Puzzle Solving Process

Yaseen Haffejee
Computer Science
University of the Witwatersrand
Johannesburg, South Africa
1827555@students.wits.ac.za

***Index Terms*—Computer Vision, Image Segmentation, Image Processing,Puzzle Solver**

I. Introduction

All the labs within the course focused on sub-tasks that were combined in order to finally build the puzzle in lab 7. However, during this process we found that certain techniques performed better at certain tasks. Since solving the puzzle is a multi task process, a pipeline needs to be constructed to ensure the puzzle building process as efficient as possible. In the rest of this paper, a general review is given of all the processes. Thereafter a pipeline is recommended and the drawbacks and advantages of the pipeline are discussed.

II. Overview of processes

The first step in the process of solving the puzzle was image segmentation.

A. Image Segmentation

Image segmentation is the task of separating the image into foreground and background pixels which ultimately produces a binary mask. Within the binary mask, the puzzle piece takes on the intensity 1, giving it a white colour, whilst the background takes on an intensity of 0 which makes it black. In lab 1, we built the first background classifier. The feature sets utilised to build this classifier were the Prewitt and Laplacian filters as well as the HSV pixels. Given these features as well as the RGB pixels of the images, the background classifier was built by modelling the background pixels only utilising a multivariate normal distribution. We trained this model and found that the model performed relatively well on the training data, however failed to generalise well on unseen data. Another issue was that the threshold utilised in order to segment the image and produce the final binary mask was extremely small. This made it extremely difficult to find an optimal threshold. In order to improve on the results, we kept the modelling procedure the same in Lab 2, however we utilised more sophisticated features. In Lab 2, we added the Gaussian, Laplacian of Gaussian, Difference of Gaussian, MR8 filter bank, Local Binary Patterns, Haar filters and textons. The addition of these sophisticated features increased the accuracy of the model on both seen and unseen data. However, the accuracy on unseen data was still not good enough to utilise in a real-world application. Also, the issue of the extremely small threshold was not resolved by the additional features, since this is due to the model we are using. In Lab 3, we utilised the MR8 feature bank as well as the HSV pixels then performed PCA on these features to extract the features used to train the classifier. We also utilised a foreground classifier in this lab. Consequently, we were able to normalize the likelihood of a point being a background pixel, which gave us probabilities between 0-1 and ultimately ensured we could utilise a reasonable threshold of 0.5. Thus the small threshold issue was fixed. The model also performed better on unseen data since we utilised PCA to find the most important features and removed any redundant features. We now had a viable segmentation model and moved onto the next step of corner detection.

B. Edges, Corners and Descriptors

In Lab 4, we focused on the Canny Edge Detector, Harris Corner Detector and the Histogram of Oriented Gradients.The purpose of utilising these techniques was to detect the corners and edges of the puzzle piece which formed vital in detecting the general shape of the model. We utilised these algorithms to detect the interest points used to fit the shape models in Section II-C.

C. Shape Models and Building the Puzzle

In Lab 6, we focused on utilising shape models in order to extract the general shape of the puzzle pieces from the masks of the puzzle pieces. The extraction of the shape of the puzzles enabled us to match shape models and find the pieces in the puzzle that did not match and thus could not be fitted together. In doing so, we could reduce the search space to the pieces that could possibly fit together and ultimately join them using a Breadth-First Search in Lab 7. In order to fit the puzzle pieces they had to undergo Affine transformations to allow us to find which pieces matched perfectly and consequently insert them into the puzzle piece. After combining all these processes, we managed to build the puzzle successfully in lab 7.

III. Recommended Pipeline

The pipeline can be broken down into the following steps:

1) Image Segmentation
2) Corner and Edge extraction
3) Shape model extraction
4) Matching shape models to build the puzzle

A. Image Segmentation

For the process of Image Segmentation, I would recommend utilising Gaussian Mixture Models in order to segment the image. They enable us to perform segmentation extremely well on smaller datasets. However, one of the drawbacks of the GMM is the requirement for handcrafted features in order to achieve optimal performance. As the dataset scales, it becomes more compute heavy to compute these features and it can become time consuming to find a feature set to improve the models performance. A possible solution to this is to utilise deep learning techniques to perform the segmentation as the datasets grow. Deep learning techniques are extremely data hungry, thus when training samples are not an issue, Deep Learning would be the best method to adopt since they learn the feature representations themselves. One caveat of Deep Learning techniques are the requirements for labels, which are expensive and not always accurate. Further research can be done to find Unsupervised segmentation models which will resolve this issue.

B. Corner Extraction

In order to detect the corners, we can utilise the Harris Corner Detector. In the case of the puzzle we solved, the algorithm proved to be efficient in detecting the edges of the puzzle pieces. However,the main caveat is that since the algorithm utilises the eigenvalues in order to determine a corner, the Harris Corner Detector is not scale invariant. In order to combat this we can utilise the SIFT feature descriptor.

C. Shape Model Extraction

In order to extract the shape models correctly, we can utilise the same approach used in Lab 6. This included extracting the contours of the puzzle pieces, then utilising the contours to extract the sides and normalise the extracted sides. Once we obtained this information , in Lab 7 we simply matched pieces using 2 pieces of information: 1) Flat sides cannot be matched with any other sides, 2) A sunken side can only be matched with a protruding side and vice versa. Even though this enabled us to reduce the search space, if we enforced the other 2 rules: 3) A piece cannot match with itself and 4)The side clockwise of the flat side of an edge piece can only be matched to the side anti-clockwise from the flat side of another edge piece (or the sides from a corner) and vice-versa. Similarly,the side opposite the flat side of an edge piece can only be matched to sides from interior pieces. Implementing these two additional constraints will further reduce our search space but also comes with an additional computation. Therefore, we need to consider the trade-off in compute between building the puzzle with the reduced search space and incurring the cost of the algorithm which reduces the search space.

D. Matching Shape Models

We can maintain the process used in Lab 7 which warps the edge pieces and fits the puzzle together utilising the a Breadth First Search. It enables us to match the pieces relatively fast due to the reduced search space. The only caveat is that since we have to consider all the possible pieces in the reduced space, for larger puzzles this might take longer.

IV. Conclusion

The general process of the pipeline is similar to that of the Labs. Except we recommend to utilise more robust methods which can be upscaled to larger puzzles and produce more robust models.