Indian Institute of Technology Bombay

IE-501, 2023

Report On

Project on Jigsaw Puzzles

Author:	Nisarg Jain - 23N0454
Author:	Neeraj Jadhav - 200050086
Author:	Aakash Verma - 23N0453
Author:	Onkar Borade - 200050022
Course Instructor:	Dr. Avinash Bhardwaj

Contents

List of Figures

1 Introduction

Jigsaw puzzles are a popular form of entertainment and a fascinating hobby. They consist of a picture or image that has been divided into numerous small, uniquely shaped pieces, and the objective is to reassemble these pieces to recreate the original image. In this project we consider square nxn images that are shuffled after dividing in blocks. Once shuffled our task remains to assemble the shuffled image from the image information only. Its an intriguing problem noted for its applications to real-world problems. For instance the automatic reconstruction of fragmented pieces is of great interest in archaeology or art restoration, where the artefacts recovered from sites are often fractured.

2 Literature Survey

- Initial solvers used only the shape information to solve the 9-piece puzzle and further work which used shape information and not the image itself.
- Jigsaw puzzles are NP-hard when multiple correct pieces are possible.
- Two main strategies:

  Greedy methods: Start from initial pairwise matches and build for larger components.

  Global methods: Directly search for a solution to maximize a global compatibility function.
- Son et al. found that puzzle cycles to exploit constraints lead to a better performance than directly solving the global problem

3 Matching Compatibility

Throughout the project we have chosen Mahalonobis Gradient Compatibility (MGC) as our compatibility measure, which penalizes strong changes in image intensity gradients across piece boundaries by encouraging similar gradient distributions on either side. This pairwise matching cost is widely used by other authors for this particular task. Mahalanobis distance is defined as

$$d(x, y) = \sqrt{(x - y)^T \Sigma^{-1} (x - y)}$$

where Σ is an estimate of the covariance matrix for some data; this implies it is symmetric. If the columns used to estimate are not linearly dependent, 2 is positive definite. Symmetric matrices are diagonalizable and their eigenvalues and eigenvectors are real.

PD matrices have eigenvalues which are all positive. The eigenvectors can be chosen to have unit length, and are orthogonal (i.e. orthonormal) so we can write

$$\Sigma = Q^T D Q \ \& \ \Sigma^{-1} = Q D^{-\frac{1}{2}} D^{-\frac{1}{2}} Q^T$$

Plugging that into the distance definition,

$$d(x, y) = \sqrt{(x-y)^T Q D^{-\frac{1}{2}} D^{-\frac{1}{2}} Q^T (x-y)} = \sqrt{z^T z}$$

4 Linear Programming Formulation

Objective function

$$C^+(\mathbf{x}, \mathbf{y}) = C^+(\mathbf{x}) + C^+(\mathbf{y})$$

$$= \sum_{(i,j,o)\in A} w_{ijo} \left|x_i - x_j - \delta_o^x\right|_1 + \sum_{(i,j,o)\in A} w_{ijo} \left|y_i - y_j - \delta_o^y\right|_1$$

which can be solved separately for x and y coordinates

$$\arg\min_{x,h} \sum_{(i,j,o)\in A^{(k)}} w_{ijo} h_{ijo}$$

$$h_{ijo} \geq x_i - x_j - \delta_o^x$$

$$h_{ijo} \geq -x_i + x_j + \delta_o^x,$$

where δ_o^x is the offset for x coordinates and similarly for y coordinates.

$$\delta_o^x = \begin{cases} 0 \ \ if \ o = 1 \\ -1 \ \ \ if \ o = 2 \\ 0 \ \ \ if \ o = 3 \\ 1 \ \ \ if \ o = 4 \end{cases}$$

$$\delta_o^y = \begin{cases} 1 \ \ if \ o = 1 \\ 0 \ \ \ if \ o = 2 \\ -1 \ \ \ if \ o = 3 \\ 0 \ \ \ if \ o = 4 \end{cases}$$

Where matching weights are decided by:

$$w_{ijo} = \frac{\min\left(\min_{k\neq i}\left(D_{kjo}\right), \min_{k\neq j}\left(D_{iko}\right)\right)}{D_{ijo}}$$

Note that these weights are large when the matching distance between pieces is relatively small and vice-versa.

5 Quadratic Programming Formulation

Once again here we use MGC for compatibility but instead of solving iteratively, we create a Quadratic Program by taking the permutation matrices and creating a global compatibility function, and solving using Gradient Ascent.
The global compatibility function looks like:

$$\varepsilon(\pi) = \sum_{(i,j)\in E_H} C_{H_{\pi(i)\pi(j)}} + \sum_{(i,j)\in E_V} C_{V_{\pi(i)\pi(j)}}$$

where e = (i, j) is the edge connecting the neighboring locations i and j; and $\pi(i)$ corresponds to the tile permuted to location i.

Each permutation π of N elements can be represented as a permutation matrix, i.e., a binary square matrix P, with one entry 1 in each row and column:

$$P_{ik} = \begin{cases} 1, & \text{if } k = \pi(i) \\ 0, & \text{if } k \neq \pi(i) \end{cases}$$

Using this notation, we can reformulate the global compatibility function as

$$\varepsilon(P) = \sum_{(i,j)\in E_H} \left(P^\top C_H P\right)_{ij} + \sum_{(i,j)\in E_V} \left(P^\top C_V P\right)_{ij}$$

If the columns p1,..,pN of matrix P, which has N $\times$ N elements, are stacked up in a column-vector p of dimension N2, we have

$$\varepsilon(P) = \sum_{(i,j)\in E_H} p_i^\top C_H p_j + \sum_{(i,j)\in E_V} p_i^\top C_V p_j$$

We can rewrite it in the canonical quadratic form pAp, where A is a symmetric non-negative matrix of dimension $N_2 \times N_2$ that represents the Hessian of ε(P). In vector form and in coordinates,

$$\varepsilon(P) = \sum_{(i,j)\in E_H} p_i^\top C_H p_j + \sum_{(i,j)\in E_V} p_i^\top C_V p_j$$

Extending the domain of ε(P) for all doubly stochastic matrices, the problem reduces to the solution of the following quadratic optimization problem:

$$\begin{aligned} &\text{Maximize } f(p) = p^\top A p \\ &\text{subject to } P\mathbb{1} = \mathbb{1}, P^\top \mathbb{1} = \mathbb{1}, \text{ and } p_{ij} \geq 0 \end{aligned}$$

The horizontal dissimilarity between two tiles t_i and t_j is computed as

$$D_{H_{ij}} = ((D_{LR}(t_i, t_j))^p + (D_{RL}(x_j, x_i))^p)^q$$

where $\mathrm{G_{ijhR}}$ (k) is the gradient from the right side of ti to the left side of tj, at row k; μ_{iL} is the mean difference between the final two columns of $\mathrm{t_i}$; and $\mathrm{S_{iL}}$ is a 3×3 covariance matrix estimated from the difference between the last column of t_{i} and the first column of t_j. Because dissimilarity D_{LR} is not symmetric, the final horizontal symmetric dissimilarity is

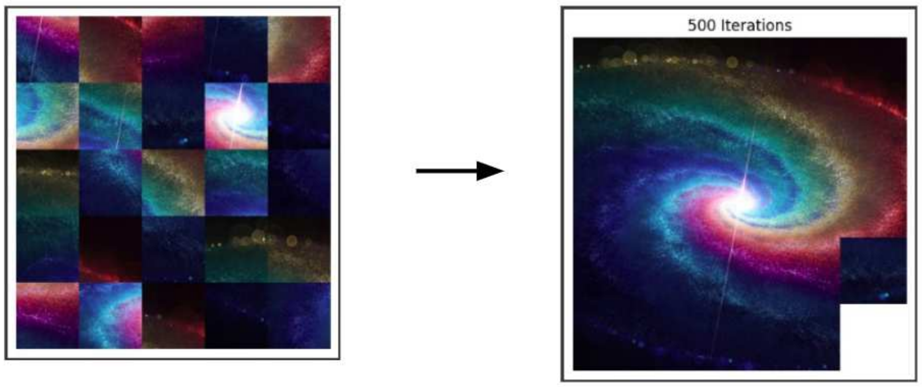
$$D_{LR}\left(t_i, t_j\right) = \sum_{k=1}^{T} \left(G_{ijLR}(k) - \mu_{iL}\right) S_{iL}^{-1} \left(G_{ijLR}(k) - \mu_{iL}\right)^T$$

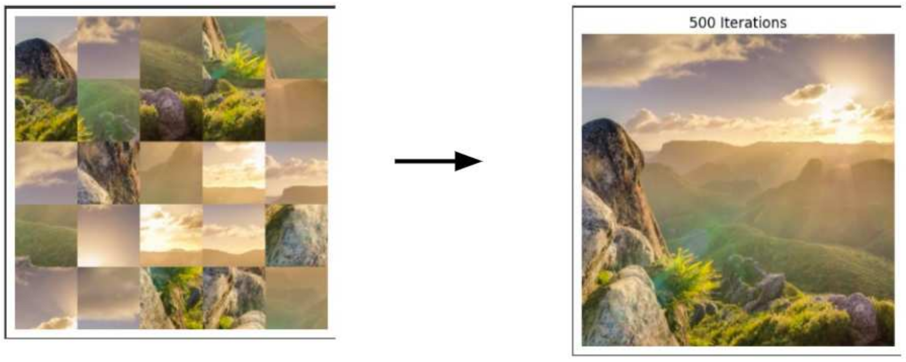
Note that here $D_{LR}(t_i, t_j)$ is the MGC distance with tile i on the left and tile j on the right.

6 Code

Given an input of 500x500 pixel image, we can solve the program for fair number of pieces. The link to the code can be found here.

7 Results





8 References

- H. Freeman and L. Garder. Apictorial jigsaw puzzles: The computer solution of a problem in pattern recognition. Electronic Computers, IEEE Transactions on, (2):118–127, 1964

- C. Gallagher. Jigsaw puzzles with pieces of unknown orientation. In CVPR 2012, 2012

- Yu, Rui & Russell, Chris & Agapito, Lourdes. (2016). Solving Jigsaw Puzzles with Linear Programming. 139.1-139.12. 10.5244/C.30.139.

- K. Son, J. Hays, and D. Cooper. Solving square jigsaw puzzles with loop constraints. In ECCV 2014, 2014.