DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

ENEE699-DIGITAL IMAGE AND VIDEO PROCESSING

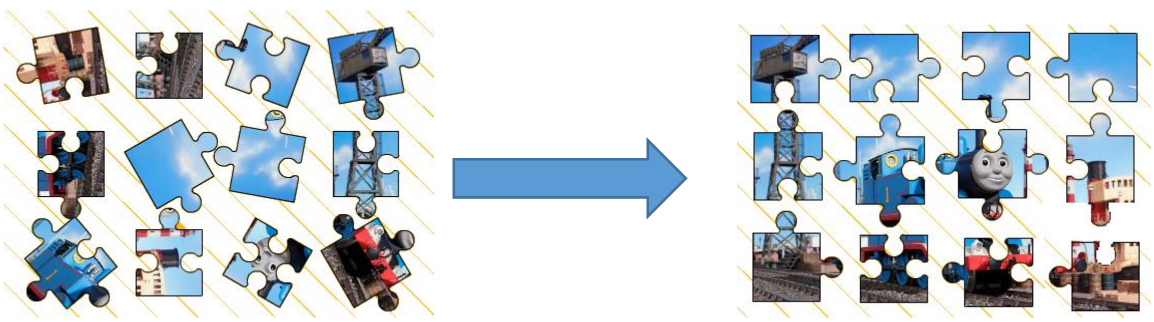
NITHYANANDA BHAT KUMBLA
UID: 113670732

FINAL-PROJECT-REPORT

Automatic Jigsaw Puzzle Solver

Abstract

Jigsaw puzzle is one of thought-provoking games where an image is broken down into smaller pieces, not necessarily of the same shape, and the challenge is to assemble them back in correct order. The image on the assembled piece should look alike to a reference image (sometimes absent). In this project, the task is to assemble pieces of a jigsaw puzzle using Image Processing Techniques. The output of the Matlab code is the correct sequence of the pieces that are laid out in random order before the task. A couple of assumptions (mentioned in detail later) are considered true before beginning the task. If the puzzle is rectangular and the number of pieces in the puzzle is known then the number of corner pieces and centre pieces can be derived mathematically. It is logical that the puzzle can be solved even without the presence of the image for the simple fact that the pieces can be put together in the same order as the solution, while facing down. For the initial processing, the image on the pieces are masked and only their shapes are analysed. The 2 essential components for computationally solving a jigsaw puzzle, a measure of jigsaw piece compatibility for adjoining a pair of jigsaw pieces (solved using characteristics of shape) and a strategy for puzzle assembly (using image on the puzzle pieces). A 12-piece puzzle is considered in this task. An Android device is used to capture the Image, which is transferred to Matlab over local IP host. The acquired image is then subjected to image processing techniques like Image Enhancement, Thresholding, Segmentation, Edge Detection, Feature Extraction etc. A self-developed algorithm then analyses the characteristics of these pieces to find the best match in the available set of pieces. The piece that this algorithm chooses is then verified with the reference image (solved version of the puzzle). The Conclusion of the task will be an ordered array of pieces that form the solution to the puzzle. In the report a computer image of the puzzle is considered for explaining the algorithm in detail. The performance of the algorithm is also shown towards the end.



Source: sproutonline.com/games/thomas-twelve-piece-jigsaw-puzzle

Introduction:

Puzzles are great means to initiate brain development and growth for the reason that the brain looks for patterns in our world — and puzzles are best means of involving in true patterning activity. Patterning is a very essential foundation for building reading, math and logic skills. This can lead to development of problem solving skills and individual success. [1a]

Jigsaw Puzzle is a picture printed on cardboard or wood and cut into various non-overlapping, interlocking pieces. The challenge is to fit these pieces together to form the picture again. Recreational solving of jigsaw puzzles belong to the class of problems for which humans have a natural aptitude but automation remains considerably more challenging. The complicated features of the template, irregular shape, unknown orientation and scaling of the pieces all contribute to the challenge of this game. Given the well-developed algorithms of detecting and analysing images, the enormous capacity of processing speed and memory, computer vision may provide a valuable aid for human jigsaw puzzle players to detect the patches in the template.

To overcome the challenges of solving jigsaw puzzles, pattern matching algorithms are required to be invariant to scales, rotations, and have a good tolerance with background clutter. The other task image processing algorithms need to carry on is to register the patch image properly with the template. The transformation from the patch to the corresponding part in the template needs to be properly constructed from the matching feature points. [1]

While solving a Jigsaw puzzle our brain processes several data related to shape compatibility and image continuity. My Intuition towards taking up this project is to enable the computer to think on these lines and solve a puzzle by itself. I have proposed an algorithm which relies both on the shapes of the pieces and on the image present on the piece. At the opposite end of the spectrum are algorithms that deal solely with image information, on puzzles with all square pieces, [2] or deal only with the shape of the puzzle. [2a]

A brief explanation on the Characteristics of Jigsaw Puzzle

Every Piece in the Jigsaw Puzzle is unique. No two pieces can fit into the same place given all the characteristics that constitute the piece.[3a] The pieces in a puzzle are said to be interlocking when the edges of the pieces form a locking combination form after the pieces are put together in the puzzle. They can be set or freed only in a particular orientation. This feature is mostly helpful to keep the solved part of the puzzle intact during the process. The shape of the interlock is generally called Knobs. The set of the available shapes of knobs (also known as tabs) is merely countless, depending only on the mercy of the imagination of the puzzle maker. In the task the convex protrusions are called heads and concave depressions are called holes. Figure 1 shows pictorial representation of heads and holes.

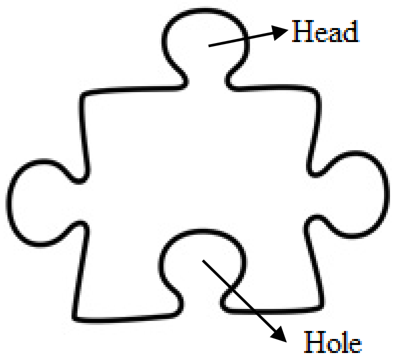


Figure 1: A general Jigsaw Puzzle Piece.

Assumptions for the project

1) Each piece of the puzzle has only 4 sides.
2) Each puzzle is kept within the marked grid Line.
3) For a 12 piece puzzle- The pieces are to be arranged in 3 rows and 4 columns.
5) The Puzzle can take any orientation inside the Grid line.
6) There is no major wear and tear along the edges of the puzzle
7) The Image acquired from the camera is exactly vertical and there are no perspective effect on the image
8) All pieces are rectangular in shape (Diagonals between opposite corners should cut at 90 degrees) and there can be only zero/one head or hole between each corners.
9) The reference image is given for verification. (A method is also proposed in case the reference image is not available).
10) The first piece in the arrangement is always the first piece in the solved puzzle

Solution Flow Chart

The Process begins with an image of a jumbled arrangement of the Jigsaw Pieces. The Background in the image is eliminated and the image pieces are processed to get the outline of the individual pieces. The 4 edges are then separated and then matched with every possible edge of the other pieces and a best possible match is chosen. For verification the image on the piece is compared with the reference image for its correctness in position

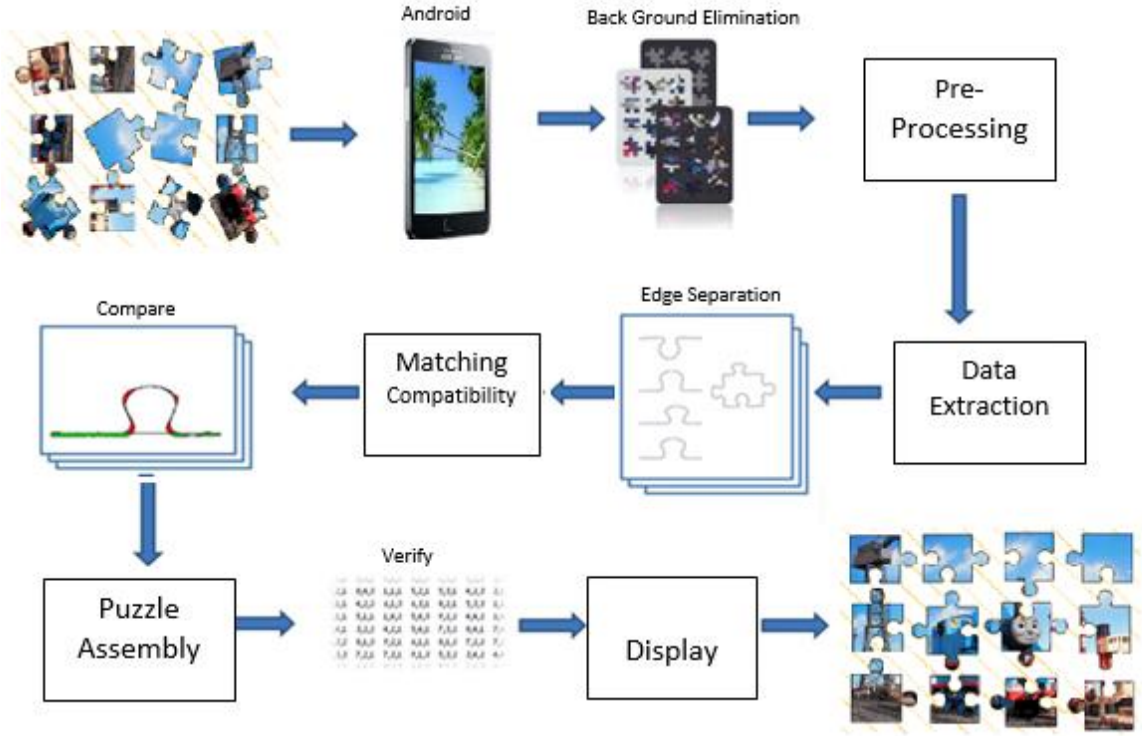


Figure 2: Flow Chart of the Process

Explanation of Process

1. **Image Acquisition**

This is the first task in the Project. The pieces of the puzzle are to be placed on a specially designed background paper. An Android app named 'IP Camera' is necessary to perform this task (This can alternatively done be saving the image acquired on the android device on the computer and performing an 'imread' operation). The resolution in the settings is to be set to maximum possible level to get a good quality image. All the assumptions are to be validated before proceeding with this step. Figure 3 represents a sample grid line arrangement. All samples should necessarily be within each grid. The black lines are overlaid on the monitor screen to guide the user to set the pieces within them. These lines are imaginary and will not be present in the final version of captured Image.

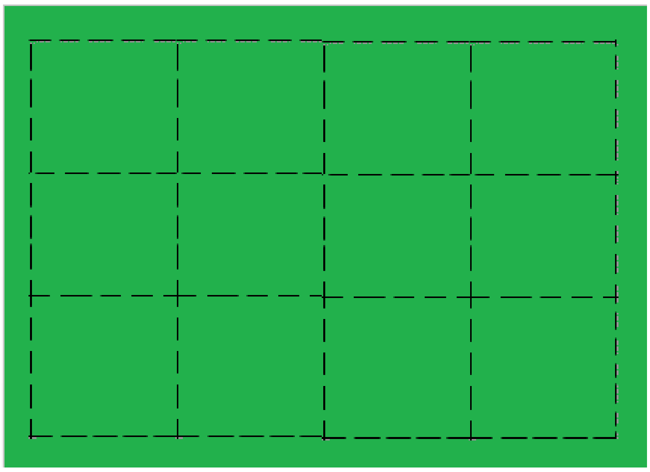


Figure 3: Representation of Background before placement of Jigsaw Pieces and the Imaginary Grid Lines overlaid on the Image.

The image acquired should not have any perspective effect of Image. The IP address displayed on the android screen must be entered into the Matlab File so that it can receive data from that IP address. The image acquired is sent to Matlab application running on the computer over local IP network.

Figure 4 is the computer version of the puzzle. All figures in the explanation are based on a computer version of the jigsaw puzzle (For clarity during explanation). An actual demo version of the images are shown at the end of the report.

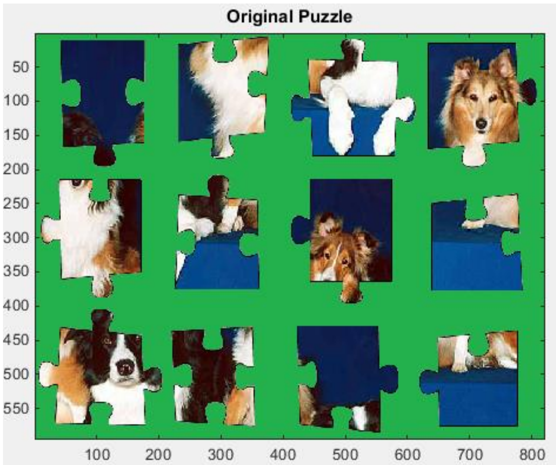


Figure 4: A Computer Version of the Jigsaw Puzzle

Code Snippet for this Image Acquisition

```
url = 'http://10.0.0.32:8080/shot.jpg';

while(1)
    ss  = imread(url);
    imshow(ss);

   [rows, cols,map]=size(ss);
    r3=rows/3; c4=cols/4;
    a=[r3 r3 2*r3 2*r3 1 rows 1 rows 1 rows];
    b=[1 cols 1 cols c4 c4 2*c4 2*c4 3*c4 3*c4];
    hold on;
    for i=1:2:10
        l = [a(i) a(i+1)];  m = [b(i) b(i+1)];
        plot(m,l,'Color','w','LineStyle','--');
    end
    hold off
end
```

2. Back Ground Elimination and Pre-Processing

The first task in the pre-processing stage is to extract the pieces from the image. For this task, only the background of the puzzle is retained. Then a thresholding operation is performed to generate the mask. This mask is then inverted and then multiplied with the Original Image to get the pieces with white background. For the rest of the Puzzle solving this image and the mask is used. Canny's Edge Detector Algorithm is applied on the mask to obtain an outline of all the pieces present in the image. Since the individual pieces are present inside the Grid Lines, they can be separated into separate images. The individual pieces are displayed after numbering every piece.

The step by step flow under pre-processing is

- Retain only Background colour and apply thresholding to generate mask
- Multiplication of Inverse of the Mask with original Image
- Applying Canny Edge Detection on mask
- Bounding Box and Centroid on the edge extracted image
- Segmentation to separate the pieces
- Representation of the pieces

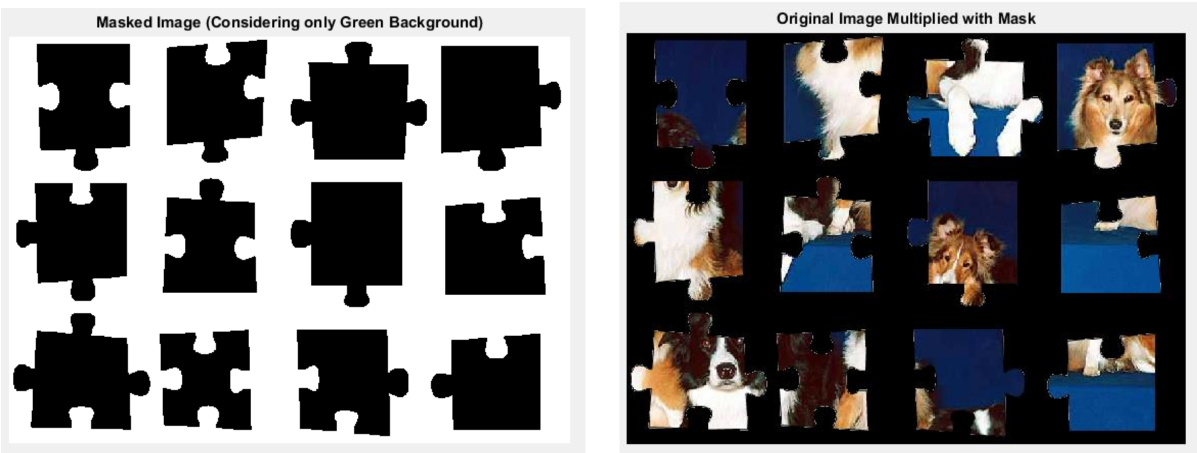


Figure 5: The mask generated by Back Ground Elimination and the corresponding image got by multiplying the inverse of the mask with the original image

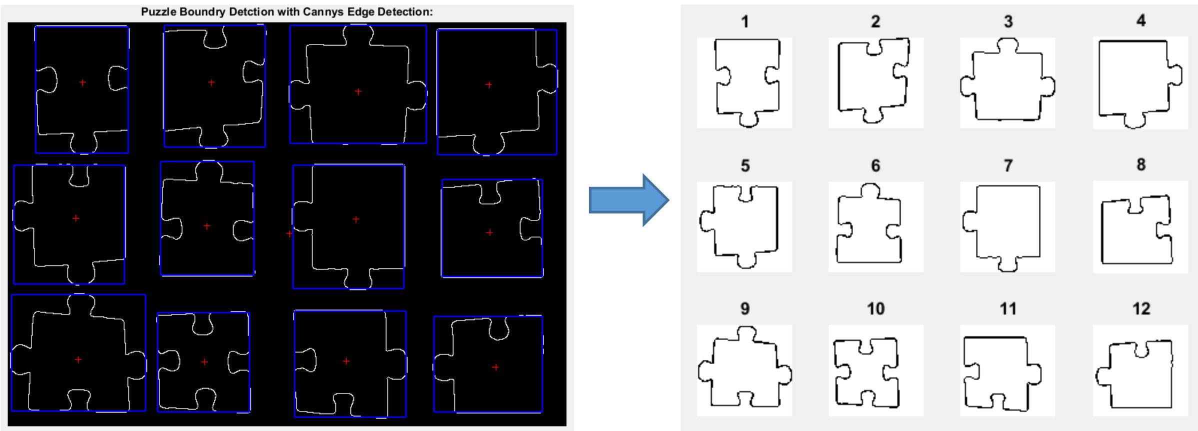


Figure 6: The image got from Canny's Edge Detection and its inverse image after the pieces are separated.

Code Snippet for removing Background and generating the mask

```
yd = double(ss)/255;

greenness = yd(:,:,2).*(yd(:,:,2)-yd(:,:,1)).*(yd(:,:,2)-
yd(:,:,3));
% Threshold the greenness value
thresh = 0.3*mean(greenness(greenness>0));
mask = greenness > thresh;
figure, imshow(mask);
```

3. Data Extraction and Edge Separation

In this third stage we extract and encode the outline of each piece in a way that will enable us to later compare the edges of two pieces and conclude whether they fit together. The encoding is done in four stages: First, we identify all the edge pixels. Second, we plot a polar plot of this contour, apply a gradient operator. Third we walk counter clockwise over the contour and determine the 4 corner of the piece. This process ensures that the algorithm is invariant to orientation of the pieces within the grid lines and finally, since each edge of the piece is compared separately.

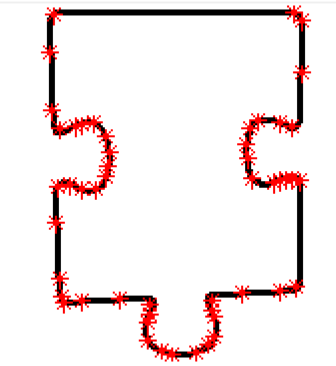
The step by step process in this stage are

- Boundary Detection and convert to phase space
- Corner Detection using differential method
- Edge Separation using corner points
- Orientation detection and rectification
- Edge length Estimation using Euclidian Distance
- Edge Shape Characterization (Head or Hole or Corner Edge)

Code Snippet for Detecting Boundary Points and plotting them in polar co-ordinates

```
% The Mask of every piece is considered for Boundary Detection
B = bwboundaries(ImagePrime, 8, 'noholes');
B = B{1};

% convert boundary from cartesian to ploar coordinates
objB = bsxfun(@minus, B, mean(B));
[theta, rho] = cart2pol(objB(:,2), objB(:,1));
```



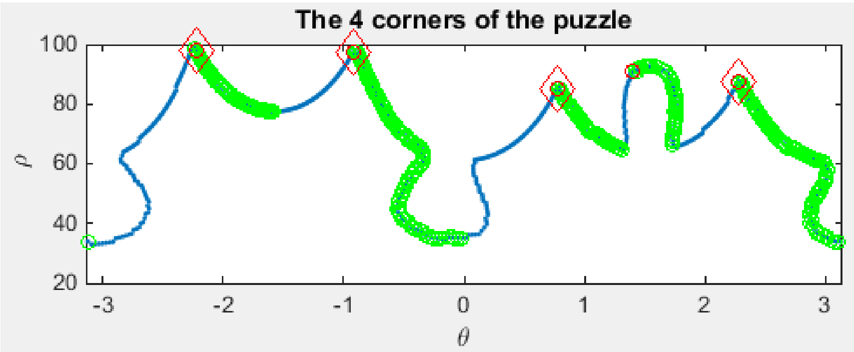


Figure 7: The result of Boundary detection for a piece and its corresponding phase plot

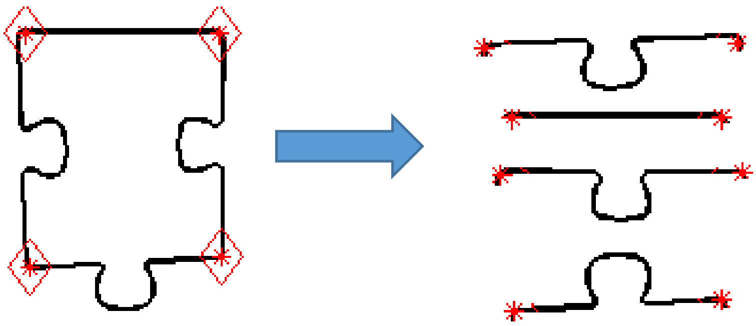


Figure 8: The result of computation of corner detection from the Phase plot of the Boundary points and edge separation

length=

140.0000	212.0122	219.9828	156.4142
217.3259	208.9117	216.8112	142.4142
167.1838	203.6690	131.0000	209.2548
134.0000	216.4264	195.2548	224.9117
171.3553	136.8284	196.4975	216.2548
190.0833	196.2548	135.0000	207.9117
193.6569	151.8284	201.1838	150.3553
210.3259	212.6690	147.8284	128.4142
84.9706	207.4975	327.7107	878.1615
96.0416	216.8406	193.2548	200.2548
130.0000	224.0833	204.2548	213.4264
167.0833	141.8284	125.0000	193.0122

angles =

-2.3085	-0.9466	0.7252	2.4144
-2.4002	-0.8430	0.7322	2.3750
-2.4508	-0.6626	0.8792	2.2432
-2.2319	-0.9082	0.8007	2.2737
-2.2273	-0.8004	0.6724	2.2102
-2.3251	-0.7052	0.8417	2.3012
-2.1657	-0.8646	0.7165	2.2812
-2.4905	-0.8660	0.8011	2.4092
-2.9642	-1.7198	-0.0292	2.4133
-2.3718	-0.8900	0.8652	2.3024
-2.3019	-0.9497	0.9051	2.3839
-2.3234	-0.7621	0.8072	2.2223

Figure 9: The computed values of the length of each edge and the angle of the corner points in the phase plot for all 12 pieces

Since we have assumed that every piece in the puzzle in the puzzle is rectangular, it is evident that each corner of the puzzle is at a phase difference of approximately pi/2 radians. This property can be exploited to eliminate the sharp points generated by the heads in the puzzle.

For example in the Figure 7, the corner point at theta = 1.5 radians is considered not a corner because the phase difference between itself and the previous corner is very much less than pi/2 radians, which is correct because this point corresponds to the head of the puzzle and not the corner point.

3.1 Orienting the piece – Making the initial orientation invariant

The initial assumption is that the jigsaw pieces can be in any orientation within the grid lines. But it is very much necessary for the analysis that all the pieces are aligned in the same orientation during the matching process. Hence every piece is rotated such that the angle of occurrence of the first corner point is at -3pi/4 in the phase plot.

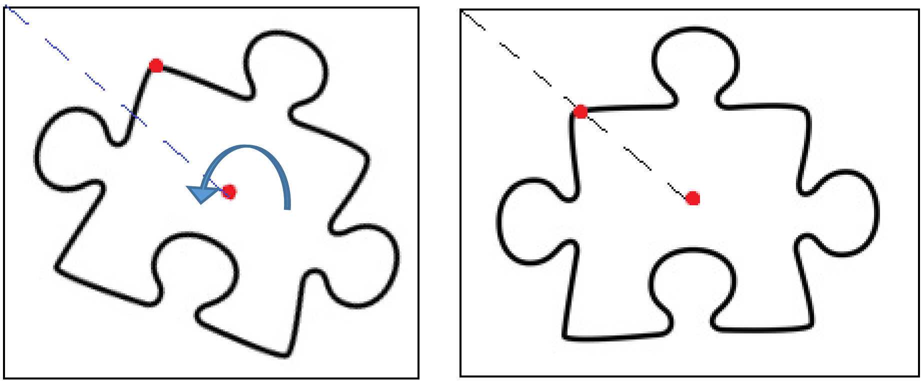


Figure 10: Pictorial Representation of the random orientation of the jigsaw piece and aligning it to a common orientation

3.2 Edge Shape Characterization

In this section we study how the points along the boundary from one corner point to the other vary in terms of perpendicular distance with the line joining the corner points. There are 3 cases in this computation

- Corner Edges – All the points on that edge along the boundary lie on the line joining the corner points
- Hole Edges – The points lie either on the line or below the line joining the corner points
- Head Edges - The points lie either on the line or above the line joining the corner points

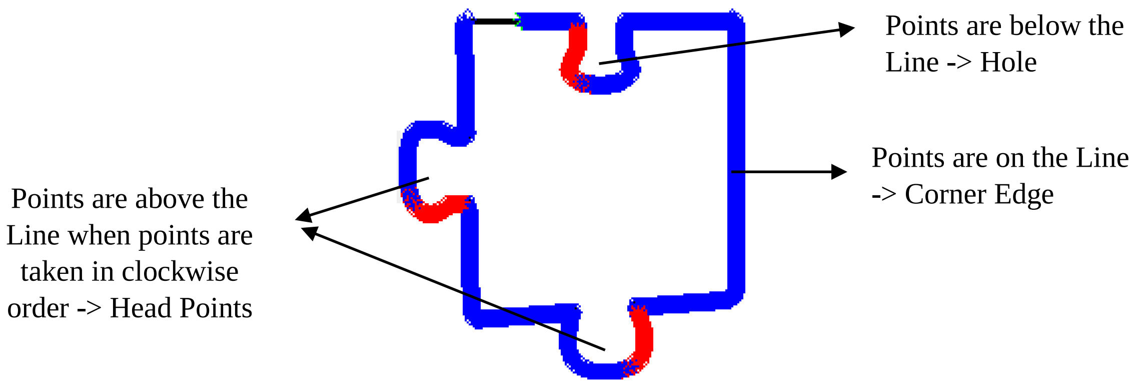


Figure 11: Edge shape characterization of edge in the piece. The red marks indicate the interesting points and the blue dots indicate the non-interesting points in edge characterization

shape =

0	-1	-1	0
1	1	-1	0
-1	-1	0	-1
0	1	-1	1
1	0	-1	-1
-1	1	0	1
1	0	-1	-1
1	1	0	0
1	-1	-1	1
1	1	1	1
0	-1	1	1
1	0	0	-1

Figure 12: The characterization of shape for the considered puzzle pieces (1, -1 and 0 denote Concave, Convex and Corner Edge Respectively)

FinalX =

18.0000	157.0000	164.0000	15.0000
21.0000	168.0000	162.0000	20.0000
26.0000	175.0000	167.0000	36.0000
31.0000	164.0000	163.0000	35.0000
34.0000	166.0000	168.0000	38.0000
23.0000	151.0000	152.0000	17.0000
42.0000	173.0000	175.0000	42.0000
21.0000	159.0000	167.0000	20.0000
2.0000	87.0000	199.0000	24.0000
11.0000	140.0000	144.0000	19.0000
10.0000	139.0000	143.0000	12.0000
32.0000	161.0000	159.0000	34.0000

FinalY =

7.0000	7.0000	153.0000	164.0000
13.0000	1.0000	141.0000	155.0000
36.0000	39.0000	178.0000	178.0000
4.0000	4.0000	157.0000	163.0000
13.0000	13.0000	149.0000	162.0000
35.0000	44.0000	176.0000	176.0000
12.0000	12.0000	163.0000	165.0000
48.0000	34.0000	178.0000	176.0000
86.0000	7.0000	101.0000	173.0000
38.0000	31.0000	180.0000	173.0000
30.0000	30.0000	188.0000	176.0000
46.0000	39.0000	180.0000	180.0000

Figure 13: The Final Value of the Corner Points after re-orienting the pieces

4. Matching Compatibility Computation

This block pre-calculates a match compatibility grade between each feasible match of two edges. This degree of match (grade) is calculated by placing both edge outlines on top of each other and counting the pixels of the areas where they are different. The steps covered in this stage are:

- Fundamental concept for Matching
- Procedure for joining the pieces of the puzzle (Match left and Top)
- Rotate for Matching
- Verification for matching

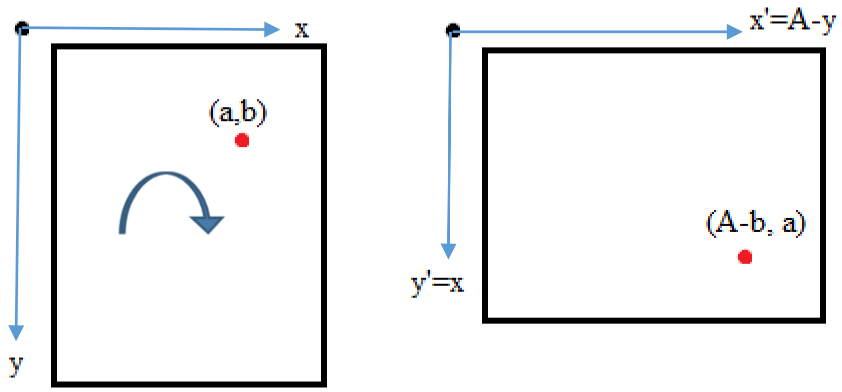


Figure 14: A pictorial representation of the mathematical computation in the co-ordinates of the corner points during rotation of a rectangular piece

Assuming that the size of the Image is (A, B). So the piece 2 has to be laid exactly on edge of the 1st piece to check if they are a correct order pair. The translation in X and Y is given by Transformation T.

$$T = [X, Y]$$

$$X = x2 - x1$$

$$Y = y2 - y1$$

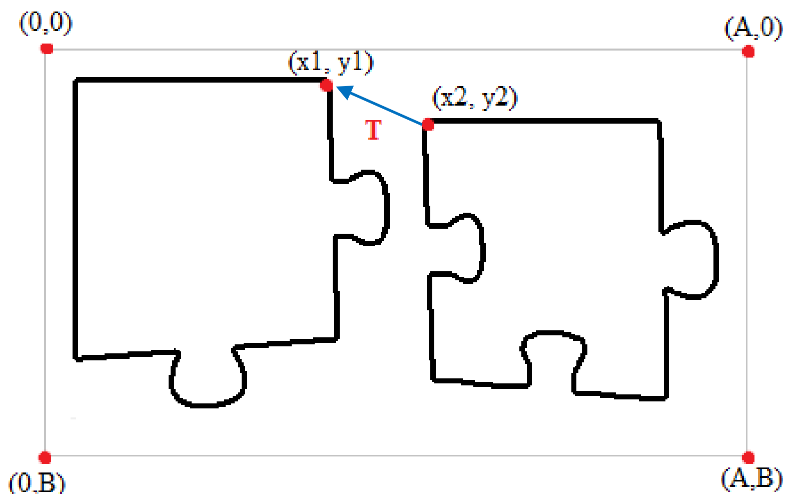


Figure 15: Pictorial Representation of the translation computation needed for aligning the edge of one piece with the other.

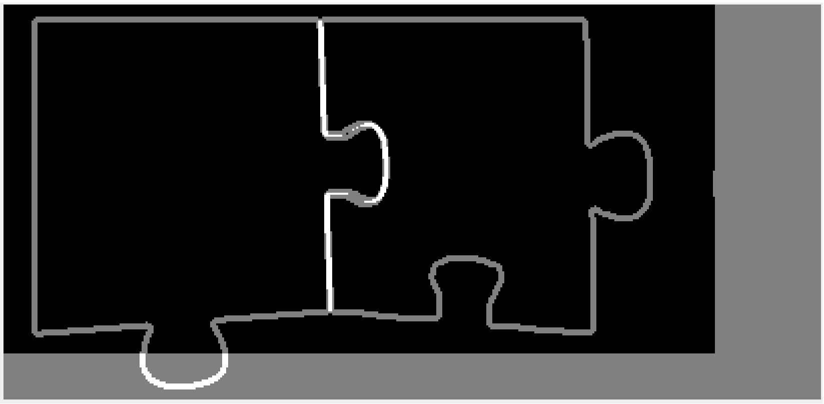


Figure 16: Pictorial Representation of result after the translation computation. 'Imfuse' function in Matlab is used to merge the 2 pieces together

Code Snippet for Rotating Puzzle for matching operation

```
function Takeaway=rotateForMatching(Piece)

    % Get the Image of the Piece being Processed
    jigsaw=strcat('image', num2str(Piece));
    [x1,x2,y1,y2]=limits(jigsaw);
    jigsaw = IOrient(x1:x2, y1:y2);


    % Rotate the image -90 degrees
    jigsaw = padarray(jigsaw,[5 5],'replicate');
    Takeaway=imrotate255(jigsaw,-90, 'bilinear','crop');
    Takeaway=Takeaway(5:(x2-x1+5), 5:(y2-y1+5));

    % Update the Image for Future Reference
    IOrient(x1:x2, y1:y2)=Takeaway;

    % Change the co-ordinates of the Corner Pieces
Accordingly. The corner points should also be shifted
circularly to account for this rotation
    shape(Piece, :)=circshift(shape(Piece, :),1,2);
    angles(Piece, :)=circshift(shape(Piece, :),1,2);
    temp=FinalY(Piece, :);
    FinalY(Piece, :)=circshift(FinalX(Piece, :),1,2);
    Dim=Flip(Piece);
    for nit=1:4
          temp(nit)=Dim-temp(nit);
    end
    FinalX(Piece, :)=circshift(temp,1,2);

end
```

5. Puzzle Assembly and Verification to matched piece

For assembling the puzzle, we start with the first row and first column, and iterating over the column. Once all pieces in that column are matched we move on to the next row and this process continues till the end of the puzzle. At every decision point the two edges that get matched are the ones with that have best fit in terms of shape and least error (PSNR-Peak Signal to Noise Ratio) in terms of image is selected. This means that the compatibility difference between the best match is maximal in relation to all other possible matches.

Let us first consider the shape check for finding a match. Assuming that the piece to be matched is in the first row and second column. Naturally the piece to the next of it should have the complimentary shape at the interlock and should have a boundary edge at the top. Figure 17 represents the match finding process. Each row indicates a pieces that is considered and rotated 4 times to check for a match on all 4 sides. The images marked with a star indicates that they have passed the test for Shape Analysis.

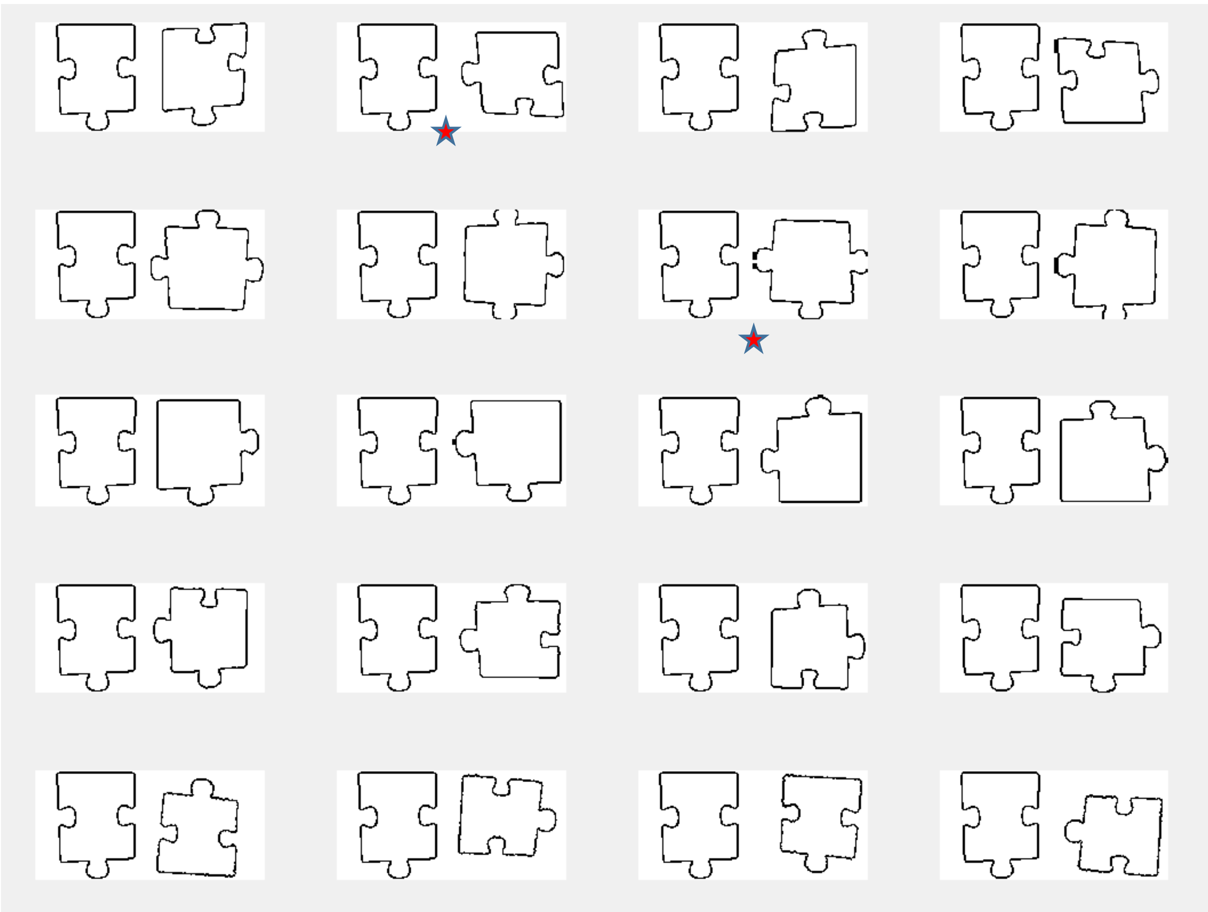


Figure 17: A Representation of Match finding by Shape Analysis

Now, once we have shortlisted the pieces and the orientation in which they match, we have to move to the image content of the pieces to find the correct match in the shortlisted pieces. There are 2 cases in Image Analysis-

1) When the Reference image is not given

The idea considered for puzzle assembly is to first verify the shape match between the piece in the puzzle and the one that is being matched. Once the shape match is found, the image in the puzzle is verified with the reference image, using Block Matching

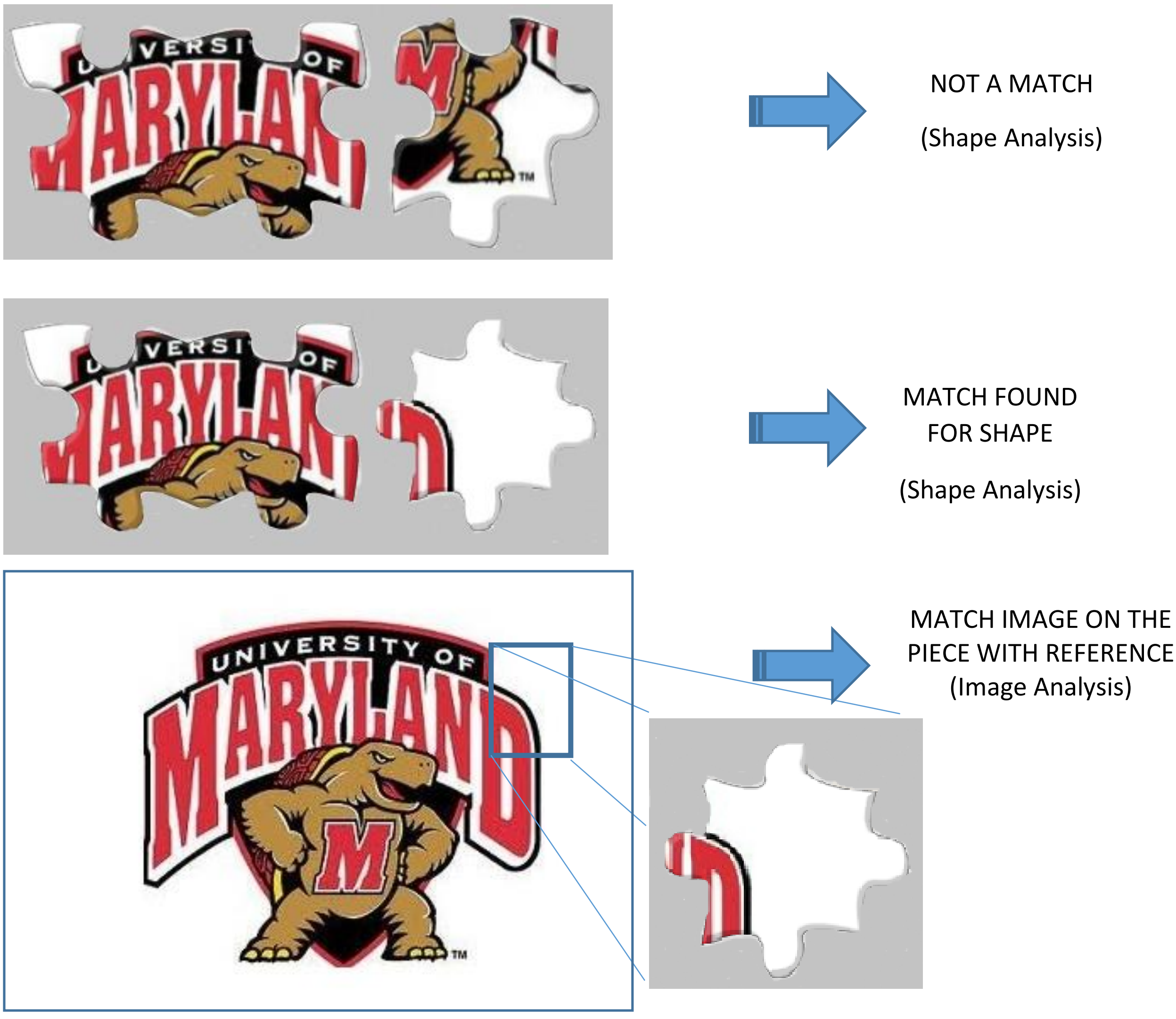


Figure 17.1: A detail on the match computability check and verification with reference image

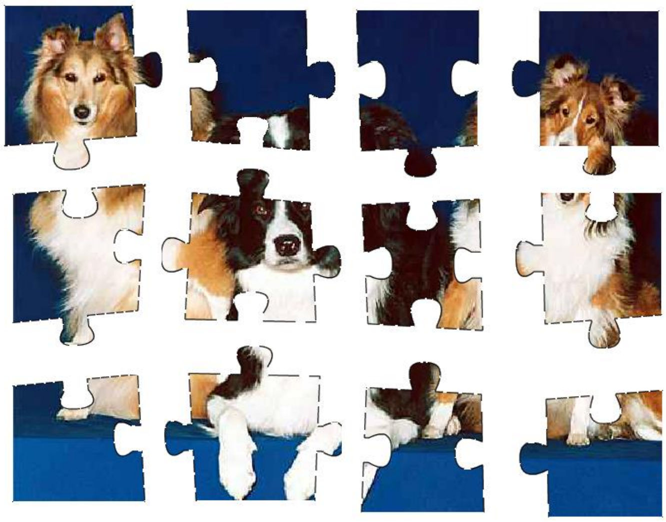


Figure 17.2: The Reference Image for the Computer Puzzle

2) When the Reference image is not given

When the reference image is not given, it is natural for us to check the match for the pieces by checking if the placement of the piece would make sense in terms of the image content. But, it is a challenging task for the computer to check for continuity in the image content which checking for a match in the pieces. While I have not been able to implement this method of Jigsaw Assembly, I have an intuition that the method can be executed by taking the contour plot along the boundary of the piece. For imagination, we can take this as sampling the intensity variation of the patch near the edges of the puzzle and using this characteristic for matching. If the intensity of the contour plots match after the shape verification step, then that piece can be considered as a match. To visualize this, a slice along the edge of the puzzle should have a continuous curve (or close enough with in a threshold value) all along the edge for 2 pieces to match.

6. Display

The last stage is displaying the constructed puzzle. Stage 4 had found the position of each piece in relation to other pieces, and now we need to construct an image which shows the right order of arranging the pieces. Each piece is rotated in the right orientation. This final image will give us an idea as to where the pieces in the Grid are to be placed to solve the puzzle.

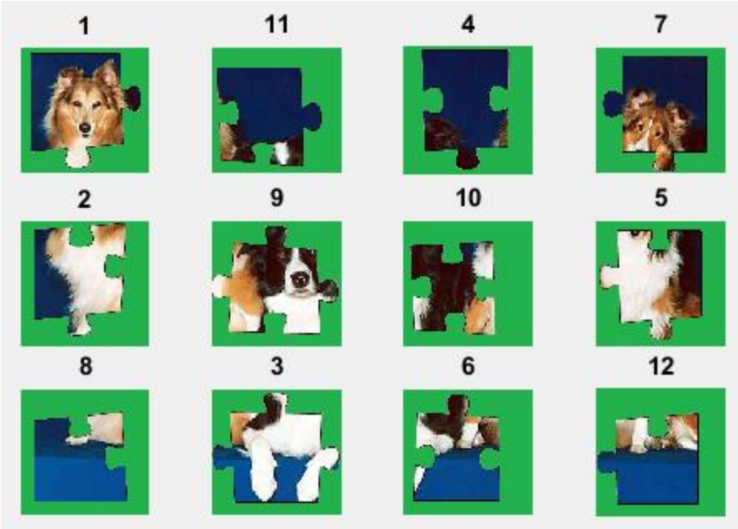


Figure 18: The final arrangement of the pieces to solve the puzzle.

The ordering of the puzzle pieces that would lead us to solve the puzzle

```
The Original Ordering is:        The solution to the puzzle is:

     1     2     3     4              1    11     4     7
     5     6     7     8              2     9    10     5
     9    10    11    12              8     3     6    12
```

A Demo of the algorithm on an Actual Puzzle

Figure 19: Screen Shot of the Android Device- IP Camera Application

Figure 20: The Image acquired on Matlab from the IP Camera Application

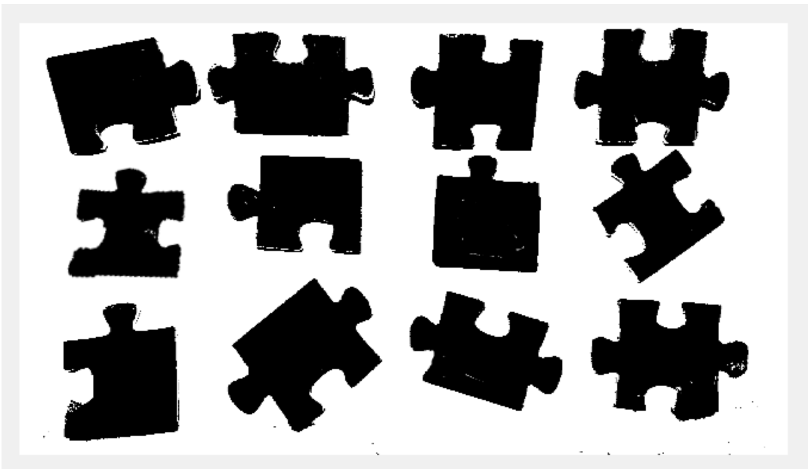


Figure 21: The Mask generated by Background elimination

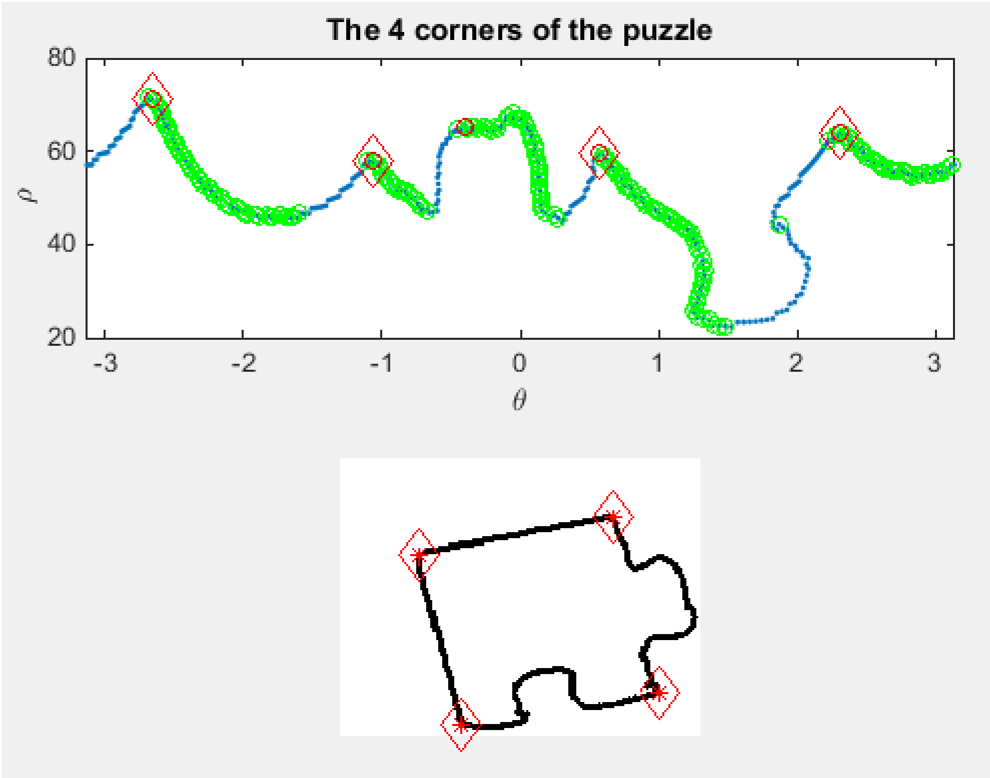


Figure 22: One Example for Boundary Detection, Phase Plot and Corner Detection

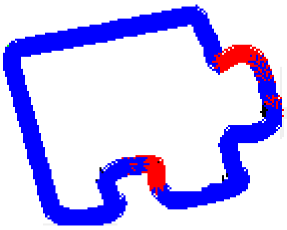


Figure 23: Characterizing the shape of the piece

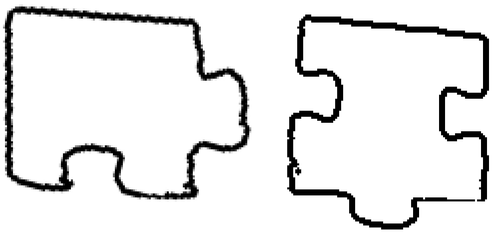


Figure 24: Match Finding using Shape Analysis

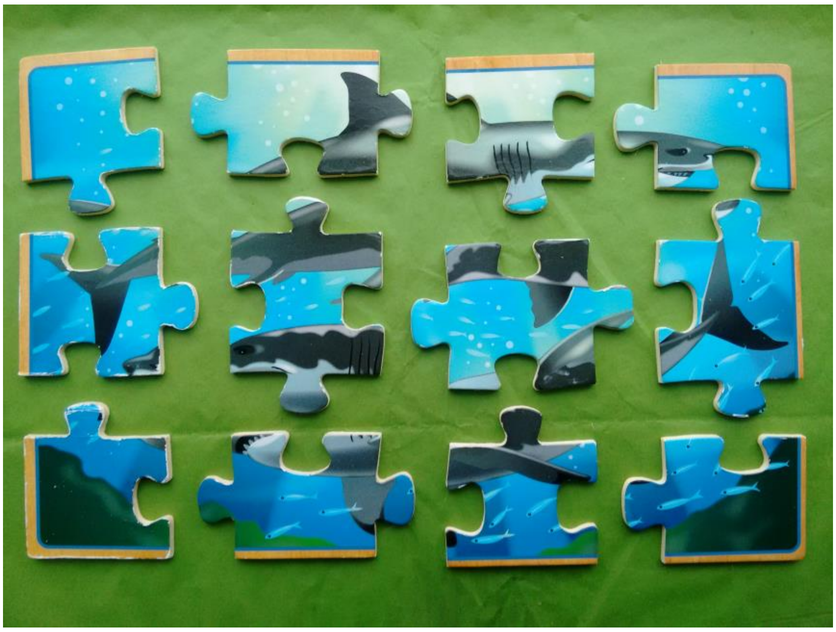


Figure 25: Reference Image of the Puzzle

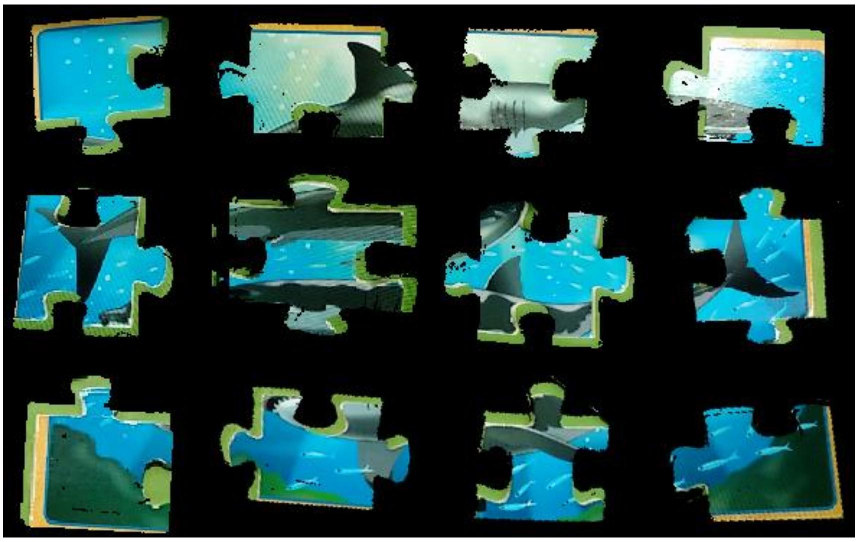


Figure 26: The Final arrangement of the pieces to solve the puzzle.

The ordering of the puzzle pieces that would lead us to solve the puzzle

```
The Original Ordering is:      The solution to the puzzle is:
     1     2     3     4            9     2     5     6
     5     6     7     8            3     4    12    10
     9    10    11    12            7    11     8     1

The Rotation of pieces needed to solve the Puzzle is:
Each value indicates the number of 90 degree rotation clockwise
                  0     2     2     0
                  2     1     2     1
                  0     0     0     2
```

Future Work

As a future extension of the project

1) Back ground colour can be taken as input from the user to overcome the draw back when the Jigsaw pieces have some sections with the same colour as the back ground. So user can choose a background colour and then input the colour during run time.
2) More Robust match find algorithm – Whenever a piece is taken up for match, the algorithm will complete faster if the match is checked at all possible places. The algorithm is currently choosing one location and finding a match only at that location.
3) To develop an algorithm to assemble the puzzle when reference image is not given.
4) To generate the final image by fusing the jigsaw pieces in addition of ordered arrangement of pieces.

References

Text Book

[1] Digital Image Processing, Gonzalez and Woods, Third Edition, Prentice-Hall, 2008.

[2] Fundamentals of Digital Image Processing, Anil K. Jain, Prentice-Hall, 1989

These Text Books have been referred for most of the Image Processing Techniques used in this project. The reference is for the whole concept that has been gained from these books.

Papers

[1] A Jigsaw Puzzle Solving Guide on Mobile Devices, Liang Liang, Zhongkai Liu (URL- https://stacks.stanford.edu/file/druid:rz261ds9725/Liang_Liu_GuidedJigsawPuzzleSolver.pdf

[2] Gallagher, A.C., 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.

Web References

[1a] www. citybizdirectory.com

Article Title: What's the "purpose" of Puzzles? –Educational Benefits

Retrieved from http://citybizdirectory.com/FYI/what_is_the%20_purpose_of_jigsaw_puzzles.htm on 16th May 2015, Time: 9:00

This article gives a brief idea on the benefits of solving puzzles

[2a] www.math.umn.edu

Article Title: Automatic Solution of Jigsaw Puzzles

Retrieved from http://www.math.umn.edu/~olver/vi_/puzzles.pdf on 16th May 2015, Time: 9:30

This article speaks about an approach to solve Jigsaw Pieces

[3a] www.oldpuzzles.com

Article Title: Jigsaw Puzzle Cutting Styles: A New Method of Classification

Retrieved from http://www.oldpuzzles.com/history-techniques-styles/jigsaw-puzzle-cutting-styles-new-method-classification on 16th May 2015, Time: 21:30.

This page give a detail on the characteristics of the Jigsaw Puzzle and explains about various types of Jigsaw Puzzles available

[4a] www.dailyjigsawpuzzles.net

Article Title: Daily Jigsaw Puzzles

Full URL: http://www.dailyjigsawpuzzles.net/puzzle-maker.html.

This website can be used to make jigsaw puzzles with our own images. The Testudo, University of Maryland Jigsaw on the Front Page and at other places in the report has been generated using this website.

--END--

--THANK YOU--