A FULLY-AUTOMATED SOLVER FOR MULTIPLE SQUARE JIGSAW PUZZLES USING HIERARCHICAL CLUSTERING

A Thesis

Presented to

The Faculty of the Department of Computer Science

San Jose Staté University

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

by

Zayd Hammoudeh

December 2016

The Designated Thesis Committee Approves the Thesis Titled

A FULLY-AUTOMATED SOLVER FOR MULTIPLE SQUARE JIGSAW PUZZLES USING HIERARCHICAL CLUSTERING

by

Zayd Hammoudeh

APPROVED FOR THE DEPARTMENT OF COMPUTER SCIENCE

SAN JOSE STATE UNIVERSITY

December 2016

Dr. Chris Pollett	Department of Computer Science
Dr. Thomas Austin	Department of Computer Science
Dr. Teng Moh	Department of Computer Science

ABSTRACT

A Fully-Automated Solver for Multiple Square Jigsaw Puzzles Using Hierarchical Clustering

by Zayd Hammoudeh

This paper is very abstract.

DEDICATION

To my mother.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

Introduction

Jigsaw puzzles were first introduced in the 1760s when they were made from wood; their name derives from the jigsaws that were used to carve the wooden pieces. The 1930s saw the introduction of the modern jigsaw puzzle where an image was printed on a cardboard sheet that was cut into a set of interlocking pieces [1, 2]. Although jigsaw puzzles had been solved by children for two centuries, it was not until 1964 that the first automated jigsaw puzzle solver was proposed by Freeman & Gardner [3]. While an automated jigsaw puzzle solver may seem trivial, the problem has been shown by Altman [4] and Demaine & Demaine [5] to be strongly NP-complete when pairwise compatibility between pieces is not a reliable metric for determining adjacency.

Jig swap puzzles are a specific type of jigsaw puzzle where all pieces are equally sized, non-overlapping squares. Jig swap puzzles are substantially more challenging to solve since piece shape cannot be considered when determining affinity between pieces. Rather, only the image information on each individual piece is used when solving the puzzle.

Solving a jigsaw puzzle simplifies to reconstructing an object from a set of component pieces. As such, techniques developed for jigsaw puzzles can be generalized to many practical problems. Examples where jigsaw puzzle solving strategies have been used include: reassembly of archaeological artifacts [6, 7], forensic analysis of deleted files [8], image editing [9], reconstruction of shredded documents [10], DNA fragment reassembly [11], and speech descrambling [12]. In most of these practical applications, the original, also known as "ground-truth," input is unknown. This significantly increases the difficulty of the problem as the structure of the complete solution must be determined solely from the bag of component pieces.

This thesis outlines the first fully-automated jigsaw puzzle solver algorithm for multiple puzzles; unlike all previous solvers, this thesis' implementation is provided no no makes no assumptions regarding the set of input pieces, including the number of ground-truth (i.e., original) puzzles. What is more, it defines a set of new metrics specifically tailored to quantify the quality of outputs of multi-puzzle solvers.

CHAPTER 2

Previous Work

Computational jigsaw puzzle solvers have been studied since the 1960s when Freeman & Gardner proposed a solver that relied only on piece shape and could puzzles with up to nine pieces [3]. Since then, the focus of research has gradually shifted from traditional jigsaw puzzles to jig swap puzzles.

Cho *et al.* [13] proposed in 2010 one of the first modern computational jig swap puzzle solvers; their approach relied on a graphical model built around a set of one or more "anchor piece(s)," which are pieces whose position is fixed in the correct location before the solver began. Cho *et al.*'s solver required that the user specify the puzzle's actual dimensions. Future solvers would improve on Cho *et al.*'s results while simultaneously reducing the amount of information (beyond the set of pieces) passed to the solver.

A significant contribution of Cho *et al.* is that they were first to use the LAB (Lightness and the A/B opponent color dimensions) colorspace to encode image pixels. LAB was selected due to its property of normalizing the lightness and color variation across all three pixel dimensions. Cho *et al.* also proposed a measure for quantifying the pairwise distance between two puzzle pieces that became the basis of most of the future work (see Section **??**).

Pomeranz *et al.* [14] proposed an iterative, greedy jig swap puzzle solver in 2011. Their solver did not rely on anchor pieces, and the only information passed to the solver were the pieces, their orientation, and the size of the puzzle. Pomeranz *et al.* also generalized and improved on Cho *et al.*'s piece pairwise distance measure by proposing a “predictive distance measure.” Finally, Pomeranz *et al.* introduced the concept of “best buddies,” which are any two pieces that are more similar to each other than they are to any other piece. Best buddies have served as both an estimation metric for the quality of solver result as well as the foundation of some solvers’ placers [15].

An additional key contribution of Pomeranz *et al.* is the creation of three image benchmarks. The first benchmark is comprised of twenty 805 piece images; the sizes of the images in the second and third benchmarks are 2,360 and 3,300 pieces respectively.

In 2012, Gallagher [16] formally categorized jig swap puzzles into four primary types. The following is Gallagher’s proposed terminology; his nomenclature is used throughout this thesis.

- **Type 1 Puzzle**: The dimensions of the puzzle (i.e., the width and height of the ground-truth image in number of pixels) is known. The orientation of each piece is also known, which means that there are exactly four pairwise relationships between any two pieces. A single anchor piece, with a known, correct, location is required with additional anchor pieces being optional. This type of puzzle is used by [13, 14].

- **Type 2 Puzzle**: This is an extension of a Type 1 puzzle, where pieces may be rotated in 90° increments (e.g., 0°, 90°, 180°, or 270°); in comparison to a Type 1 puzzle, this change alone increases the number of possible solutions by a factor of 4^n, where n is the number of puzzle pieces. What is more, no piece locations are known in advance; this change eliminates the use of anchor piece(s). Lastly, the dimensions of the ground-truth image may be unknown.

- **Type 3 Puzzle**: All puzzle piece locations are known and only the rotation of the pieces is unknown. This is the least computationally complex of the puzzle variants and is generally considered the least interesting. Type 3 puzzles are not explored as part of this thesis.

- **Mixed-Bag Puzzles**: The input set of pieces are from multiple puzzles, or there are extra pieces in the input set that belong to no puzzle. The solver may output either a single, merged puzzle, or it may separate the input pieces into disjoint sets that ideally align the set of ground-truth puzzles. This type of puzzle is the primary focus of this thesis.

Sholomon *et al.* [17] in 2013 proposed a genetic algorithm based solver for Type 1 puzzles. By moving away from the greedy approach used by Pomeranz *et al.*, Sholomon *et al.*'s approach is more immune to suboptimal decisions early in the placement process. Sholomon *et al.*'s algorithm is able to solve puzzles of significantly larger size than previous techniques (e.g., greater than 23,000 pieces). What is more, Sholomon *et al.* defined three new large image (e.g., 5,015, 10,375, and 22,834 piece) benchmarks [18].

Paikin & Tal [15] published in 2015 a greedy solver that handles both Type 1 and Type 2 puzzles, even if those puzzles are missing pieces. What is more, their algorithm is one of the first to support solving Mixed-Bag Puzzles. Paikin & Tal's algorithm is used as the basis for much of this thesis and is discussed in significant depth in Section **??**.

CHAPTER 3

Quantifying the Quality of a Solver Output

Modern jig swap puzzle solvers are not able to perfectly reconstruct the ground-truth input in most cases. As such, quantifiable metrics are required to objectively compare the quality of outputs from different solvers. Cho *et al.* [13] defined two such metrics namely: direct accuracy and neighbor accuracy. These metrics have been used by others including [17, 14, 15, 19, 16]. This section describes the existing quality metrics, their weaknesses, and proposes enhancements to these metrics to make them more meaningful for Type 2 and Mixed-Bag puzzles.

In the final subsection, tools developed as part of this thesis to visualize the solver output quality are discussed.

3.0.1 Direct Accuracy

Direct accuracy is a relatively naïve quality; it is defined as the fraction of pieces placed in the same location in the ground-truth (i.e., original) and solved image with respect to the total number of pieces. Equation (1) shows the formal definition of direct accuracy (DA), where n is the total number of pieces and c is the number of pieces placed in their original (i.e., correct) location.

$$DA = \frac{c}{n} \tag{1}$$

Direct accuracy is vulnerable to shifts in the solved image where even a few misplaced pieces can cause a significant decrease in accuracy. As shown in Section 3.0.1.2, this can be particularly true when the ground-truth image's dimensions are not known by the solver.

This thesis proposes two new direct accuracy metrics namely: Enhanced Direct Accuracy Score (EDAS) and Shiftable Enhanced Direct Accuracy Score (SEDAS). They are described in the following two subsections; the complementary relationship between EDAS and SEDAS is described in the third subsection.

3.0.1.1 Enhanced Direct Accuracy Score

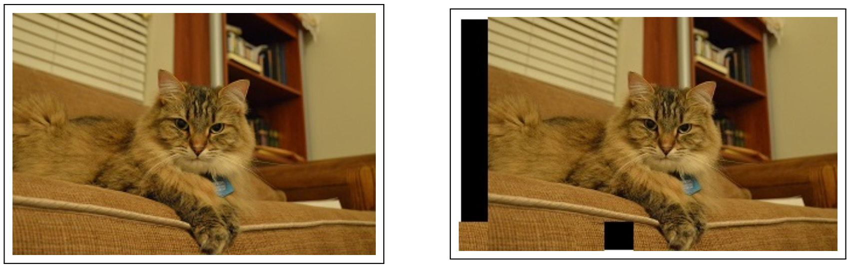
The standard direct accuracy metric does not account for the possibility that there may be pieces from multiple puzzles in the same solver output. For a given a puzzle P_i in the set of input puzzles P (where $P_i \in P$) and a set of solved puzzles S where S_j is in S, Enhanced Direct Accuracy Score (EDAS) is defined as shown in Equation (2).

$$EDAS_{P_i} = \underset{S_j \in S}{\arg\max} \frac{c_{i,j}}{n_i + \sum_{k \neq i}(m_{k,j})} \tag{2}$$

c_{ij} is the number of pieces from input puzzle P_i correctly placed (with no rotation for Type 2 puzzles) in solved puzzle S_j while n_i is the number of pieces in puzzle P_i. $m_{k,j}$ is the number of pieces from an input puzzle P_k (where $k \neq i$) that are also in S_j.

When solving only a single puzzle, EDAS and standard direct accuracy as defined Equation (1) are equivalent. When solving multiple puzzles simultaneously, EDAS necessarily marks as incorrect any pieces from P_i that are not in S_j by dividing by n_i. What is more, the summation of $m_{k,j}$ in EDAS is used to penalize for any puzzle pieces not from P_i. Combined, these two factors enable EDAS to penalize for both extra and misplaced pieces.

It is important to note that EDAS is a score and not a measure of accuracy.



(a) Ground-Truth Image (b) Solver Output

Figure 1: Solver Output where a Single Misplaced Piece Catastrophically Affects the Direct Accuracy

While its value is bounded between 0 and 1 (inclusive), it is not specifically defined as the number of correct placements divided by the total number of placements since the denominator of Equation (2) is greater than or equal to the number of pieces in both P_i and S_j.

3.0.1.2 Shiftable Enhanced Direct Accuracy Score

As mentioned previously, the direct accuracy decreases if there are shifts in the solved image. In many cases such, direct accuracy is overly punitive.

Figure 1 shows a ground-truth image and an actual solver output when the puzzle boundaries were not fixed. Note that only a single piece is misplaced; this shifted all other pieces to the right one location causing the direct accuracy to drop to zero. Had this same piece been misplaced along either the right or bottom side of the image, the direct accuracy would have been largely unaffected. The fact that direct accuracy can give such vastly differing results for essentially the same error shows that direct accuracy has a serious flaw. This thesis proposes Shiftable Enhanced Direct Accuracy Score (SEDAS) to address the often misleadingly punitive nature of direct accuracy.

Let d_{min} be the Manhattan distance between the upper left corner of the solved image and the nearest placed puzzle piece. Also let L be the set of all puzzle piece locations within distance d_{min} of the upper left corner of the image. Given that l is a location in L that is used as the reference point for determining the absolute location of all pieces, then SEDAS is defined as shown in Equation (3).

$$SEDAS_{P_i} = \arg\max_{l \in L} \left(\arg\max_{S_j \in S} \frac{c_{i,j,l}}{n_i + \sum_{k \neq i}(m_{k,j})} \right) \tag{3}$$

In the standard definition of direct accuracy proposed by Cho *et al.*, l is fixed at the upper left corner of the image. In contrast, SEDAS shifts this reference point within a radius of the upper left corner of the image in order to find a more meaningful value for direct accuracy.

Rather than defining SEDAS based off the distance d_{min}, an alternative approach is to use the point anywhere in the image that maximizes Equation (3). However, that approach can be significantly more computationally complex in particular in large puzzles with several thousand pieces. Hence, this thesis' approach balances finding a meaningful direct accuracy score with computational efficiency.

3.0.1.3 The Necessity to Use Both EDAS and SEDAS

While EDAS can be misleadingly punitive, it cannot be wholly replaced by SEDAS. Rather, EDAS and SEDAS serve complementary roles. First, EDAS must necessarily be calculated as part of SEDAS since the upper left corner location is inherently a member of the set L. Hence, there is no additional time required to calculate EDAS. What is more, by continuing to use EDAS along with SEDAS, some shifts in the solved image may be quantified; this would not be possible if SEDAS was used alone.

3.0.2 Neighbor Accuracy

Cho *et al.* [13] defined neighbor accuracy as the ratio of the number of puzzle piece sides adjacent to the same piece's side in both the ground-truth and solved image versus the total number of puzzle piece sides. Formally, let q be the number of sides each piece has (i.e., four in a jig swap puzzle) and n be the number of pieces. If a is the number of puzzle piece sides adjacent in both the ground-truth and solved images, then the neighbor accuracy, NA, is defined as shown in Equation (4).

$$NA = \frac{a}{n\ q} \tag{4}$$

Unlike direct accuracy, neighbor accuracy is largely unaffected by shifts in the solved image since it considers only a piece's neighbors and not its absolute location. However, the standard definition of neighbor accuracy cannot encompass the case where pieces from multiple input puzzles may be present in the same solver output.

3.0.2.1 Enhanced Neighbor Accuracy Score

Enhanced Neighbor Accuracy Score (ENAS) improves the neighbor accuracy metric by providing a framework to quantify the quality of Mixed-Bag solver outputs.

Let n_i be the number of puzzles pieces in the input puzzle P_i and $a_{i,j}$ be the number of puzzle piece sides adjacent in P_i and S_j. If $m_{k,j}$ is the number of puzzle pieces from an input puzzle P_k (where $k \neq i$) in S_j, then the ENAS for P_i is defined as shown in Equation (5).

$$ENAS_{P_i} = \operatorname*{arg\,max}_{S_j \in S} \frac{a_{i,j}}{q\ \left(n_i + \sum_{k \neq i}(m_{k,j}\right)} \tag{5}$$

In the same fashion as the technique described for EDAS in Section 3.0.1.1, ENAS divides by the number of pieces n_i in input puzzle P_i. By doing so, it effectively marks as incorrect any pieces from P_i that are not in S_j. What is more, by including a summation of all $m_{k,j}$ in the denominator of (5), ENAS marks as incorrect any pieces not from P_i that are in S_j. The combination of these two factors allows ENAS to account for extra and misplaced pieces.

3.0.3 Visualizing Solver Output Quality

In images with thousands of pieces, it is often difficult to visually determine the location of individual pieces that are incorrectly placed. What is more, visual tools help developers quickly detect and fix latent bugs.

The following two subsections describe the tools developed as part of this thesis for visualizing direct accuracy and neighbor accuracy.

3.0.3.1 Visualizing EDAS and SEDAS

In standard direct accuracy, EDAS, and SEDAS, each puzzle piece is assigned a single value (i.e., correct or incorrect). Due to that, the direct accuracy visualization represents each puzzle by a square filled with a solid color. One additional refinement used in this thesis is to subdivide the “incorrect” placements into a set of subcategories; they are (in order of precedence): wrong puzzle, wrong location, and wrong rotation. Table 1 shows the colors assigned to puzzle pieces depending on their direct accuracy classification.

Wrong Puzzle	Wrong Location	Wrong Rotation	Correct Location	No Piece Present

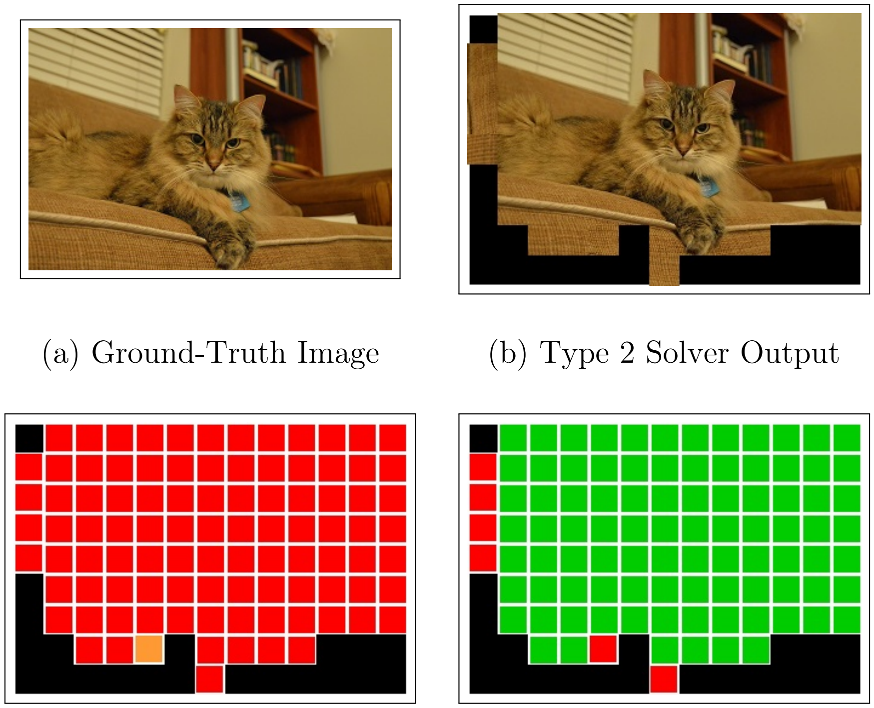
Table 1: Color Scheme for Puzzles Pieces in Direct Accuracy Visualizations

Figure 2 shows a Type 2 solver output along with the associated EDAS and SEDAS visualizations. Since four puzzle pieces were erroneously placed on the left of the image, almost all pieces had the wrong location according to EDAS; the only exception is a single piece that had the right location but wrong rotation. In contrast, almost all pieces have the correct location in the SEDAS representation; note that the piece in the correct location but with wrong rotation in EDAS has the wrong location in SEDAS.

3.0.3.2 Visualizing ENAS

The visualization for neighbor accuracy is very similar to the techniques described in Section **??** for visualizing best buddies where each puzzle piece is divided into four equal-sized isosceles triangles (one for each side). The triangles are assigned colors depending on whether their neighbors in the solver output and ground-truth image match. The visualization includes a subcategory known as "wrong puzzle" which is a special case that occurs when solving Mixed-Big puzzles and some of the pieces in the solved puzzle are not from the puzzle of interest, P_i. Table 2 defines the colors used to represent the different classifications of puzzle piece sides in neighbor accuracy visualizations.

Figure 3 shows an actual output when solving a Mixed-Bag puzzle with two images. Note that that the puzzle of interest P_i is the glass and stone building while



(a) Ground-Truth Image (b) Type 2 Solver Output

(c) EDAS Visualization (d) SEDAS Visualization

Figure 2: Example Solver Output Visualizations for EDAS and SEDAS

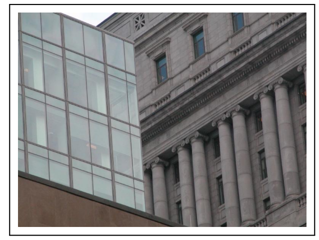
Wrong Puzzle	Wrong Neighbor	Correct Neighbor	No Piece Present

Table 2: Color Scheme for Puzzles Piece Sides in Neighbor Accuracy Visualizations

the other puzzle P_k is a rainforest house.

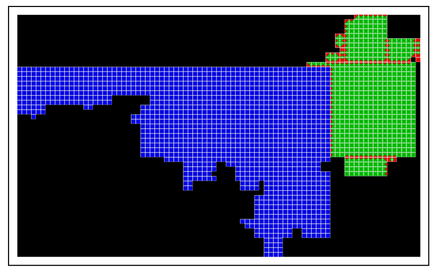
All pieces that came from the rainforest house image are shown as blue despite being assembled correctly; this is because they are not from the puzzle of interest. All neighbors from the puzzle of interest (i.e., the glass and stone building) that are placed next to their original neighbor are represented by green triangles while all incorrect neighbors, such as those bordering the rainforest house image, are shown

(a) Input Image # 1 - Rainforest House [20]



(b) Input Image # 2 - Building Exterior [21]

(c) Solver Output



(d) ENAS Visualization

Figure 3: Example Solver Output Visualization for ENAS

with red triangles.

LIST OF REFERENCES

[1] A. D. Williams, *Jigsaw Puzzles: An Illustrated History and Price Guide.* Wallace-Homestead Book Co., 1990.

[2] A. D. Williams, *The Jigsaw Puzzle: Piecing Together a History.* Berkley Books, 2004.

[3] H. Freeman and L. Gardner, ''Apictorial jigsaw puzzles: The computer solution of a problem in pattern recognition,'' *IEEE Transactions on Electronic Computers*, vol. 13, pp. 118--127, 1964.

[4] T. Altman, ''Solving the jigsaw puzzle problem in linear time,'' *Applied Artificial Intelligence*, vol. 3, no. 4, pp. 453--462, Jan. 1990.

[5] E. D. Demaine and M. L. Demaine, ''Jigsaw puzzles, edge matching, and polyomino packing: Connections and complexity,'' *Graphs and Combinatorics*, vol. 23 (Supplement), pp. 195--208, June 2007.

[6] B. J. Brown, C. Toler-Franklin, D. Nehab, M. Burns, D. Dobkin, A. Vlachopoulos, C. Doumas, S. Rusinkiewicz, and T. Weyrich, ''A system for high-volume acquisition and matching of fresco fragments: Reassembling Theran wall paintings,'' *ACM Transactions on Graphics*, vol. 27, no. 3, Aug. 2008.

[7] M. L. David Koller, ''Computer-aided reconstruction and new matches in the Forma Urbis Romae,'' *Bullettino Della Commissione Archeologica Comunale di Roma*, vol. 2, pp. 103--125, 2006.

[8] S. L. Garfinkel, ''Digital forensics research: The next 10 years,'' *Digital Investigation*, vol. 7, Aug. 2010.

[9] T. S. Cho, M. Butman, S. Avidan, and W. T. Freeman, ''The patch transform and its applications to image editing,'' *Proceedings of the 2008 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2008.

[10] L. Zhu, Z. Zhou, and D. Hu, ''Globally consistent reconstruction of ripped-up documents,'' *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 30, pp. 1--13, 2008.

[11] W. Marande and G. Burger, ''Mitochondrial DNA as a genomic jigsaw puzzle,'' *Science*, vol. 318, no. 5849, pp. 415--415, 2007.

[12] Y.-X. Zhao, M.-C. Su, Z.-L. Chou, and J. Lee, ‘‘A puzzle solver and its application in speech descrambling,’’ in *Proceedings of the 2007 International Conference on Computer Engineering and Applications*. World Scientific and Engineering Academy and Society, 2007, pp. 171--176.

[13] T. S. Cho, S. Avidan, and W. T. Freeman, ‘‘A probabilistic image jigsaw puzzle solver,’’ in *Proceedings of the 2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, ser. CVPR ’10. IEEE Computer Society, 2010, pp. 183--190.

[14] D. Pomeranz, M. Shemesh, and O. Ben-Shahar, ‘‘A fully automated greedy square jigsaw puzzle solver,’’ in *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, ser. CVPR ’11. IEEE Computer Society, 2011, pp. 9--16.

[15] G. Paikin and A. Tal, ‘‘Solving multiple square jigsaw puzzles with missing pieces,’’ in *Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, ser. CVPR ’15. IEEE Computer Society, 2015.

[16] A. C. Gallagher, ‘‘Jigsaw puzzles with pieces of unknown orientation,’’ in *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, ser. CVPR ’12. IEEE Computer Society, 2012, pp. 382--389.

[17] D. Sholomon, O. David, and N. S. Netanyahu, ‘‘A genetic algorithm-based solver for very large jigsaw puzzles,’’ in *Proceedings of the 2013 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, ser. CVPR ’13. IEEE Computer Society, 2013, pp. 1767--1774.

[18] D. Sholomon, O. David, and N. S. Netanyahu, ‘‘Datasets of larger images and GA-based solver’s results on these and other sets,’’ http://u.cs.biu.ac.il/~nathan/Jigsaw/, 2013, (Accessed on 05/01/2016).

[19] K. Son, J. Hays, and D. B. Cooper, ‘‘Solving square jigsaw puzzles with loop constraints,’’ in *Proceedings of the 2014 European Conference on Computer Vision (ECCV)*. Springer, 2014, pp. 32--46.

[20] D. Pomeranz, M. Shemesh, and O. Ben-Shahar, ‘‘Computational jigsaw puzzle solving,’’ https://www.cs.bgu.ac.il/~icvl/icvl_projects/automatic-jigsaw-puzzle-solving/, 2011, (Accessed on 05/01/2016).

[21] A. Olmos and F. A. A. Kingdom, ‘‘McGill calibrated colour image database,’’ http://tabby.vision.mcgill.ca/, 2005, (Accessed on 05/01/2016).