Foundation of Computer Vision
Project : Sudoku Box Detection and Digit Recognition

Abstract

This assignment is about exploring how to perform object segmentation which in this case is a Sudoku on a given sample newspaper clip. The imaging chain required for image processing contains a lot of noise cleaning methods and morphology techniques before performing Hough transform to detect the lines of the outer Sudoku box that will eventually help to segment out the same from the given sample image. After segmenting out the sudoku box, the next step performed is to make the image well suited to perform character recognition by making the image rotate by some angle to bring it upright. Later on, the upright image of the sudoku box is used to segment out each and every digits from the cell and store them in the MATLAB matrix. Future work would be to use this sudoku matrix to actually solve the sudoku puzzle. Thus, this assignment helps us in the path of solving Sudoku just by using image processing.

Overview

This assignment contains different sample of clips of newspapers locating somewhere in it, a Sudoku puzzle. We need to figure out where this Sudoku puzzle is located and then outline it which will eventually help to crop that particular rectangular section of the image for further Sudoku cell detection. The same cropped sudoku might be tilted and thus for character recognition or any kind of template matching, all sudoku images should be brought to a fixed position by using rotation. Thus, using Hough transform we can find the angle of rotation of the sudoku box by finding out the prominent lines representing the sudoku box. After performing rotation, the upright sudoku image would go through same object segmentation techniques to determine 81 square cells of the puzzle and eventually extract the digits out of the same cells using optical character recognition search engine provided by the MATLAB.

The image chain for this color segmentation using morphology and noise removal contains some fixed amount of steps which helped me perform the same almost perfectly. I was able to achieve results of each of the given sample images perfectly, but this imaging chain might not work for all possible combinations of erroneous images that might fail this imaging chain process.

The general imaging chain used for the given sample images is as follows –

1. Firstly, read the image and convert it into double format for image processing and performing arithmetic operations with better precision while doing noise removal, morphology or binarizing the image.
2. Remove the unnecessary black background that might create potential problem while blob detection. Crop just the newspaper clip from the given image for further processing.
3. Binarize the cropped image of the newspaper clip using greythresh() and imbinarize() to convert the pixels signifying two values namely, black(0) and white(1). This will convert the Sudoku grid and other objects on the newspaper with black pixels and newspaper background as white pixels.
4. Invert the binarized image to convert objects on newspaper from black pixels to white pixels for white blob detection.
5. Use structuring element function strel() as ‘squares’ and ‘line’ to detect the square and line objects on the newspaper and perform imclose() operation to eliminate all the unnecessary letters and other objects that does not show evidence of a Sudoku object.
6. Finally, use Hough transform to detect Sudoku grid lines which will prominently show on the image at a particular location thereby finding the same location and the angle at which the Sudoku is tilted.
7. Crop the Sudoku location using the blob detection with largest area and then outline the same using visboundaries() function with magenta color.
8. Use bwareafilt() function to get rid of the small minute blobs that appear in some of the sample images popping up touching the boundaries and just get big white Sudoku blob.
9. Display the cropped Sudoku image with magenta box outlining the outer box of the Sudoku.

10. Again perform the morphology using line as the structuring element for the strel() function to obtain the lines representing thee vertical lines of the sudoku grid. These vertical lines will help to find the rotation of the image.
11. Use Hough transform to detect the prominent vertical lines and check whether there exists 10 such lines with similar lengths and angle of rotation.
12. Finally, rotate the original cropped sudoku image with the angle of rotation of one of those prominent lines to bring it in the upright position.
13. Again, binarize the original cropped sudoku image and perform filtering operation to get rid of the noise created while rotating the image.
14. Perform the morphology using square structuring element with strel() function to obtain clear 81 cells that are represented in the sudoku grid. These 81 cells contains the information about the digits required for extraction.
15. Use bwlabel() function to label these cell objects and crop each such cell object from the original cropped upright sudoku image.
16. Perform imclearborder() function to get rid of the noise and make the digit look prominent before performing optical character recognition on the same.
17. Finally, perform optical character recognition on the filtered & cropped cell object image using certain parameters to make the algorithm only expect the outcome to be digits. The parameters used are as follows -

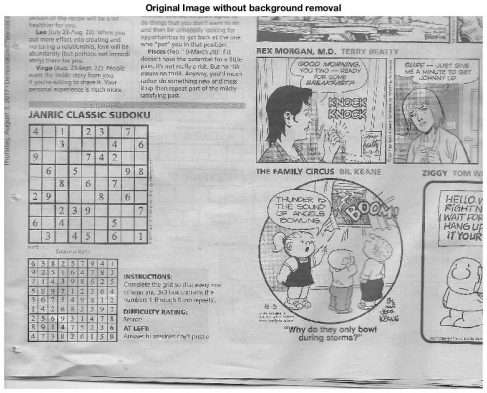
    ocr(cell, 'CharacterSet', '123456789', 'TextLayout', 'Block');

    Here, cell is the actual cell object image, the character set that needs to be recognized are digits in the range 1-9 and the text layout is block making the algorithm to learn that everything should be counted as a single outcome.
18. Outcome of step 17 might include unnecessary erroneous characters. To get rid of such characters, use regular expression match to filter out the same and only obtain a single digit.

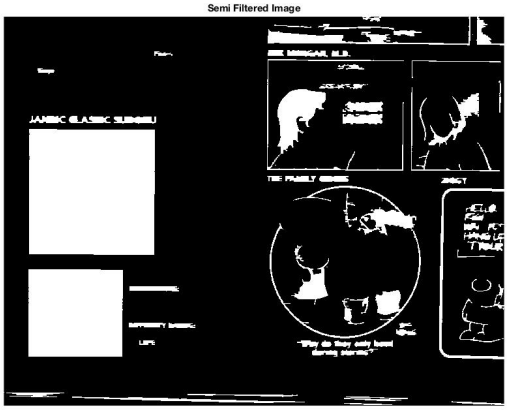
19. Final string obtained might be empty as well representing no digit. Check if the string length is 0 or not. If it is 0, then assume that digit value to be 0. Store each of these digits obtained in a 9X9 matrix in MATLAB.
20. **Future work:** Use these sudoku matrices obtained from different newspaper clips and stored in the MATLAB variable to actually solve the puzzle.

After running the above imaging chain to perform object segmentation on the given sample images, I was able to achieve perfect Sudoku box segmentation from the given newspapers, but this imaging chain might not work for all possible combinations of erroneous images that might fail this imaging chain process. Some sample Sudoku segmentations with the desired images are as follows –

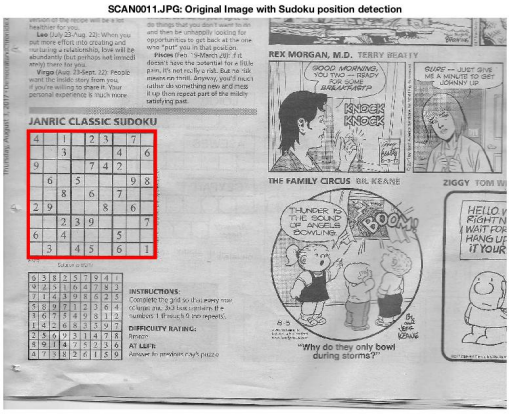
A. Image – ‘SCAN0011.JPG’



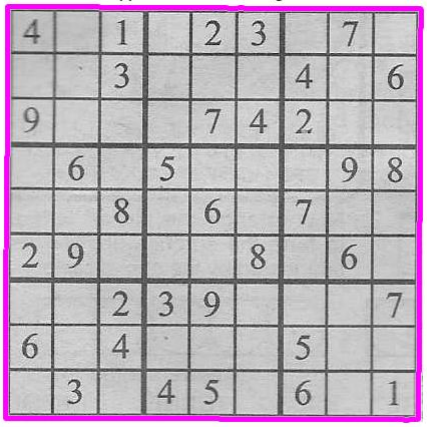
SCAN0011.JPG



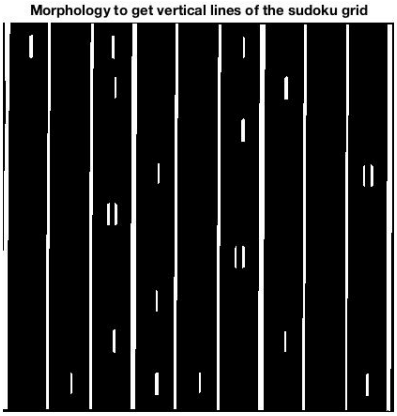
Filtered Image



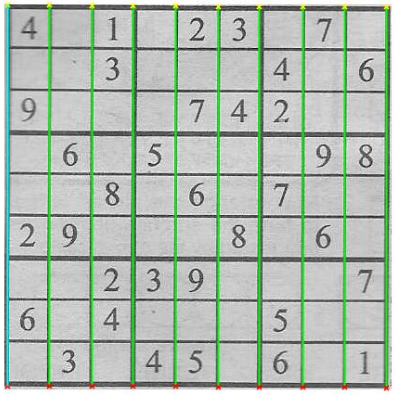
Sudoku location detection using Hough and Morphology



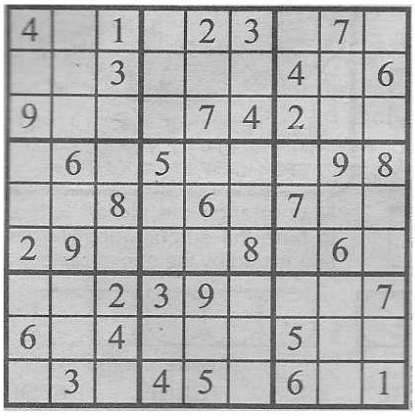
Sudoku Grid Detected



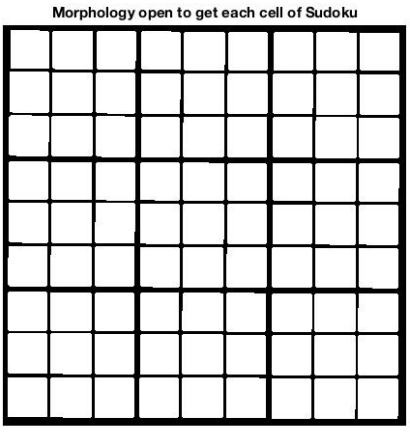
Morphology to get lines



Lines detected



Rotated Sudoku



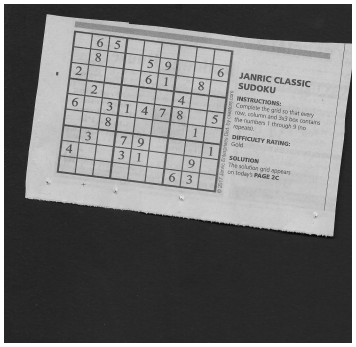
Morphology to get each cells

Output -

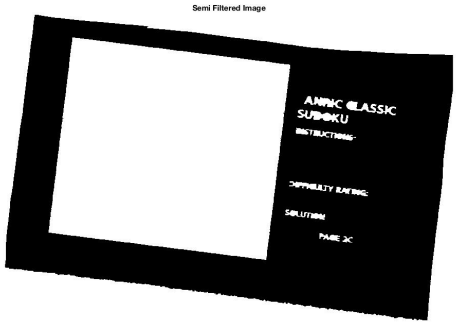
```
The final sudoku from the newspaper clip SCAN0011.JPG --->
sudokuMatrix =

     4     0     1     0     2     3     0     7     0
     0     0     3     0     0     0     4     0     6
     9     0     0     0     7     4     2     0     0
     0     6     0     5     0     0     0     9     8
     0     0     8     0     6     0     7     0     0
     2     9     0     0     0     8     0     6     0
     0     0     2     3     9     0     0     0     7
     6     0     4     0     0     0     5     0     0
     0     3     0     4     5     0     6     0     1
```

SCAN0011.JPG

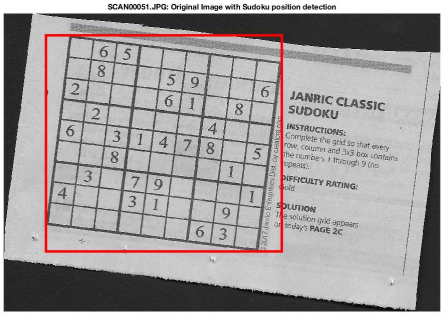
B. Image – ‘SCAN00051.JPG’



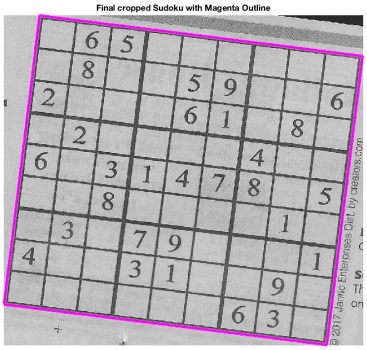
SCAN00051.JPG



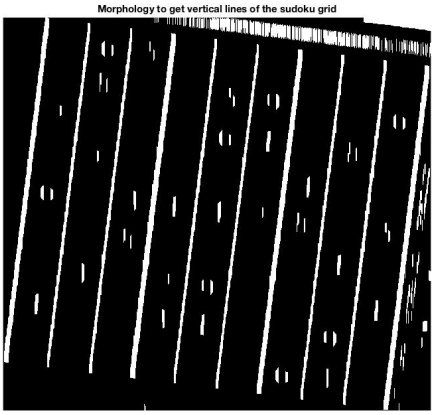
Filtered Image



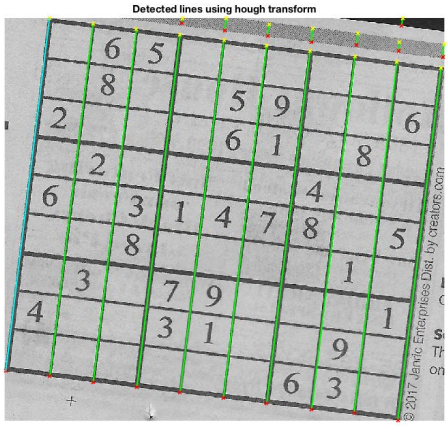
Sudoku location detection using Hough and Morphology



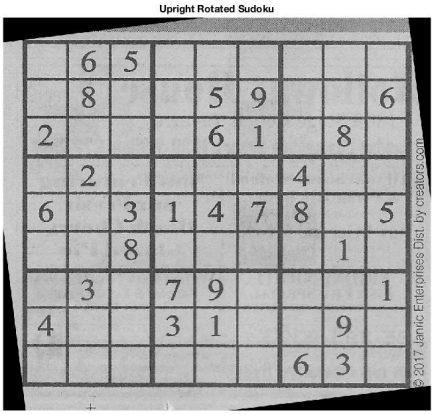
Sudoku Grid Detected



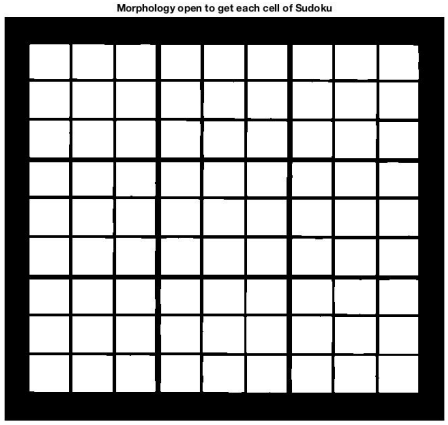
Morphology to get lines



Lines detected



Rotated Sudoku



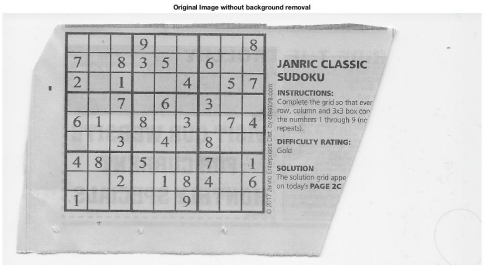
Morphology to get each cells

Output -

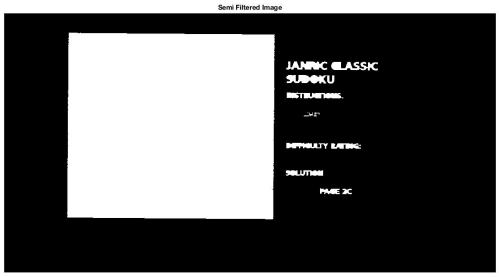
```
The final sudoku from the newspaper clip SCAN00051.JPG --->
sudokuMatrix =

     0     6     5     0     0     0     0     0     0
     0     8     0     0     5     9     0     0     6
     2     0     0     0     6     1     0     8     0
     0     2     0     0     0     0     4     0     0
     6     0     3     1     4     7     8     0     5
     0     0     8     0     0     0     0     1     0
     0     3     0     7     9     0     0     0     1
     4     0     0     3     1     0     0     9     0
     0     0     0     0     0     0     6     3     0
```

SCAN0051.JPG

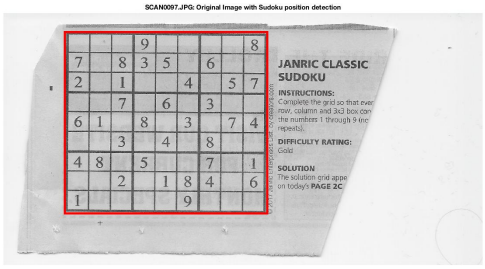
C. Image – ‘SCAN0097.JPG’



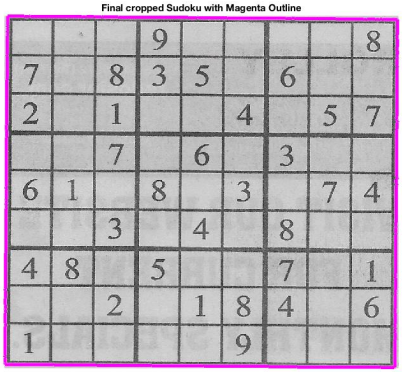
SCAN0097.JPG



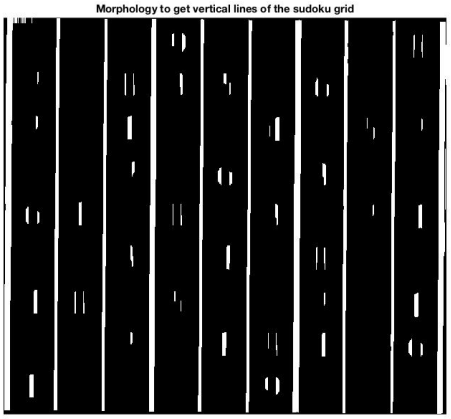
Filtered Image



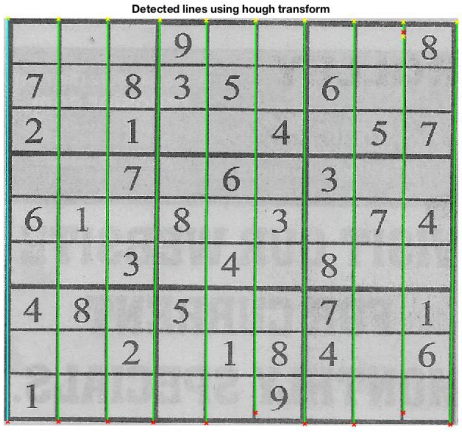
Sudoku location detection using Hough and Morphology



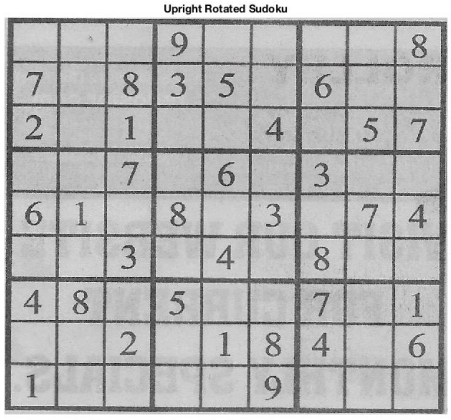
Sudoku Grid Detected



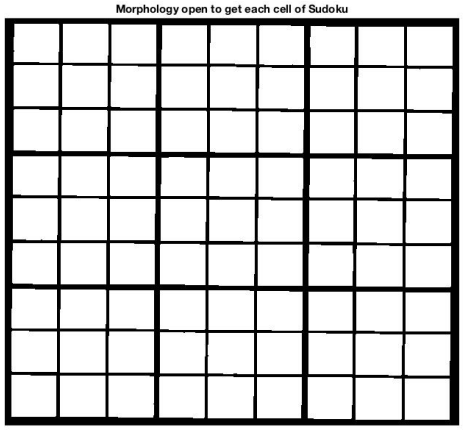
Morphology to get lines



Lines detected



Rotated Sudoku



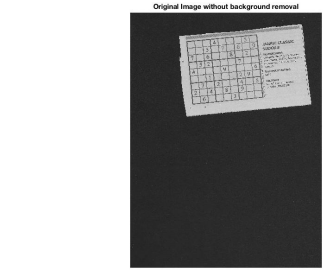
Morphology to get each cells

Output -

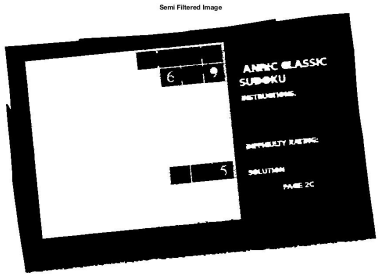
```
The final sudoku from the newspaper clip SCAN0097.JPG --->
sudokuMatrix =

     0     0     0     9     0     0     0     0     8
     7     0     8     3     5     0     6     0     0
     2     0     1     0     0     4     0     5     7
     0     0     7     0     6     0     3     0     0
     6     1     0     8     0     3     0     7     4
     0     0     3     0     4     0     8     0     0
     4     8     0     5     0     0     7     0     1
     0     0     2     0     1     8     4     0     6
     1     0     0     0     0     9     0     0     0
```

SCAN0097.JPG

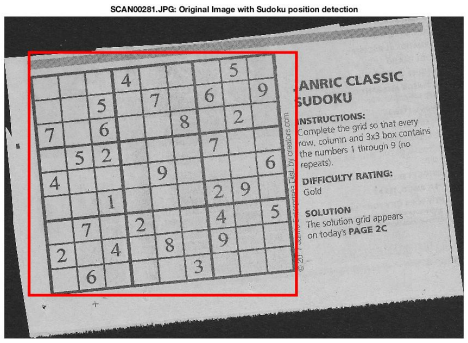
D. Image – ‘SCAN00281.JPG’



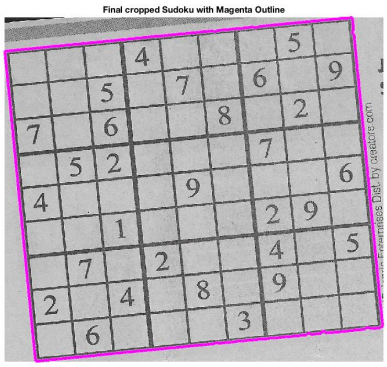
SCAN00281.JPG



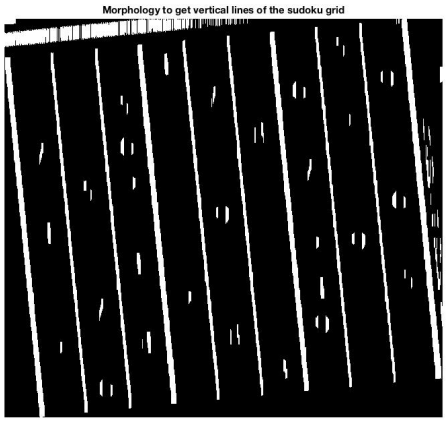
Filtered Image



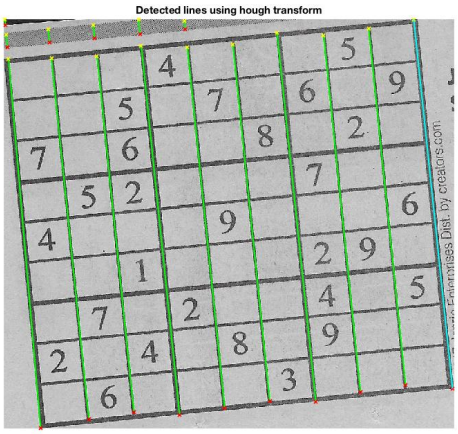
Sudoku location detection using Hough



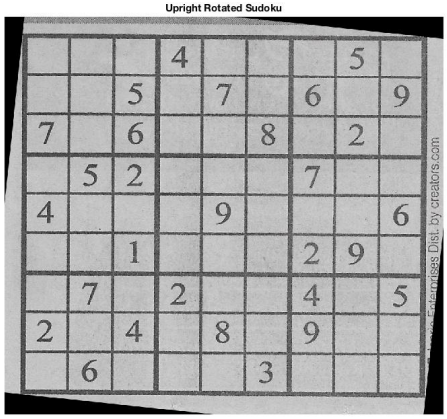
Sudoku Grid Detected



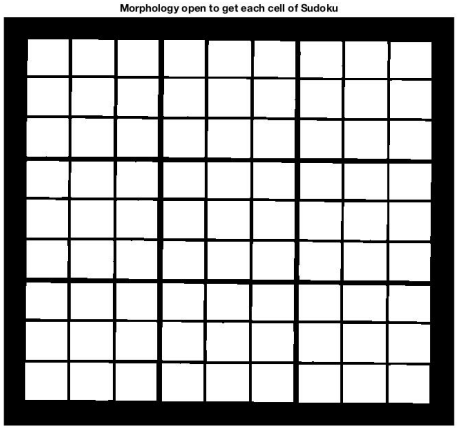
Morphology to get lines



Lines detected



Rotated Sudoku



Morphology to get each cells

Output -

```
The final sudoku from the newspaper clip SCAN00281.JPG --->
sudokuMatrix =

     0     0     0     4     0     0     0     5     0
     0     0     5     0     7     0     6     0     9
     7     0     6     0     0     8     0     2     0
     0     5     2     0     0     0     7     0     0
     4     0     0     0     9     0     0     0     6
     0     0     1     0     0     0     2     9     0
     0     7     0     2     0     0     4     0     5
     2     0     4     0     8     0     9     0     0
     0     6     0     0     0     3     0     0     0
```

SCAN00281.JPG

The above imaging chain might not work for all possible cases of the sudoku detection. For example, if there is a scribbling on the top of sudoku grid, then that might cause problems while going through the imaging chain. Also, if the image is rotated at more than 45 degree in any direction, then the rotation used in this algorithm might rotate the image to an angle which might not lead to upright position. Such cases needs further processing to be performed by actually using the digits inside the cells to get the actual tilt of the sudoku.

Conclusion – From this assignment, I learned how to perform object segmentation which in this case is a Sudoku on a given sample newspaper clip. The imaging chain required for image processing contains a lot of noise cleaning methods and morphology techniques before performing Hough transform to detect the lines of the outer Sudoku box that will eventually help to segment out the same from the given sample image. After segmenting out the sudoku box, the next step performed is to make the image well suited to perform character recognition by making the image rotate by some angle to bring it upright. Later on, the upright image of the sudoku box is used to segment out each and every digits from the cell and store them in the MATLAB matrix. Future work would be to use this sudoku matrix to actually solve the sudoku puzzle. Thus, this assignment helped us in the path of solving Sudoku just by using image processing.

Credits-[1] Slides provided by professor in MyCourses, Images provided by professor in MyCourses