Jigsopu: Square Jigsaw Puzzle Solver with Pieces of Unknown Orientation

Abdullah M. Moussa
Electrical Engineering Department
Faculty of Engineering
Port-Said University, Port-Said, Egypt

***Abstract*—In this paper, we consider the square jigsaw puzzle problem in which one is required to reassemble the complete image from a number of unordered square puzzle pieces. Here we focus on the special case where both location and orientation of each piece are unknown. We propose a new automatic solver for such problem without assuming prior knowledge about the original image or its dimensions. We use an accelerated edge matching based greedy method with combined compatibility measures to provide fast performance while maintaining robust results. Complexity analysis and experimental results reveal that the new solver is fast and efficient.**

Keywords—jigsaw puzzle; image merging; edge matching; jigsaw puzzle assembly; automatic solver

I. Introduction

It is generally accepted that the first modern jigsaw puzzle was built in 1760 by London map maker John Spilsbury for educational purposes [1]. Since then, several different manufacturers around the world are manufacturing jigsaw puzzles in many shapes, sizes and piece types. The jigsaw puzzle is provably technically challenging. It has been shown by Demaine et al. [2] that the jigsaw puzzle problem is NP-complete in the general case when the pairwise affinity of jigsaw pieces is unreliable. The computational problem of jigsaw puzzle assembly was first introduced nearly fifty years ago in a fundamental work by Freeman and Gardner [3]. In addition to being an interesting problem in its own right, the computational jigsaw assembly has many applications in recovering shredded documents or photographs [4, 5, 6, 7], reassembling archaeological artifacts [8], DNA/RNA modeling [9] and speech descrambling [10].

Many attempts have been made to handle the problem. Several papers [11, 12] assume using classic jigsaw pieces with distinct shapes, and focus on matching the shape of the pieces to solve the puzzle. Some others use both of image contents and boundary shape [13, 14, 15]. In this paper, we follow the lead of recent work [16, 17, 18] and consider jigsaw puzzles with square pieces. We believe that assuming prior knowledge about the dimensions of the complete image of the puzzle pieces, as what the majority of the existing algorithms do, can reduce the applicability of the algorithm used. So, we relax the condition that the dimensions of the complete image should be previously known. We focus on the special case where both location and orientation of each piece are unknown. We present a new fast algorithm to tackle such problem. Our algorithm, named JigSoPU (Jigsaw Solver with Pieces of Unknown orientation), uses an edge matching based greedy technique along with a combined compatibility score functions to provide an accelerated performance while maintaining robust results. Complexity analysis and experimental results show that the new algorithm is fast and efficient.

The rest of the paper is organized as follows: Section II introduces the new jigsaw assembly algorithm. Section III presents the complexity analysis of the proposed algorithm. In Section IV the experimental results are presented. Finally, conclusions are summarized in Section V.

II. Algorithm Description

To solve the square jigsaw puzzle problem, we need to consider two aspects, a criterion of compatibility between jigsaw pair of pieces, and a specific strategy to assemble the pieces. In section II-A, we will present the compatibility measure used, and in section II-B the new edge matching based assembly strategy will be proposed.

A. Pairwise Compatibility Criterion

When the jigsaw puzzle is correctly assembled, it can be observed that the adjoining pieces have often adjacent edges with pixels of similar intensity values. Such feature is the base of jigsaw edge matching based solvers. Common dissimilarity measures can be used to calculate the minimum difference between the pixels of pieces' edges in search of the best match between two candidate pieces. However, depending on a single dissimilarity measure may make the algorithm getting trapped in an incorrect assembly. So, we propose to use a combined measure that jointly minimizes the mean value of SAD (sum of absolute difference) of candidate adjacent edges along with the amount of pixel pairs that have a SAD value above a predefined threshold. We found experimentally that using the proposed combined compatibility functions is efficient enough to even handle color images after converting them to grayscale versions without a need to deal with each color channel alone.

B. Assembly Strategy

In this section, we introduce the new edge matching based greedy assembly algorithm. The algorithm is inspired by the FRoTeMa algorithm [19, 20] for template matching. Exhaustive matching of edges in search of the complete image should be robust, but it is not efficient in time because such procedure will have a complexity of $O(N^2)$ where N is the number of edges. This costly complexity makes the exhaustive matching procedure difficult to be applied to large jigsaw

puzzles. So, there is always a need for faster algorithms that can tackle the problem without sacrificing robustness.

In order to be able to provide a faster solver, we should reduce the search space of the problem. We did this based on the observation mentioned in the previous section that the adjoining pieces in a correctly solved jigsaw puzzle have often similar intensity values of the pixels in the adjacent edges. Based on this observation, the sums of pixel intensities of adjacent edges tend to be similar. The following steps are made to make use of the last concept:

1) Calculate sum of pixel intensities for each edge in all pieces. We will call such sum as the weight of the edge.

2) Store the calculated weights in a vector S_v and then sort the vector in ascending order.

3) Pick the edge associated with a random weight in S_v.

4) Assign the picked edge to e_{ct}, its weight index in S_v to i_{ct} and the piece associated with it to p_{ct}.

Using a predefined threshold t_c, the sorted vector S_v can be used to check the possible match between e_{ct} and a subset of the edges within the range of indices in S_v of: $i_{ct} \pm t_c$. Within the mentioned range, each edge is checked for a possible match with e_{ct} using the criterion described in section II.A, if one of the edges meets the requirements of the compatibility measure, the piece associated with such edge will be a *candidate piece* p_{cand}. To even increase the robustness of the process, not all candidate pieces are considered as correct pieces, Referring to Fig. 1, the piece in dark blue is an example of p_{ct} and the piece in light blue is an example of p_{cand}.

To ensure that p_{cand} is a correct match with p_{ct}, we check the compatibility of the pieces around p_{ct} and p_{cand} from the two sides that are perpendicular to e_{ct} (the pieces in gray in Fig. 1) using the same strategy. If this patch of pieces is compatible, i.e., each piece of them and its candidate adjoining ones meet the requirements of the compatibility measure, in this case p_{cand} will be considered as a correct match and will be assigned to p_{ct}. If such patch of pieces is not compatible or none of the edges in the range $i_{ct} \pm t_c$ meets the requirements of the compatibility measure, the algorithm will consider p_{ct} as a *boundary piece*, i.e., p_{ct} is considered to have no adjacent pieces from the side of e_{ct}.

In this case and based on the direction of the blue arrow, one of the two perpendicular edges to e_{ct} will be the new e_{ct}. The process is continued sequentially using the direction of the blue arrow until assembling the rest of the pieces or reaching the corner, i.e., when reaching a piece which is a boundary piece from two sides. If the algorithm reaches a corner, the process is continued sequentially from the other side of the first picked piece (the pink arrow) until assembling the rest of the pieces.

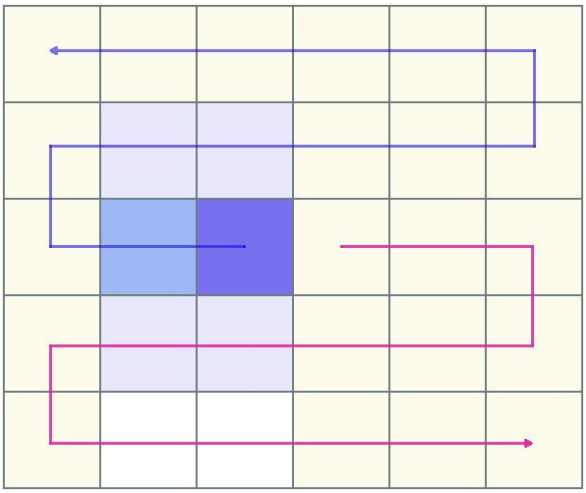

Fig. 1. Overview of JigSoPU assemply strategy

III. Complexity Analysis

Let N be the number of edges of all pieces, M the number of pixels in each edge and C the number of candidate adjacent edges checked for each edge. The operations used to calculate S_v have a complexity of $O(NM)$, while the operations used to sort S_v have a complexity of $O(N\ log\ N)$. Also, the algorithm uses $O(NMC)$ computations to assemble pieces. Thus, the overall complexity of the algorithm is $O(N\ (M\ C + log\ N))$. Such complexity makes a big difference with respect to speed of computation in comparison with the quadratic complexity we get when we match edges exhaustively.

IV. Experimental Evaluation

To check the performance of the new algorithm, we have applied the algorithm on 10 images from ETHZ dataset [21]. All experiments have been run on a Core i-5 (2.3-GHz PC) with 4 GB of RAM.

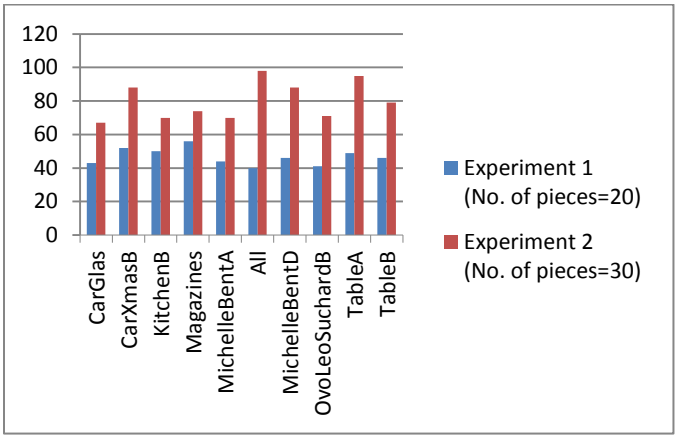

Fig. 2. Execution time of JigSoPU in milliseconds for each experiment

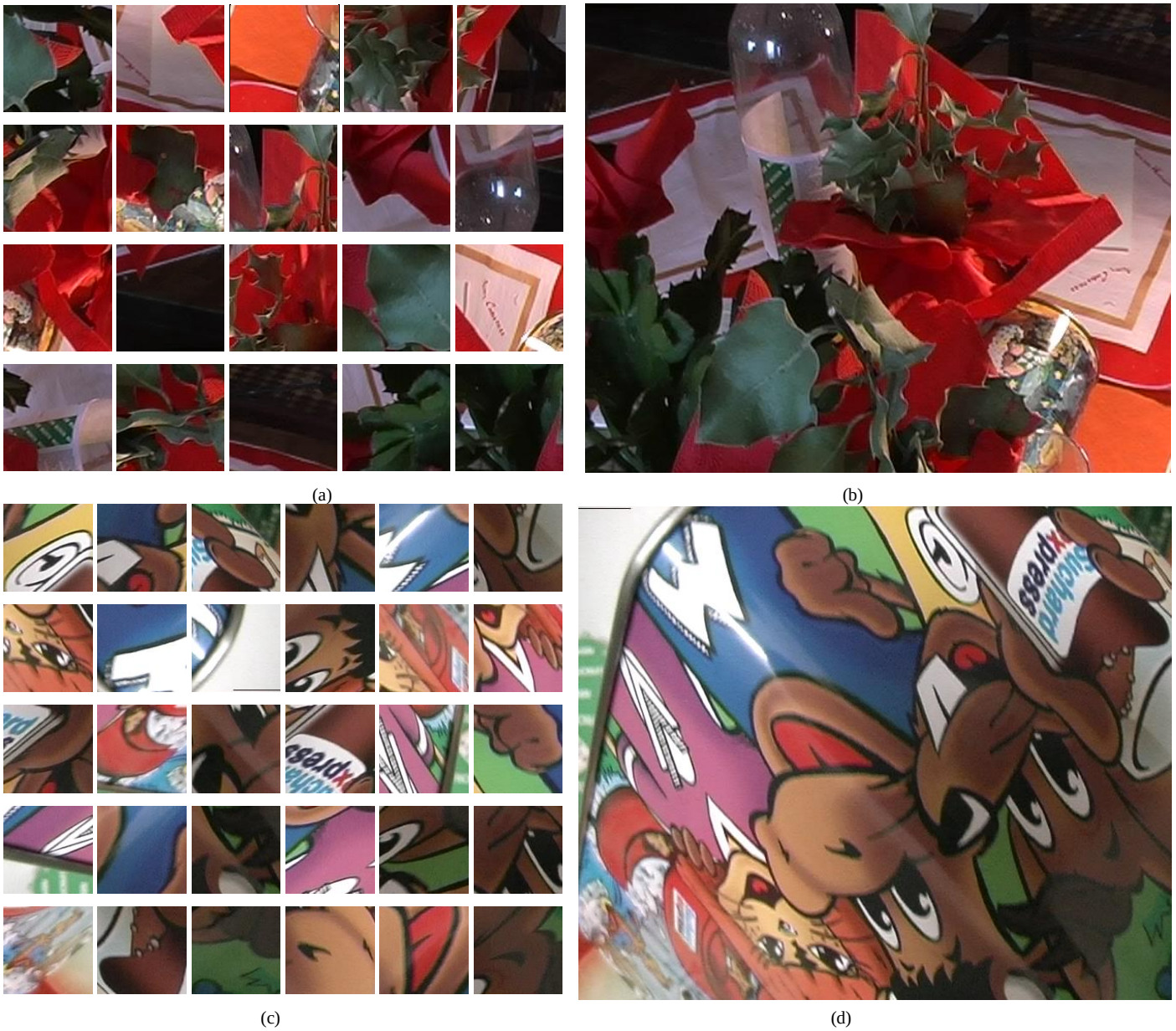

Fig. 3. Some examples of JigSoPU output. In (a) and (c), jigsaw puzzles of 20 and 30 pieces respectively. Perfect assembly is achieved in (b) and (d)

The algorithm has been tested twice for each image, in the first experiment, each image is split into 20 unordered randomly oriented square pieces, while in the second experiment, each image is split into 30 pieces in a similar manner. The proposed algorithm exhibited 100% accuracy for all of the tested images.

We have calculated the execution time for each experiment. Fig. 2 summarizes the performance of the proposed algorithm for each tested image. As shown in Fig. 2, the proposed algorithm provides fast computation as we can expect regarding its computational complexity. Some examples of JigSoPU performance are illustrated in Fig. 3. As we can see in Fig. 3, JigSoPU can assemble the complete images perfectly.

V. CONCLUSION

Computational jigsaw assembly is an interesting problem, which plays a vital role in many different scientific fields, such as image processing, computer vision, archeology and genomics. In this paper, we have presented a new fast algorithm to handle the problem of assembling square jigsaw puzzles where both location and orientation of each piece are unknown and without assuming prior knowledge about the complete image or its dimensions. The proposed algorithm has been tested against different images using different number of pieces. Complexity analysis and experimental results reveal that the new algorithm can provide fast computation and robust results.

Future work includes verifying the performance of the new algorithm against larger images and checking the effect of increasing the number of puzzle pieces on the performance. Also, it will be interesting to evaluate the applicability of the method to archaeological and genomic datasets.

REFERENCES

[1] R. Tybon. Generating Solutions to the Jigsaw Puzzle Problem. PhD thesis, Griffith University, Australia, 2004.

[2] E. D. Demaine and M. L. Demaine. "Jigsaw puzzles, edge matching, and polyomino packing: Connections and complexity," Graphs and Combinatorics, 23, 2007.

[3] H. Freeman and L. Gardner. "Apictorial jigsaw puzzles: The computer solution of a problem in pattern recognition," IEEE. Trans. on Electronic Computers,1964.

[4] S. Cao, H. Liu, and S. Yan. "Automated assembly of shredded pieces from multiple photos," In Proc. ICME, 2010.

[5] E. Justino, L. Oliveria, and C. Freitas. "Reconstructing shredded documents through feature matching," Forensic Science International, 2006.

[6] M. Marques and C. Freitas. "Reconstructing strip-shredded documents using color as feature matching," Proc. ACM Symposium on Applied Computing, 2009.

[7] L. Zhu, Z. Zhou, and D. Hu. "Globally consistent reconstruction of ripped-up documents," IEEE TPAMI, 2008.

[8] B. J. Brown, C. Toler-Franklin, D. Nehab, M. Burns, D. Dobkin, A. Vlachopoulos, C. Doumas, and T. W. Szymon Rusinkiewicz. "A system for high-volume acquisition and matching of fresco fragments: Reassembling theran wall paintings," ACM TOG (SIGGRAPH), 2008.

[9] W. Marande and G. Burger. "Mitochondrial DNA as a genomic jigsaw puzzle," Science, 2007.

[10] Y.-X. Zhao, M.-C. Su, Z.-L. Chou, and J. Lee. "A puzzle solver and its application in speech descrambling," In ICCEA,2007.

[11] D. Goldberg, C. Malon, and M. Bern. "A global approach to automatic solution of jigsaw puzzles," In Symposium on Computational Geometry, 2002.

[12] W. Kong and B. B. Kimia. "On solving 2D and 3D puzzles using curve matching," In IEEE CVPR, 2001.

[13] F.-H. Yao and G.-F. Shao. "A shape and image merging technique to solve jigsaw puzzles," PRL, 2003.

[14] T. R. Nielsen, P. Drewsen, and K. Hansen. "Solving jigsaw puzzles using image features," PRL, 2008.

[15] M. Makridis and N. Papamarkos. "A new technique for solving a jigsaw puzzle," In IEEE ICIP, 2006.

[16] N. Alajlan. "Solving square jigsaw puzzles using dynamic programming and the hungarian procedure," Amer. Journ. Applied Sciences, 2009.

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

[18] A.C., Gallagher, "Jigsaw puzzles with pieces of unknown orientation," in: 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), IEEE Comp. Soc. Press, Los Alamitos, CA, 2012, pp. 382–389.

[19] A. M. Moussa, M.I. Habib, and R. Y. Rizk. "FRoTeMa: Fast and Robust Template Matching," , unpublished.

[20] A. M. Moussa, Robust Template Matching with Rotation, Scale, Brightness and Contrast Invariance. Master's thesis, Port-Said University, Egypt, 2015.

[21] V. Ferrari, T. Tuytelaars, and L. V. Gool, "Simultaneous object recognition and segmentation from single or multiple model views," Int. J. Computer Vision, 2006.