing true pairs (false negative). A single false negative maximally removes two loops of four pieces so that the minimum number of the C_1 is $(m_1 - 1)(m_2 - 1) - 2|F_n|$ and the C_1 should be non-negative. We derive Inequality 6 ,when $|F_n|$ in Inequality 22 is substituted with Inequality 20

The expected number of false loops of four pieces where the two adjacent pair matches in a loop are true positive and the others are false positive matches, case C_2 in Fig. 4 is below,

$$C_2 \leq |T_p||F_{ps}|^2 P_m. \tag{23}$$

There are maximally $|T_p|$ number of two of adjacent true pairwise matching and each two of adjacent true pairs has $|F_{ps}|^2$ number of chances to form a false loop maximally with probability P_m.

$|F_{ps}|$ is the expected number of false pairs on each edge of the pieces,

$$|F_{ps}| = \frac{|F_p|}{2m_1m_2} \; < \; \frac{\alpha(1-\beta)}{\beta}. \tag{24}$$

$|F_{ps}|$ is calculated by dividing the total number of false positive matches $|F_p|$ by the number of all edges $4m_1m_2$. We divide $2|F_p|$ by $4m_1m_2$ because a false pair match needs two edges. Here, we assume that false positive matches are uniformly distributed on the edges of the pieces.[4] This assumption can be achieved by limiting a maximum number of matches that each edge of the pieces can have when we perform Local, Pairwise Matching step. In our method, we set the limitation as $|F_{ps}| \; < \; \zeta = 10$.

For the each chance, the probability to form a false loop is same as to the probability that an edge forms a wrong pair with a specific edge,

$$P_m = \frac{1}{4(m_1m_2 - 1) - 1} \; < \; \frac{1}{3m_1m_2}, \tag{25}$$

where the Inequality stands when $m_1m_2 > 5$. There are $4(m_1m_2 - 1)$ possible edges to form a wrong pair. Since we know that the pair is false, we should avoid the true edge of a piece. Here, we assume that the probability that an edge forms a pair with any other wrong edge is all the same. We derive Inequality 7 when we substitute terms in Inequality 23 with Inequality 19, 24 and 25. The other cases C_3, C_4 and C_5 are analyzed in the similar way

Acknowledgments

Kilho Son and David B. Cooper acknowledge partial support from NSF Grant # 0808718.

References

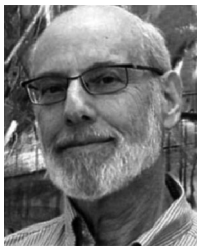
[1] *Oxford Dictionary of English*, 3rd ed. London, U.K.: Oxford Univ. Press, 2010.
[2] K. Kurihara, M. Kikuchi, S. Imaizumi, S. Shiota, and H. Kiya, "An encryption-then-compression system for JPEG/motion JPEG standard," *IEICE Trans. Fundamentals Electron. Commun. Comput. Sci.*, vol. E98.A, no. 11, pp. 2238–2245, 2015.
[3] T. S. Cho, S. Avidan, and W. T. Freeman, "A probabilistic image jigsaw puzzle solver," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2010, pp. 183–190.
[4] K. Son, E. B. Almeida, and D. B. Cooper, "Axially symmetric 3D pots configuration system using axis of symmetry and break curve," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2013, pp. 257–264.
[5] G. Oxholm and K. Nishino, "Reassembling thin artifacts of unknown geometry," in *Proc. Int. Symp. Virtual Reality Archaeology Cultural Heritage*, pp. 49–56, 2011.
[6] L. Zhu, Z. Zhou, and D. Hu, "Globally consistent reconstruction of ripped-up documents," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 1, pp. 1–13, Jan. 2008.
[7] H. Liu, S. Cao, and S. Yan, "Automated assembly of shredded pieces from multiple photos," in *Proc. IEEE Int. Conf. Multimedia Expo*, 2010, pp. 358–363.
[8] P. Liu, "A system for computational analysis and reconstruction of 3D comminuted bone fractures," Ph.D. dissertation, Electrical and Computer Engineering, The University of North Carolina, Chapel Hill, NC, 2012.
[9] T. S. Cho, S. Avidan, and W. T. Freeman, "The patch transform," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 8, pp. 1489–1501, Aug. 2010.
[10] H. Kato and T. Harada, "Image reconstruction from bag-of-visual-words," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2014, pp. 955–962.
[11] E. M. Airoldi, D. M. Blei, S. E. Fienberg, and E. P. Xing, "Mixed membership stochastic blockmodels," *J. Mach. Learn. Res.*, vol. 9, pp. 1981–2014, Jun. 2008. [Online]. Available: http://dl.acm.org/citation.cfm?id=1390681.1442798
[12] A. Goldenberg, A. X. Zheng, S. E. Fienberg, and E. M. Airoldi, "A survey of statistical network models," *Found. Trends Mach. Learn.*, vol. 2, no. 2, pp. 129–233, Feb. 2010. [Online]. Available: http://dx.doi.org/10.1561/2200000005

4. For example, if there exist 8 false pair matches among 4 pieces, each edge of the piece has 1 false match by the assumption. This is because 8 false pair matches need 16 edges and the 16 edges evenly share the false matches.

[13] A. L. Barabási and Z. N. Oltvai, "Network biology: Understanding the cell's functional organization," *Nature Genetics*, vol. 5, pp. 101–114, 2013.
[14] C. Zach, M. Klopschitz, and M. Pollefeys, "Disambiguating visual relations using loop constraints," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2010, pp. 1426–1433.
[15] Z. Kalal, K. Mikolajczyk, and J. Matas, "Forward-backward error: Automatic detection of tracking failures," in *Proc. 20th Int. Conf. Pattern Recognit.*, 2010, pp. 2756–2759.
[16] T. Dekel, S. Oron, S. Avidan, M. Rubinstein, and W. Freeman, "Best buddies similarity for robust template matching," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 2021–2029.
[17] H. Freeman and L. Garder, "Apictorial jigsaw puzzles: The computer solution of a problem in pattern recognition," *Electron. Comput.*, vol. 13, pp. 118–127, 1964.
[18] N. Alajlan, "Solving square jigsaw puzzles using dynamic programming and the hungarian procedure," *Amer. J. Appl. Sci.*, vol. 5, no. 11, pp. 1941–1947, 2009.
[19] X. Yang, N. Adluru, and L. J. Latecki, "Particle filter with state permutations for solving image jigsaw puzzles," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2011, pp. 2873–2880.
[20] D. Pomeranz, M. Shemesh, and O. Ben-Shahar, "A fully automated greedy square jigsaw puzzle solver," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2011, pp. 9–16.
[21] A. C. Gallagher, "Jigsaw puzzles with pieces of unknown orientation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2012, pp. 382–389.
[22] S. Goldenstein, Fernanda A. Andalo, and G. Taubin, "Solving image puzzles with a simple quadratic programming formulation," in *Proc. Conf. Graph. Patterns Images*, 2012, pp. 63–70.
[23] D. Sholomon, O. David, and N. Netanyahu, "A genetic algorithm-based solver for very large jigsaw puzzles," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2013, pp. 1767–1774.
[24] K. Son, J. Hays, and D. B. Cooper, "Solving square jigsaw puzzles with loop constraints," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 32–46.
[25] G. Paikin and A. Tal, "Solving multiple square jigsaw puzzles with missing pieces," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 4832–4839.
[26] K. Son, D. Moreno, J. Hays, and D. B. Cooper, "Solving small-piece jigsaw puzzles by growing consensu," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 1193–1201.
[27] A. Willis and D. B. Cooper, "Computational reconstruction of ancient artifacts," *IEEE Signal Process. Mag.*, vol. 25, no. 4, pp. 65–83, Jul. 2008.
[28] E. D. Demaine and M. L. Demaine, "Jigsaw puzzles, edge matching, and polyomino packing: Connections and complexity," *Graphs Combinatorics*, vol. 23 (Supplement), pp. 195—208, Jun. 2007.
[29] J. Joseph and B. Kruskal, "On the shortest spanning subtree of a graph and the traveling salesman problem," *Proc. Amer. Math. Soc.*, vol. 7, pp. 48–50, 1956.
[30] R. Yu, C. Russell, and L. Agapito, "Solving jigsaw puzzles with linear programming," in *Proc. Brit. Mach. Vis. Conf.*, Sep. 2016, pp. 139.1–139.12. [Online]. Available: https://dx.doi.org/10.5244/C.30.139
[31] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. New York, NY, USA: Cambridge Univ. Press, 2003.
[32] R. Raguram, O. Chum, M. Pollefeys, J. Matas, and J. Frahm, "USAC: A universal framework for random sample consensus," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 2022–2038, Aug. 2013.
[33] N. Snavely, S. M. Seitz, and R. Szeliski, "Photo tourism: Exploring photo collections in 3D," *ACM Trans. Graph.*, vol. 25, no. 3, pp. 835–846, Jul. 2006. [Online]. Available: http://doi.acm.org/10.1145/1141911.1141964
[34] A. Chakrabarti, Y. Xiong, S. J. Gortler, and T. Zickler, "Low-level vision by consensus in a spatial hierarchy of regions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2015, pp. 4009–4017.
[35] A. Olmos and F. A. A. Kingdom, "A biologically inspired algorithm for the recovery of shading and reflectance images," *Perception*, vol. 33, pp. 1463–1473, 2004.
[36] (2013). [Online]. Available: http://cdb.paradice-insight.us/

Kilho Son received the BS degree in electronics from Sogang University, in 2006, the MS degree in electrical engineering from the Korea Advanced Institute of Science and Technology, in 2008, and the PhD degree in computer engineering from Brown University, in 2016. From 2008 to 2010, he worked for Agency for Defense Development in South Korea as a researcher. He is now a computer scientist at Stanford Research Institute International. His research interests include algorithms for computer vision, machine learning, robotics, and artificial intelligence. He is a member of the IEEE.

James Hays received the BS degree in computer science from the Georgia Institute of Technology, in 2003, and the PhD degree from Carnegie Mellon University, in 2009. Following a postdoc with Massachusetts Institute of Technology, he was an assistant professor with Brown University for six years. He is now an associate professor in the School of Interactive Computing at Georgia Tech. His research interests span computer vision, computer graphics, robotics, and machine learning. He is the recipient of the NSF CAREER award (2012) and Sloan Fellowship (2015). He is a member of the IEEE.



David B. Cooper received the BSc and ScM degrees in electrical engineering from the Massachusetts Institute of Technology (MIT) under the industrial co-op program, in 1957, and the PhD degree in applied mathematics from Columbia University, New York, in 1966. From 1957 to 1966, he worked for Sylvania Electric Products, Inc., Mountain View, California and then for the Raytheon Co., Waltham, Massachusetts, on communications and radar systems analysis. Since 1966, he has been a professor of engineering at Brown University, Providence, Rhode Island. He was the cofounder and the associate director of the Brown University Laboratory for Engineering Man/Machine Systems (LEMS) from 1982 through 1997 and served on the executive committee of the Division of Engineering, Brown University for a few years. His more recent research interests have been on use of algebraic 2D curves and 3D surfaces for shape representation, 2D and 3D object estimation and recognition from many unreliable fragments of information obtained from video or LADAR, geometric learning, ad hoc sensor networks specifically the theory for how to use 1,000 cameras, and various data analysis problems for extraction of information from fragments found at archaeological sites. He was named a fellow of the IEEE for his research contributions, among the earliest papers, on unsupervised learning, combined supervised and unsupervised learning, and the Bayesian approach to image boundary estimation, segmentation, 3D stereo reconstruction, and optimally combining unreliable pieces of information in 3D.