JIGSAW PUZZLE SOLVER

Sunil Venkatesh Rao, Sanjana Sreedhar, Amritha Subramanian
Fall 2022 ECE 4554/5554 Computer Vision: Course Project
Virginia Tech

ABSTRACT

There has been a lot of research over the past few decades on image stitching and reconstruction, especially to obtain information from shredded documents [1]. Our project, a jigsaw puzzle solver, primarily takes on the same problem with a fundamentally similar approach. Our goal is to solve an unsolved jigsaw puzzle by trying to match the corresponding edges of the puzzle pieces. We have used OpenCV library functions to perform "segmentation", "contouring", "edge detection" and "affine transformations". Our model performed slightly better than the baseline. However, the results were tightly coupled and heavily dependent on the preprocessing stage outputs.

PROBLEM STATEMENT

To develop a software that takes the image of an unsolved jigsaw puzzle with different individual pieces as input and outputs a solved stitched image.

Fig1. Unsolved image

Fig2. Solved image

INTRODUCTION

Solving jigsaw puzzles through computer vision has interesting problems associated with it that are not evident right off the bat. For example, there are real-world problems like the wear and tear of the puzzle pieces that may affect the performance of our model. In this project, the proposed model will aid to solve the jigsaw puzzle when the expected final output image is not available. We will use the image content i.e color information along the edges along with the edge contour information to find the cost function. We feel this will boost the performance of the model. The primary motivation for this project is to evaluate the performance of our model while applying different existing techniques.

APPROACH

The technical approach [2] needed to solve the jigsaw puzzle is decribed in the block diagram.

Fig3. Block Diagram

I. Initial Assumptions

1. All the puzzle pieces are canonical.
2. All the puzzle pieces are unique.
3. The image shall be captured from top view to ensure that the thickness of the puzzle edges are not seen.

II. Image Acquisition

1. All the puzzle pieces are placed on a green screen and image is captured using an iphone camera.

Obstacles faced: While we could capture the image of all puzzle pieces at once from a single picture, the problem with this method is that the greater the number of puzzle pieces, the lower will be the resolution of each piece in the input. This would cause problems while feature extraction. However, if we decided to take individual pictures of each puzzle piece, it would be very difficult to ensure multiple factors such as 'shadows', 'light intensity' and 'scale' to be a constant among the inputs. We found out that the former method had lesser drawbacks over the latter.

III. Image Pre-Processing

1. Create binary mask.
   • Convert image from RGB to HSV space.
   • Apply Median blurring.
   • Perform thresholding to remove the green background.
2. Obtain masked image from input image.

Obstacles faced: The threshold values for HSV are dependent on the input image captured. This stage cannot be generalised for different jigsaw puzzles.

IV. Feature Extraction

1. Perform contour detection using cv2.findContour function. [3].
2. Find the centroid of each image using the image moment for the contour.
3. Convert the cartesian co-ordinates of the contour detected to polar 'rho - theta' space.
4. Perform peak detection for distance curve along the border of each piece in order to find the corners.
5. Extract individual edge contours from the original contour using corner locations.
6. The edges of the puzzle are identified and classified as below [4]:
   • Border
   • Head
   • Hull
7. The puzzle pieces themselves are classified based on the number of borders as below:
   • Corner piece
   • Border piece
   • Center piece
8. Extract the color information along the edges based on the HLS values using colorsys package.
7. Determine the angle of rotation to detect the orientation of the masked image.

Obstacles faced: Harris corner detection output had multiple points detected as corners (even after careful thresholding) which also included the actual corners. We had to come up with a function that would iteratively take a combination of 4 corner points from the pool of points and determine which of them were the real corners. This method was computationally expensive and roughly took 8 to 9 minutes just to find the corners of 1 puzzle piece! Hence we chose the method described in our approach and settled for corners that could potentially be offset by a few pixels.

V. Puzzle Piece Matching

1. Consider a corner piece as the first piece.
2. Determine its potential neighbor by comparing the edge colors and edge contours along all the orientations.
3. Calculate he euclidean distance for both the color and distance vectors between the edges to find a match. The lesser the distance, the better the match.
4. Amongst all the potential neighbors, find the matching piece with the loweset best-fit factor.
5. Store the considered piece and its matching piece as an image pair for further processing.

Obstacles faced: Since the size of each puzzle piece was small, imperfections in the edge contours played a significant role in edge matching. Puzzle piece matching solely based on the contour edge information did not provide expected reults and hence color information along the edges were also considered. Even after having multiple features to determine the cost-function, it was difficult to give weightage to each feature without having the actual test data.

VI. Image Transformation and Stitching

As per our initial proposal application of Homography and Warping techniques did not help with puzzle assemble. We made use of an existing code which would first translate and then overlay the puzzle pieces on a bigger canvas. This portion of the project is implemented with the help of code obtained online from [5].

EXPERIMENTS AND RESULTS

We initially started our experiment by taking a 48 piece puzzle captured as a single image on a green-screen. We had a tough time detecting corners of all 48 images. We could successfully find corners only for about 46 pieces or so despite careful thresholding and iterative methods. Thus, to overcome this, we decided to capture each puzzle piece as a separte input so that we have greater detail along the edges. While we were able to find all 48 corners accurately, puzzle piece matching became a problem because of the difference in scale factor of each piece. Also, since we had to deal with bigger contours, the algorithm had to deal with a few thousand pixels along each edge which drastically slowed the matching process. With the above said problems, we decided to go for a smaller puzzle with only 20 pieces. The image was taken on a single green-screen background. With the output of the pre-processing stage we were able to determine the corners with a method as decribed in [6]. A few of the intermediate stage results while trying to determine the filter sizes are shown as below. One can notice the slight offset in the corner locations detected as a consequence of image noise.

Fig4. Solved image with median filter size = 25

Fig5. Solved image with median filter size = 21

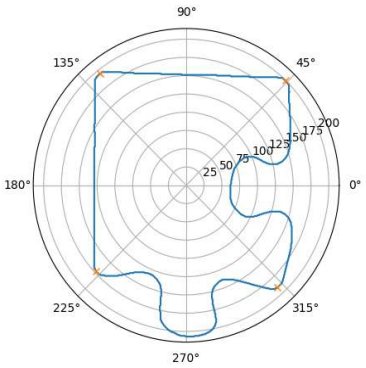

Fig6. Corners detected

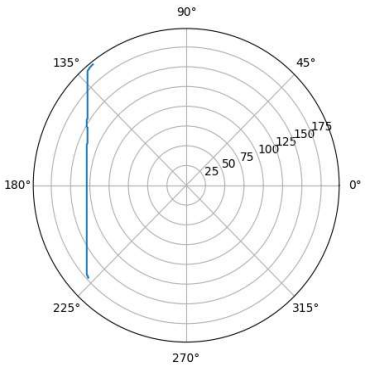

Fig7. Extracted edge 1

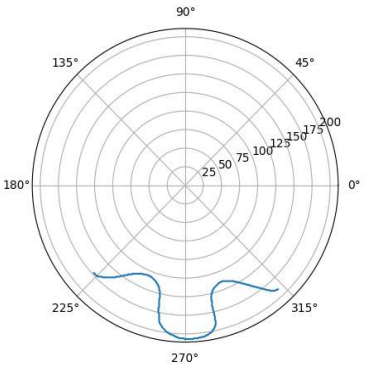

Fig8. Extracted edge 2

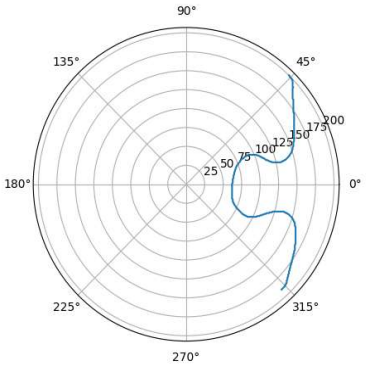

Fig9. Extracted edge 3

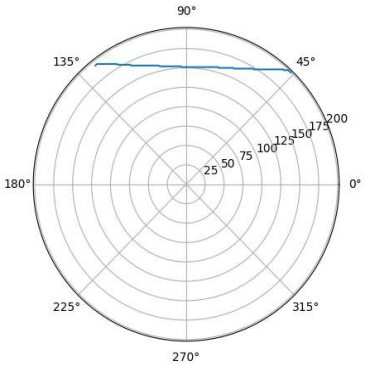

Fig10. Extracted edge 4

The corner piece is picked as the first piece to solve. Then the corresponding possible border piece to its left is matched. The test pieces are rotated in order to get the right orientation. The rotation angle is calculated based on the relative angle i.e. angle between two adjacent points on the edge. The compatibility is detrmined based on the head, hull and straight edge information. After finding a match for the piece the best fit value is calculated based on the euclidean distance between the corresponding points on the edges. The euclidian distance of color information on the corresponding puzzle pieces are calculated as well [7]. The piece with the least distance in both the scenarios is considered to be the perfect match. Once the pieces along the borders are placed the centre pieces are validated for compatibility.

The final result validation method is simply based on human evaluation of a correct solution. One thing that is to be considered is that the orientation of the final image may be rotated by a factor of 90 degrees.

QUALITATIVE RESULTS

Fig.11 - Input image.

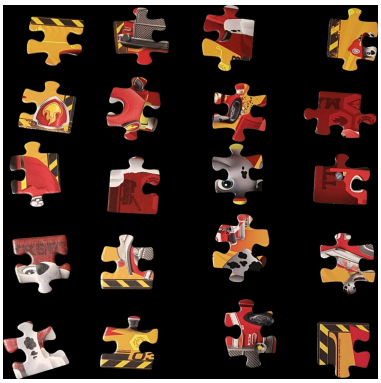

Fig.12 - Green background removed.

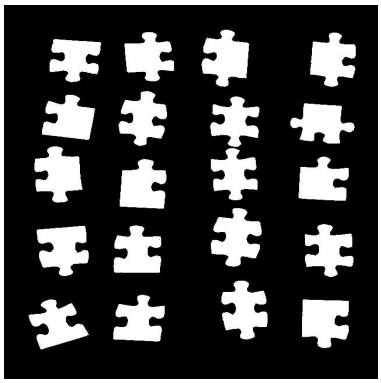

Fig.13 - Binary masked image.

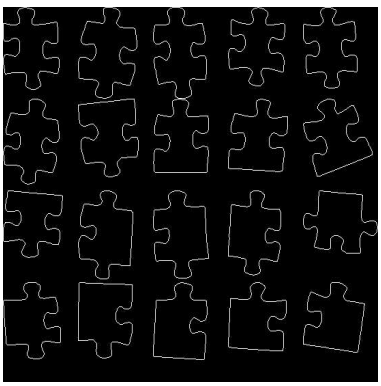

Fig.14 - Contour detection.

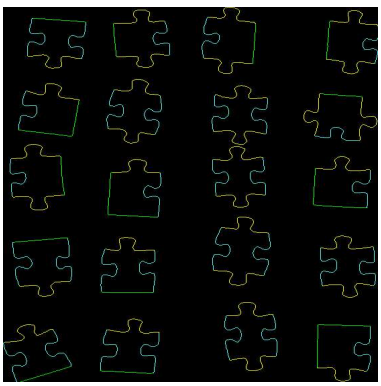

Fig.15 - Edge classification.

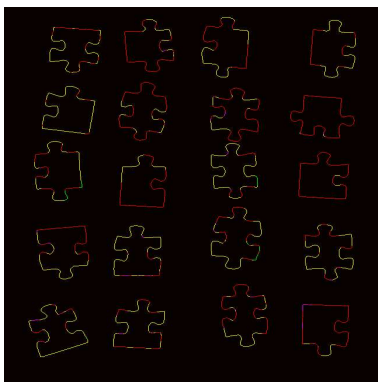

Fig.16 - Colored borders.

Compatibity validation

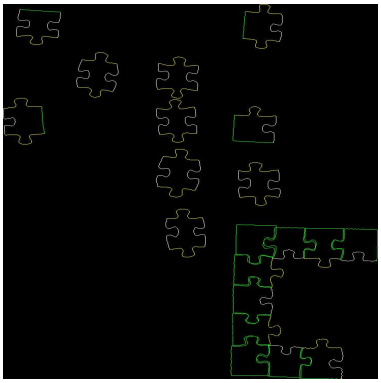

Fig.17 - Stage I.

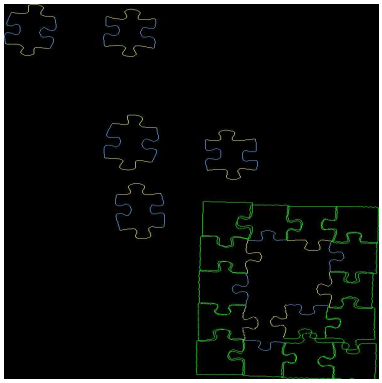

Fig.18 - Stage II.

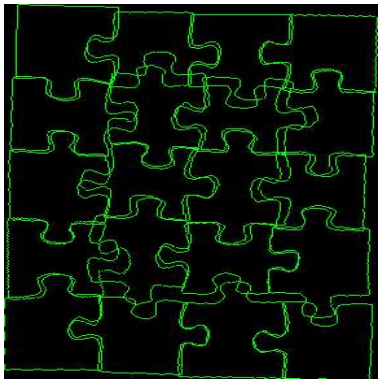

Fig.19 - Stage III.

Puzzle Assembly

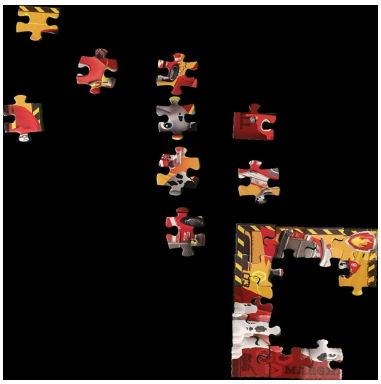

Fig.20 - Stage I.

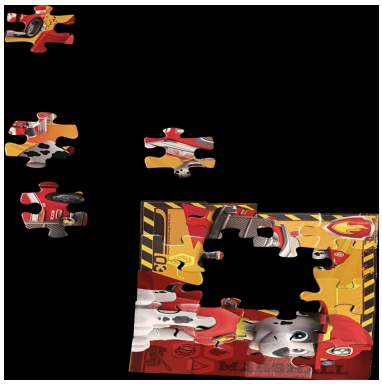

Fig.21 - Stage II.

Fig.22 - Stage III.

CONCLUSION

As a result of this project we have gained a great deal of knowledge by implementing the theoretical approaches to model a practical solution. This report has described the methodology to solve unsolved jigsaw puzzle when the expected output is not available. This is implemented by performing image masking, contour detection, corner detection, edge detection, color information extraction, rotating the puzzle pieces in right orientation followed by puzzle assembly. Despite of all the obstacle we faced during the course of this project, we were pushed to think out of the box and come up with solutions based on Computer Vision techniques.

REFERENCES

[1] Pan, Zixiao & Wang, Mei. (2017). A New Method of Shredded Paper Image Stitching and Restoration. 55-58. 10.1109/ICIICII.2017.33.
[2] Solving jigsaw puzzle using Computer Vision
[3] Contour Detection
[4] Using Computer Vision to Solve Jigsaw Puzzles
[5] Zolver
[6] D. A. Kosiba, P. M. Devaux, S. Balasubramanian, T. L. Gandhi, and K. Kasturi. An automatic jigsaw puzzle solver. In Pattern Recognition, 1994. Vol. 1 - Conference A: Computer Vision amp; Image Processing., Proceedings of the 12th IAPR International Conference on, volume 1, pages 616-618 vol.1, Oct 1994.
[7] Jigsaw puzzle solver using shape and color
[8] Using Computer Vision to Solve Jigsaw Puzzles
