A Puzzle Solver and Its Application in Speech Descrambling

YU-XIANG ZHAO, MU-CHUN SU, ZHONG-LIE CHOU, AND JONATHAN LEE
Department of Computer Science & Information Engineering
National Central University
TAIWAN, R.O.C.
http://cilab.csie.ncu.edu.tw/

Abstract: - The security problem of speech communication has always been a demanding problem in military and business areas. A common approach to realizing end-to-end security is the use of a scrambler. Most of the scramblers are based on permutation of speech signals in the time domain and/or frequency domain. On the other hand, descramblers are used to eavesdrop information from scrambled speech signals. In this paper we propose a new approach to implement a descrambler. We treat the descrambling problem as a puzzle solving problem. In this considered puzzle problem, each piece is a rectangular-shaped gray scaled image puzzle. We propose two different methods to assemble puzzles. While the first method is based on human heuristics, the second method is based on the Ant Colony System (ACS) algorithm. Sixty scambled images were used to test the proposed methods.

Key-Words: - puzzle, speech scrambling, optimization algorithm, ant colony system, swarm intelligence

1 Introduction

Secure speech communications have been a major concern in the military and business areas. The techniques for achieving end-to-end security of speech communications have been evolved over many years [1]-[11]. These techniques can be dichotomized into two categories: digital speech encryptors and analog speech scramblers. Digital speech encryptors can offer high grade security via the use of encryption; however, they usually require high bit rates to maintain speech quality. Therefore, analog speech scramblers, which can provide considerable grade security with acceptable speech quality, are still widely adopted in many applications.

The aim of an analog scrambler is to corrupt a speech signal as much as possible to prevent eavesdroppers from tapping information from communication channels, but the scrambled speech signal can still be recovered to an intelligible speech signal at the destination. Speech signals can be scrambled in either the time domain or the frequency domain. There are two main ways to scramble speech signals in the time domain [3]. They are time element reversal and time element permutation. As for the frequency domain scramblers, there are also two main approaches: inverters and band scramblers.

To increase security, a hybrid scrambler which scrambles speech signals in both time and frequency domain is usually adopted. A practical example of a speech signal scrambled in both time and frequency is shown in Fig. 1. The original speech signal in time domain is shown in Fig. 1(a) and its corresponding spectrogram is shown in Fig. 1(c). We scramble the speech signal in both time and frequency domain. The scrambled spectrogram is shown in Fig. 1(d) and its corresponding speech signal in time domain is shown in Fig. 1(b). If we directly output the scrambled signal via a speaker we won't be able to tell what message was conveyed by this voice signal. For a scrambled signal with M frequency bands and N time segments, if there is no information about how the signal was scrambled we are unable to recover the signal from the scrambled signal since there are $(M \times N)!$ permutations. Therefore, it is a challenge to descramble scrambled signals.

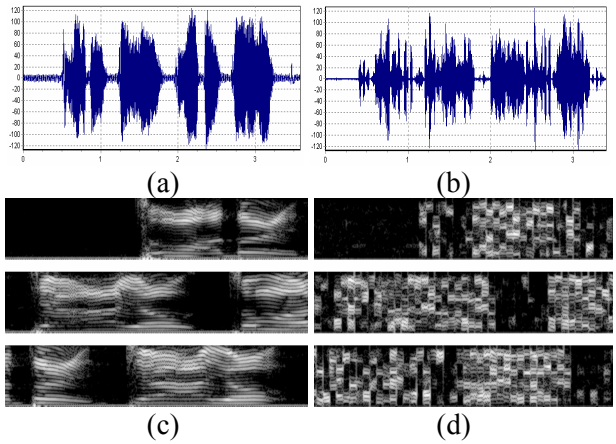


Fig. 1. The idea of scrambling a speech signal: (a) the original speech signal in time domain, (b) the scrambled speech signal in time domain. (c) the original spectrogram, and (d) the scrambled spectrogram.

In this paper, we propose an interesting approach

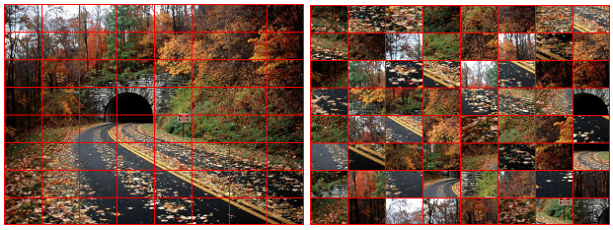
to descramble scrambled signals. We regard the problem of descrambling scrambled signals as a puzzle solving problem. For this special puzzle problem, each puzzle piece is a rectangular-shaped gray scaled image. We propose two different methods to assemble puzzles. While the first method is based on human heuristics, the second method is based on the Ant Colony System (ACS) algorithm [12]-[14]. The remaining of this paper is as follows. Section 2 gives a brief review of puzzle problems. Section 3 introduces the algorithm for solving the puzzle problem. The simulation results are presented in section 4. Finally, section 5 concludes the paper.

2 Background

In this section we will briefly review the puzzle problem. Puzzle problems are popular benchmarks for the evaluation of techniques such as computer vision, artificial intelligence, etc. many studies have been done to solve many different types of puzzle problems. In [15], two combinatorial puzzles, the *n*-Queen problem and polynomino puzzle problem, were solved by using a neural-network-based energy minimization method. While the goal of the *n*-Queen problem is to place n queens on an $n \times n$ chessboard such that they cannot capture one another, the goal of the polynomial puzzles problem is to fill variously shaped polynominoes exactly in a rectangle board. Yamamoto et al. proposed a neuro-based optimization algorithm for the 3-D rectangular puzzle problem whose goal is to arrange the irregular-shaped blocks to perfectly fit into a fixed 3-D rectangular shape [16]. Two new approaches based on the Hopfield network were proposed to solve another kind of puzzle problem. In this of puzzle problem, there is a 5×5 table, numbered 1 to 24 together with an empty box, used for moving the numbers [17]. The goal of this puzzle problem is to use the empty box to move the numbers around and finally end with the numbers sorted in ascending order.

In addition to the aforementioned puzzle problems, the jigsaw puzzle problem is another well known benchmark since it can be applied to diverse practical problems such as restoration of archaeological findings, repair of broken objects, molecular docking problem for drug design, etc [18]. Many different approaches have been proposed to solve the jigsaw puzzle problems [18]-[29]. A jigsaw puzzle solving system usually involves many techniques such as computer vision, partial boundary matching, pattern recognition, and combinatorial optimization. While some approaches [19]-[24] principally utilize the geometric shape information of the puzzle pieces, the jigsaw puzzle solver proposed in [18] uses chromatic information and partial boundary information. The assembly algorithms employed at the final stage can be roughly divided into three approaches. One approach purely depends on the trial-and-error scheme to assemble puzzle on the piece-by-piece basis [26]. Some approaches utilize heuristics to reduce the search space and then regard the assembly problem as the traveling salesman problem [18], [25].

In this paper, we consider a special puzzle problem where the shape of each puzzle piece is a rectangle. To our best knowledge, only one article addressed such a puzzle problem [30]. Toyama et al. proposed a GA-based approach to solve the rectangle piece puzzle assembly problem. In the problem considered by them, the shape of each puzzle piece is a rectangle and a picture of each puzzle is a black-and white image. In addition, they assume that each puzzle piece does not rotate. Fig. 2 illustrates such a puzzle problem.



(a) (b)

Fig. 2. An example of a rectangular puzzle problem: (a) the original image, and (b) the scrambled puzzle with 8 x 8 pieces.

3 The Assembly Method

For a puzzle problem with $M \times N$ rectangular-shape gray scaled image pieces, there are $(M \times N)!$ permutations. Therefore, it is a challenge to solve puzzle problems when the number of puzzle pieces is large.

We propose two different methods to solve this kind of puzzle problems. The first method referred to as the heuristic method which is based on common human heuristics in puzzle assembly. The second method is to apply the ACS algorithm to assemble puzzles.

3.1 The Heuristic Method

For jigsaw puzzles, puzzle pieces can be divided into three classes: (1) corner pieces where each piece is with two straight edges, (2) one-edge pieces where each piece is with one straight edge, and (3) internal pieces where each piece is without straight edge [24]. One may first assembly outer frame pieces which are consisted of corner pieces and one-edge pieces. Then a greedy algorithm can be used to assemble the interior [22]. Unfortunately, puzzle pieces in our

puzzle problems are all of the same shape. Therefore, straight line sides can't be used to decide which pieces are corner pieces since each piece has four straight lines.

In this paper, we first propose to adopt the heuristic method to assemble puzzle pieces. The goal of the assembly procedure is to place puzzle pieces in the scrambled map to their correct locations in the destination puzzle map as shown in Fig. 3. Before we present the heuristic method, we have to introduce the definitions of the so-called "vertical distance" and "horizontal distance". Similar to the distance definition proposed in [30], the distance between two pieces is defined as the root mean square error between every adjacent pixel pair on the touching border line of two pieces. Then the vertical (horizontal) distance is the distance between two vertically (horizontally) connected pieces, as shown in Fig. 4.

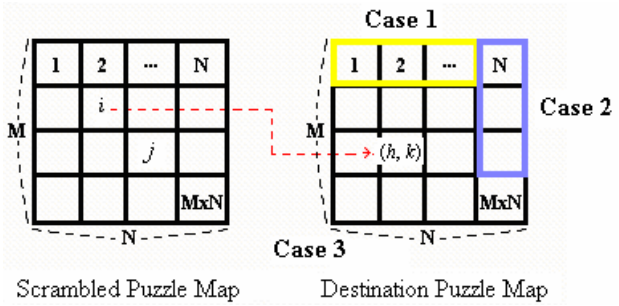


Fig. 3. The three cases when the *i*th puzzle piece in the scrambled puzzle map is assigned to the location (*h*, *k*) of the destination puzzle map.

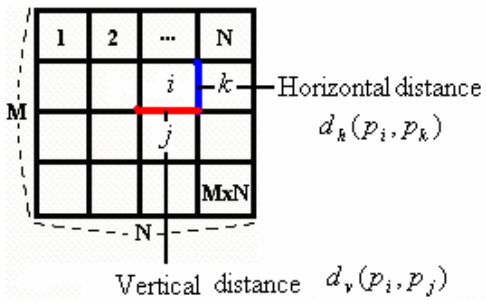


Fig. 4. The distance definition.

The first thing we need to decide is which piece is the upper left corner piece. After the upper left corner piece has been found, the piece which has the smallest "horizontal distance" to the upper left corner piece is chosen to be the second piece on the first row of the puzzle. The remaining pieces on the first row can be chosen according to the same rule. After the first row has been decided, the piece which has the smallest "vertical distance" to the upper left corner piece is chosen to be the first piece on the second row of the puzzle. According to the same rules all the remaining pieces can be found piece by piece until all pieces are placed to their correct positions.

Now we return to the method of finding the upper left corner piece. The simplest way is to try each piece as a tentative upper left corner piece and then proceed to assemble the remaining pieces. Finally, the best resultant puzzle from the $M \times N$ resultant puzzles is chosen to be the solution. The measure for evaluating the quality of the resultant puzzle will be given in Eq. (5).

From many simulation results, we found that the heuristic method could have 50% chance of successfully solving puzzle problems. We propose to adopt the ACS algorithm to further improve the performance of the heuristic method.

3.2 The ACS-Based Method

Recently, the social insect metaphor for solving problems has attracted a lot of attention from many different fields [31]-[32]. The ant colony optimization approach initiated by Dorigo [12]-[14], in collaboration with Colorni and Dorigo [33], has been receiving increasing amounts of attention due to its simplicity. The basic idea of the ACS algorithm is to use a positive feedback mechanism based on an analogy with the trail-laying and trail-following behavior of ants to reinforce those portions of good solutions that contribute to the quality of these solutions [31].

Instead of directly apply the original ACS algorithm to solve the puzzle problem we have to make several modifications in order to make the ACS algorithm to fit our problem. The probability for ant *k* to go from the *i*th puzzle piece to the j*i*th puzzle piece while building its *t*th tour is according to the following transition rule:

$$P_{ij}^{k}(t) = \frac{[\tau_{ij}(t)] \cdot [\eta_{ij}]^{\beta}}{\sum_{l \in J_i^k} [\tau_{ij}(t)] \cdot [\eta_{ij}]^{\beta}} \qquad (1)$$

where β is an adjustable parameter which controls the relative weight of the visibility, η_{ij}. The set J_i^k is consisted of the pieces which have to be visited by ant *k* when the ant is on piece *i*. The visibility , η_{ij} , is the so-called visibility which is based on strictly local information and represents the heuristic desirability of choosing puzzle piece *j* when in puzzle piece *i*. The trail intensity, τ_{ij} , is the virtual pheromone trail which represents the learned

desirability on the edge connects the *i*th puzzle piece to the *j*th puzzle piece. Note that at the first iteration, the piece *j* with the largest value of P_{ij}^k is selected.

Assume that piece *i* has been assigned to the location (*h*, *k*) of the destination puzzle map. The value of the visibility, η_{ij}, is defined according to which one of the following three cases is met (as shown in Fig. 4).

Case 1: If $h = 1$ and $1 \le k \le N-1$ then $\eta_{ij} = 1/d_h(p_i, p_j)$.

Case 2: If $k = N$ and $1 \le h \le M-1$ then $\eta_{ij} = 1/d_v(p^d_{(h,1)}, p_j)$. The puzzle piece, $p^d_{(h,1)}$, represents the puzzle piece which has been assigned to the location (*h*,1) of the destination puzzle map.

Case 3: For the other situations, $\eta_{ij} = 1/[d_h(p_i, p_j) + d_v(p^d_{(h-1,k+1)}, p_j)]$. The puzzle piece, $p^d_{(h-1,k+1)}$, represents the puzzle piece which has been assigned to the location (*h*-1,*k*+1) of the destination puzzle map.

The tour length visited by each ant is computed as follows:

$$L^k(t) = \sum_{i=1}^{M}\sum_{j=1}^{N-1} d_h(p^d_{(i,j)}, p^d_{(i,j+1)}) + \sum_{j=1}^{N}\sum_{i=1}^{M-1} d_v(p^d_{(i,j)}, p^d_{(i+1,j)}) \quad (2)$$

The tour length is a measure which reflects the degree of the completeness of the resultant puzzle map assembled by ant *k*.

The pheromone trail information is updated on-line during the problem solving procedure to reflect the experience acquired by ants. After the completion of an assignment, the ant that generated the best tour is allowed to globally update the concentrations of pheromone on the edges belonging to the best tour as follows:

$$\tau_{ij}(t) \leftarrow (1-\rho)\tau_{ij}(t) + \Delta\tau_{ij}(t) \quad (3)$$

$$\Delta\tau_{ij}(t) = 1/L^+ \quad (4)$$

where L^+ is the length of the best tour. In addition to the global updates, local updates of pheromone trails will be performed to make other potential solution can emerge. Detailed descriptions about AS and ACS algorithm can be found in [31].

4 Simulation Results

Twenty images downloaded from the database in the Web site Caltech 101 [34] were used to evaluate the performance of the proposed puzzle-solving methods. Fig. 5 shows some examples of the images used in our experiments. Each image was divided into three different sizes, 4×4, 4×6, and 4×8. Then each divided image was randomly scrambled to consist of the data set to be solved by our methods.

Two measures were used to evaluate the performance of the proposed methods. The first measure was the ratio of the number of the correct resultant puzzle maps to the number of the total scrambled puzzle maps. The second measure reflects how much completeness has been improved. These two measures are defined as follows:

$$C_r = \frac{Number\ of\ the\ correct\ images}{Number\ of\ the\ total\ images} \quad (5)$$

$$C_c = \frac{C_s - C_r}{C_s - C_o} \times 100\% \quad (6)$$

where C_o, C_s, and C_r represent the completeness degree of the original image, the scrambled image, and the resultant image, respectively. The degree of completeness is computed by Eq. (2). Note that the completeness degree of the original image is not zero as we originally expect.

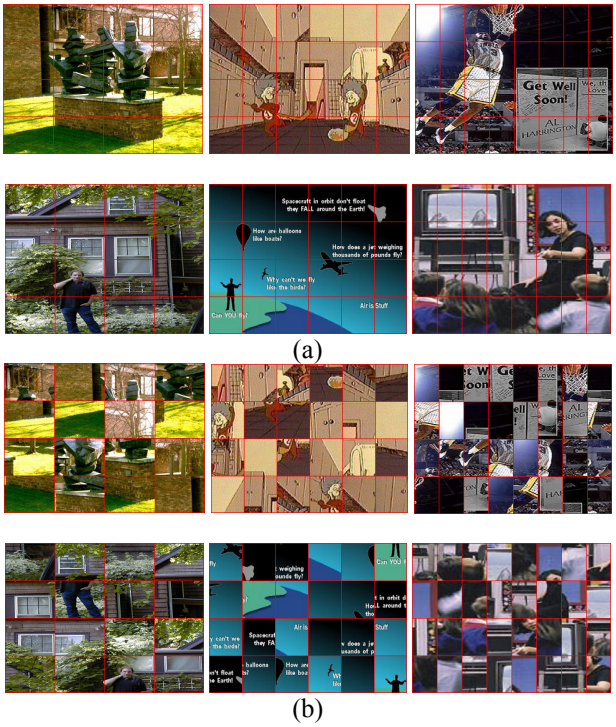
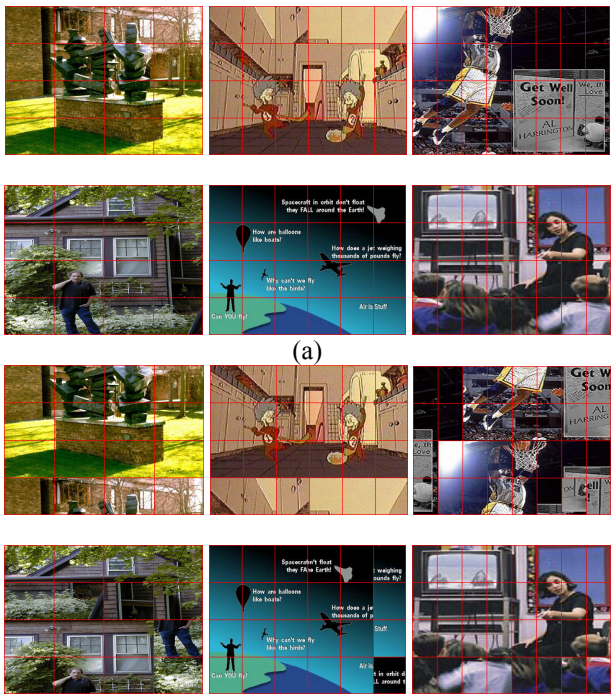


Fig. 5. Some testing images. (a) The original images. (b) The scrambled images.



(a)

(b)

Fig. 6. Some resultant images. (a) The unsuccessful images achieved by the heuristic method. (b) The successful results achieved by the ACS-based method.

Table 1. The performance achieved by the proposed methods.

Methods	Measure	4 x 4	4 x 6	4 x 8
Heuristic Method	C_r	50%	50%	50%
	C_c	93.08	91.08	93.96
ACS-based Method	C_r	75%	70%	60%
	C_c	98.67	97.84	97.25

Table 1 shows the performance achieved by the two methods. The ACS-based method outperformed the heuristic method based on the comparisons of the two measures. For the ACS-based method, the smaller the number of the puzzle pieces the better the performance could be achieved. Fig. 6 shows the images which could be successfully solved by the ACS-based method but not the heuristic method.

5 Conclusions

In this paper we treat the descrambling problem as a special puzzle solving problem. In this considered puzzle problem, each piece is a rectangular-shaped gray scaled image puzzle. Two different approaches to solving assemble the special puzzles were proposed. The first method is based on common human heuristics. It is very straightforward but effective for many puzzles problems. For some puzzle problems where the first approach doesn't work well, the second approach which is based on ACS algorithm can provide appealing solutions.

Acknowledgements

This work was partly supported by the NSC Program for Promoting Academic Excellent of Universities (Phase II) under the grant number NSC-95-2752-E-008-002-PAE, the National Science Council, Taiwan, R.O.C, under the NSC-95-2221-E-008-128 and the NSC-95-2524-S-008-001, and the NSC-95-2524-S-008-005-EC3, the Ministry of Economic Affairs under the 95-EC-17-A-02-S1-029 and the NCU Project of Promoting Academic Excellence & Developing World Class Research Centers-Applied Informatics and Creative Contents: Service-Oriented Information Platform.

References:

[1] N. S. Jayant, "Analog scramblers for speech privacy," *Computers and Security*, Nov.-Dec. 1982.

[2] H. J. Beker, and F. C. Piper, *Secure Speech Communications*, Academic, 1985.

[3] A. Matsunaga, K. Koga, and M. Ohkawa, "An Analog Speech Scrambling System Using the FFT Technique with High-Level Security," *IEEE Journal on Selected Areas in Communications*, vol. 7, no. 4, pp. 540-547, May 1989.

[4] E. V. Stansfield, D. Harmer, and M. F. Kerrigan, "Speech processing techniques for HF radio security," *IEE Proceedings on Communications, Speech and Vision*, vol. 136, no. 1, pp. 25-46, Feb. 1989.

[5] R. Yarlagadda, and K. Rao, *Hadamard matrix analysis and synthesis: with applications to communications and signal/image processing*, Kluwer Academic Publishers, 1997.

[6] V. Senk, V. D. Delic, and V. S. Milosevic, "A New Speech Scrambling Concept Based on Hadamard Matrices," *IEEE Signal Processing Letters*, vol. 4, no. 6, pp. 161-163, June 1997.

[7] A. Matsunaga, D. Koga, and M. Ohkawa, "An Analog Speech Scrambling System Using the FFT Technique with High-Level Security," *IEEE Journal on Selected Areas in Communications*, vol. 7, no. 4, pp. 540.-547, May 1989.

[8] B. Goldburg, S. Sridharan, and E. Dawson, "Design and Cryptanalysis of Transform-Based Analog Speech Scramblers," *IEEE Journal on*

Selected Areas in Communications, vol. 11, no. 5, pp. 735-744, June 1993.

[9] R. J. Sutton, *Secure Communications*, John Wiley & Sons, Inc., 2002.

[10] Y. Wu, and B. P. Ng, "Speech Scrambling with Hadamard Transform in Frequency Domain", *Proceedings 2002 IEEE 6th International Conference on Signal Processing*, vol. 2, pp. 1560- 1563, Aug. 2002.

[11] J. Ahmed, and N. Ikram, "Frequency-domain speech scrambling/descrambling techniques implementation and evaluation on DSP," *Proceedings IEEE INMIC'2003*, Dec. 2003.

[12] M. Dorigo, "Ottimizzazione, Apprendimento Automatico, ed Algoritmi Basati su Metafora Naturale." *Ph.D. Dissertation, Politecnico di Milano, Press*, 1992.

[13] M. Dorigo, and L. M. Gambardella,"Ant Conlony System: A Cooperative Learning Approach to the Traveling Salesman Problem," *IEEE Trans. Evol. Comp. 1*, pp. 53-56, 1997.

[14] M. Dorigo, and L. M. Gambardella, "Ant Colonies for the Traveling Salesman Problem," *BioSystems 43*, pp. 73-81, 1997.

[15] M. Kajiura, Y. Akiyama, and Y. Anzai, "Solving large scale puzzles with neural networks," *Proceedings IEEE TAI'89*, pp. 562-569, Oct. 1989.

[16] H. Yamamoto, H. Ninomiya, and H. Asai, "Application of Neuro-Based Optimization to 3-D Rectangular Puzzles," *Proceedings IEEE&INNS/IJCNN'98 (WCCI'98)*, May 1998.

[17] J. Taheri, "Hierarchical Hopfield neural network in solving the puzzle problem," *Proceedings 2004 IEEE International Joint Conference on Neural Networks*, vol. 3, pp. 2337-2342, July 2004.

[18] M. G. Chung, M. Fleck, and D.A. Forsyth, "Jigsaw Puzzle Solver Using Shape and Color," *Proceedings IEEE ICSP'98*, pp. 877–880, 1998.

[19] 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, April 1964.

[20] G. M. Radack, and N. I. Badler, "Jigsaw puzzle matching using a boundary-centered polar encoding," *Computer Graphics and Image Processing*, vol. 19, pp. 1-17, 1982.

[21] H. Wolfson, E. Schonberg, A. Kalvin, and Y. Lamdan, "Solving jigsaw puzzles by computer," *Annals of Operations Research*, vol. 12, pp. 51-64, 1988.

[22] T. Altman, "Solving the jigsaw puzzle problem in linear time," *Applied Artificial Intelligence*, vol. 3, pp. 453-462, 1989.

[23] R. W. Webster, P. S. LaFollette, and R. L. Stafford, "Isthmus critical points for solving jigsaw puzzles in computer vision," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 21, pp. 1271-1278, 1991.

[24] C. A. Rothwell, A. Zisserman, D. A. Forsyth, and J. L. Mundy, "Canonical frames for planar object recognition," *Proceedings of 2nd European Conference on Computer Vision*, pp. 757-772, 1992.

[25] G. C. Burdea, and H. J. Wolfson, "Solving jigsaw puzzles by a robot," *IEEE Transactions on Robotics and Automation*, vol. 5, no. 6, pp. 752-764, 1989.

[26] D. Kosiba, P. M. Devaux, S. Balasubramanian, T. Gandhi, and R. Kasturi, "An Automatic Jigsaw Puzzle Solver," *Proceedings of the 12th IAPR International Conference on Pattern Recognition*, IEEE Computer Society Press, Jerusalem, vol. 1, pp. 616-618. 1994.

[27] J. D. Bock, P. D. Smet, W. Philips, J. D'Haeyer, "Constructing the Topological Solution of Jigsaw Puzzles," *Proceedings ICIP'04*, pp. 2127-2130, 2004.

[28] D. Goldberg, C. Malon, and M. Bern, "A global approach to automatic solution of jigsaw puzzles," *Proceedings of the 18th Annual Symposium on Computational Geometry*, pp. 82-87, 2002.

[29] F. H. Yao, and G. F. Shao, "A shape and image merging technique to solve jigsaw puzzles," *Pattern Recognition Letters*, vol. 24, no. 12, pp. 1819-1835, 2003.

[30] F. Toyama, K. Shoji, and J. Miyamichi, "Assembly of Puzzles Using a Genetic Algorithm," *Proceedings ICPR'02*, vol.4, pp. 389-392, 2002.

[31] E. Bonabeau, M. Dorigo, and G. Theraulaz, *Swarm Intelligence: From Natural to Artificial Systems*, Oxford University Press, 1999.

[32] J. Kennedy, R. C. Eberhart, and Y. Shi, *Swarm Intelligence*, New York: Academic Press, 2001.

[33] A. Colorni, M. Dorigo, and V. Maniezzo, "Distributed Optimization by Ant Colonies," *In Processings First Europ. Conference on Artifical Life*, edited by F. Varela and P. Bourgine, Cambridge, MA: MIT Press, pp. 134-142. 1991.

[34] C. Chang, and C. Lin, *LIBSVM: a library for SVMs*, 2001. Available: http://www.vision.caltech.edu/Image_Datasets/Caltech101/Caltech101.html.