GENETIC ALGORITHM OF JIGSAW PUZZLES

Zhuo Wang
Shanghai Jiao Tong University

1. INTRODUCTION

Jigsaw puzzles are a popular form of entertainment, first produced around 1760 by John Spilsbury, a Londonian engraver and mapmaker. Given n different non-overlapping pieces of an image, the player has to reconstruct the original image, taking advantage of both the shape and chromatic information of each piece. Despite the popularity and vast distribution of jigsaw puzzles, their assembly is not trivial, from a computational standpoint, as this problem was proven to be NP-hard (Altman 1989; Demaine and Demaine 2007).

I would like to try the problem about the square puzzles, where individual works lack features. Curve shape because all parts have straight edges and even rectangular size. In this case, the image information only provides us with information and Guidance provided there. This type of puzzle is sometimes called edge matching.

2. BACKGROUND AND RELATED WORK

Jigsaw puzzles were first produced around 1760 by John Spilsbury, a Londonian engraver and mapmaker. Nevertheless, the first attempt by the scientific community to computationally solve the problem is attributed to Freeman and Garder [8] who in 1964 introduced a solver which could handle up to nine-piece problems. Ever since then, the research focus regarding the problem has shifted from shape-based to merely color-based solvers of square-tile puzzles. Recent years we have witnessed a vast improvement in the research and development of automatic jigsaw puzzle solvers, manifested in both puzzle size, solution accuracy, and amount of manual human intervention required. Pomeranz et al. [17] introduced, for the first time, a fully automated square jigsaw puzzle solver that could handle puzzles of up to 3,000 pieces. Reported state-of-the-art solvers are fully automated and can handle puzzles of up to 9000 pieces.

3. MOTIVATION

Based on the knowledge I have learned, the square jigsaw puzzle refers to puzzles having square pieces featuring straight borders and uniform size which conform a grid that completes the solution image, as they lack of a curvilinear shape, it is possible that this kind of jigsaw puzzles have multiple solutions to the same picture.

Besides, solutions to this problem might benet the elds of biology, chemistry, literature, speech descrambling, archeology, image editing and the recovery of shredded documents or photographs.

4. PROBLEM FORMULATION

Given a image of N x M pieces, which represent a chromosome by an N x M matrix, each entry of which corresponds to a piece number, a piece is assigned a number according to its initial location in the given puzzle). Thus, each chromosome represents a complete solution to the jigsaw puzzle problem.

The N x M matrix for each individual and each cell c_i, j corresponds to one piece of puzzle. The matrix depicts the relevant arrangement of puzzle pieces in a very straightforward manner that the left/right/up/down adjacency in the matrix means the same interrelationship between the relevant puzzle pieces.

The assembly of the puzzle is performed by using the pixel values on the border lines of the individual pieces.

5. PROPOSED METHOD

5.1 The Genetic Algorithm

First, the initial population of candidate solutions, also known as chromosomes, is randomly generated. Each chromosome is a complete solution to the problem, such as the suggested arrangement of puzzle pieces. Next, various biologically inspired operators, such as selection, reproduction and mutation, in order to mimic natural selection, the rate of chromosome reproduction, ie the number of times it is selected for reproduction and hence the number of its offspring, is directly proportional to it. . Fitness. Fitness is the score obtained by the fitness function and it represents the quality of a given solution. Therefore, a "good" solution will have more descendants than other solutions. Chromosomes are more likely to multiply with other good chromosomes. A recurring operator called a cross should allow better features from the parent to be passed and combined into the sub-solution, potentially creating an improved solution. The success of the GA depends primarily on the selection of the appropriate chromosome representation, cross-counting Sub and fitness functions. The fitness function must correctly detect the chromosomes that contain the promising solution part for delivery to the next generation.

The basic GA framework for solving puzzle game problems is given by the pseudo code of Algorithm 1.

Algorithm 1 Pseudocode of GA framework

```
1:  population ← generate 1000 random chromosomes
2:  for generation_number = 1 → 100 do
3:      evaluate all chromosomes using the fitness function
4:      new_population ← NULL
5:      copy 4 best chromosomes to new_population
6:      while size(new_population) ≤ 1000 do
7:          parent1 ← select chromosome
8:          parent2 ← select chromosome
9:          child ← crossover(parent1, parent2)
10:         add child to new_population
11:     end while
12:     population ← new_population
13: end for
```

5.2 Fitness

The fitness function is evaluated for all chromosomes for the purpose of selection. Each piece of the puzzle is represented by matrix LxLx3, where L is the width and length of the piece (i.e. the number pixels along the border). The fitness function of our algorithm is based on two measures of dissimilarity. Horizontal dissimilarity between two pieces x_i and x_j, where x_j is placed to the right of x_i

is defined as:

$$D(p_i, p_j) = \sqrt{\sum_{k=1}^{W}\sum_{b=1}^{3}(p_i(k, W, b) - p_j(k, 1, b))^2}.$$

The fitness function of a given chromosome is the sum of pairwise dissimilarities over all neighboring pieces (whose configuration is represented by the chromosome). Representing a chromosome by an (N x M) matrix, where a matrix entry $x_{i,j}$ (i = 1..N, j = 1..M) corresponds to a single puzzle piece, we define its fitness as:

$$\sum_{i=1}^{N}\sum_{j=1}^{M-1}(D(x_{i,j}, x_{i,j+1}, r)) + \sum_{i=1}^{N-1}\sum_{j=1}^{M}(D(x_{i,j}, x_{i+1,j}, d))$$

5.3 Crossover

Given two parent chromosomes, two complete (different) permutations of all puzzles, the crossover operator constructs the subchromosomes in the form of kernel growth, using both parents as "consultants." The operator starts with a single piece and gradually joins the other pieces at the available boundaries. New works may only be adjacent to existing works, so new images are always continuous. The operator continually adds pieces from a set of available pieces until there are no more pieces left. Therefore, each piece appears only once in the resulting image. Since the image size is known in advance, the operator can ensure that there are no border violations. Therefore, by using each component accurately within the frame of the correct size, the operator is guaranteed to obtain an effective image.

Algorithm 2 Crossover operator simplified

1: If any available boundary meets the criterion of Phase 1 (both parents agree on a piece), place the piece there and goto (1); otherwise continue.
2: If any available boundary meets the criterion of Phase 2 (one parent contains a best-buddy piece), place the piece there and goto (1); otherwise continue.
3: Randomly choose a boundary, place the most compatible available piece there and goto (1).

A key feature of kernel growth techniques is that the final absolute position of each segment is determined only when the kernel reaches its final size and the sub-chromosomes are intact. Prior to this, depending on the kernel's growth vector, all parts may be moved. For example, the first block may end up in the lower left corner of the image, and if the kernel only grows up and to the right after the block is allocated. Instead, the same first piece may end up in the center of the image, in the upper right corner or at any other location.

It is still a question of which one to choose from the available fragment libraries and where to find it among children. Given a kernel, a partial image, we can mark all the boundaries that might place a new segment. The segment boundary is represented by a pair (xi, R) and consists of a segment number and a spatial relationship. The operator invokes a three-stage procedure. First, given all existing boundaries, the operator checks to see if there are two parental consent fragments of the segment xj (meaning that both contain the segment in the spatial direction R of xi). If there is such a part, place it in the correct position. If the parent agrees to two or more boundaries, randomly select one of the boundaries and assign the corresponding portion. Obviously, works that have already been used cannot be redistributed, so any such work will be ignored, as if the parents disagreed with that particular boundary. If there is no agreement between parents on any of the boundaries, the second phase begins. If multiple best-buddy pieces are available, one is chosen at random. If a best-buddy piece is found but was already assigned, it is ignored and the search continues for other best-buddy pieces. Finally, if no best-buddy piece exists, the operator randomly selects a boundary and assigns it the most compatible piece available.

5.4 Rationale

In a GA framework, good traits should be passed on to the child. Since position independence of pieces is encouraged, the trait of interest is captured by a pieces set of neighbors. Correct puzzle segments correspond to a correct placement of pieces next to each other. The notion that piece xi is in spatial relation R to piece x_j is key to solving the jigsaw problem. Nevertheless, every chromosome accounts for a complete placement of all the pieces. Taking into account the random nature of the first generation, the procedure must actively seek the better traits of piece relations.

Therefore, we assume that a trait common to both parents has propagated through the generations and comprises the reason for their survival and selection. In other words, if both parents agree on a relation, we regard it as true with high probability. Note that not all agreed relations are copied immediately to the child. Since a kernel-growing algorithm is used, some agreed pieces might prematurely serve as most compatible pieces at another boundary and be subsequently disqualified for later use. Thus, random agreements in early generations are likely to be nullified. As for the second stage, where the parents agree on no piece, one might be inclined to randomly pick a parent and follow its lead. Another option might be to just choose the most compatible piece in a greedy manner, or check if a best-buddy piece is available. Since piece placement in parents might be random and since even best-buddy pieces might not capture the correct match, we combine the two. The fact that two pieces are both best-buddies and are adjacent at a parent is a good indication for the validity of this match. A different perspective is to consider that every chromosome contains some correct segments. The passing of correct segments from parents to children is at the heart of the GA. Moreover, if each parent contains a correct segment and these segments partially overlap, the overlapping (agreed upon) part will be copied to the child in the first phase and be completed from both parents at the second stage, thus combining the segments into a larger correct segment, creating a better child solution, and advancing the pursuit of the entire correct image.

6. EXPERIMENT

I have experimented with five different pictures that were divided into N x M pieces. So the size of the picture changed accordingly the number of pieces taken into account. I use the dictionary to store the original image of the segmentation, remember the original dictionary dictionary order, and then compare the moved dictionary order with the original image. I test some images with GENERATIONS = 30 and POPULATION = 200. The results are shown in

the table below.

No. of pieces	similarity
103	94.25%
300	95.30%
420	94.06%
532	93.66%
628	94.72%

I have also faced some problems of dealing with the images. For example, the storage order of OpenCV for RGB image data is BGR, thus, make BGR2GRB. Then, use the SizeDetector function to detects piece size in pixels from given image Image is split into RGB single-channel images. Single-channel images are combined (R + G, R + B, G + B) in order to cover special edge cases where input image have one dominant color commponent. For each single channel-image size candidates are found and candidate with most occurances is selected.

The results is Ok for pictures which is no more than around 800x800 so far. The generationThere are still some opportunities to improve the way of points of fraction location as well as the way of matching the individual segments.

7. CONCLUSION

My algorithm is capable of solving puzzles consisting of several hundred pieces. The GA progresses very fast with better solutions coming in every few iterations, but this tends to saturate in the later stages where the improvements are very small.

8. REFERENCE

[1] Richter, F., Ries, C. X., Cebron, N., Lienhart, R.: Learning to Reassemble Shredded Documents. IEEE Transactions on Multimedia, vol. 15, no. 3, pp. 582 593 (2013).
[2] Demaine, E.D., Demaine, M.L.: Jigsaw Puzzles, Edge Matching, and Polyomino Packing: Connections and Complexity. Graphs and Combinatorics 23, 195208 (2007)
[3] D. Pomeranz, M. Shemesh, and O. Ben-Shahar. A fully automated greedy square jigsaw puzzle solver. In IEEE Conference on Computer Vision and Pattern Recognition, pages 916, 2011.
[4] Gindre, F., Trejo Pizzo, D.A., Barrera, G., Lopez De Luise, M.D.: A Criterion-based Genetic Algorithm Solution to the Jigsaw Puzzle NP-Complete Problem. Lecture Notes in Engineering and Computer Science, vol. 2186 (1) (2010)
[5] Murakami, T., Toyama, F., Shoji, K., Miyamichi, J.: Assembly of puzzles by connecting between blocks. In Proc. of 19th International Conference on Pattern Recognition, Tampa, FL, pp. 1-4, (2008).
[6] T. Cho, S. Avidan, W. Freeman, A probabilistic image jigsaw puzzle solver, in IEEE Conference onComputer Vision and Pattern Recognition (2010), pp. 183190
[7] A. Deever, A. Gallagher, Semi-automatic assembly of real cross-cut shredded documents, in IEEEInternational Conference on Image Processing (2012), pp. 233236
[8]K. Son, J. Hays, D.B. Cooper, Solving square jigsaw puzzles with loop constraints, in EuropeanConference on Computer Vision 2014 (Springer, 2014), LNCS 8694, pp. 3246
[9] D. Sholomon, O. David, and N. Netanyahu. Datasets of larger images and GA-based solvers results on these and other sets. http://www.cs.biu.ac.il/nathan/Jigsaw.