MMU®
MULTIMEDIA UNIVERSITY

TDS3651 VISUAL INFORMATION PROCESSING: TRIMESTER 2130

Name	MUHAMMAD WAIEE BIN ZAINOL
Student ID	1191103225
Lecture/Tutorial	TC1L/TT1L
Email	1191103225@student.mmu.edu.my

Table Of Contents

1. Abstract

An algorithm has been successfully designed to segment lesions from healthy skin regions using the 50 dermoscopic images along with ground truth given. Lesion area has been visualised in White colour, while healthy skin areas represented in Black.

All methods involved in this algorithm are Smoothing, Thresholding, Morphology, Edge Detection, Contour Detection, and Image Normalisation. These methods have been used to ensure this algorithm gives the best segmentation result.

For the overall result, this algorithm has achieved Adapted Rand Error 9%, Precision 95%, Recall 89%, and IoU 85%. The output images have been segmented correctly where it detects the lesion area well even though some details are not captured well especially the edges and shape of lesion area.

2. Introduction

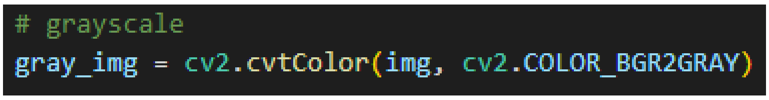
This assignment required to input the image as a 3D numpy array of row*col*3 in BGR format. Then, the output must be a 2D numpy array segmented image with the respective intensities. The task given is to design an algorithm in segmentImage.py to segment lesions from healthy skin regions. There are two datasets given, main dataset and additional dataset that is more challenging. Both dataset consists of 50 dermoscopic images along with the corresponding ground truth. For the segmented images, lesion area is represented in White (1) while healthy skin represented in Black (1) to visualise it properly.

Skin lesion segmentation is a crucial step in dermatological image analysis, with potential applications in early detection of skin diseases and computer-aided diagnosis. Automating this process can significantly aid dermatologists in their diagnostic workflow and improve the efficiency of skin lesion analysis.

The scope of this program extends to clinical practice, telemedicine and ongoing research in the field of dermatology. Automated tissue segmentation seamlessly integrates into clinical workflows to assist dermatologists in their diagnostic evaluations. In remote or small areas, this technology can support mobile health apps, which can monitor skin conditions.

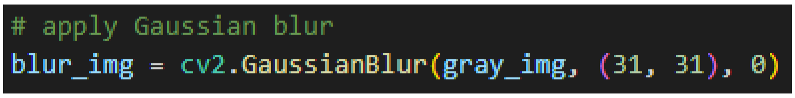
3. Description of Method

The “segmentImage” function utilises a step-by-step process of image processing techniques to achieve the final segmentation results for identifying skin lesions.



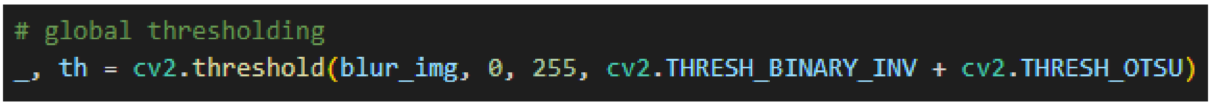
```
# grayscale
gray_img = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
```

The first step in the provided code above is the conversion of the input colour image (img) into a grayscale image (gray_img). This grayscale representation simplifies subsequent processing by condensing pixel intensities into a single channel, making it computationally efficient and often sufficient for medical imaging tasks where colour information may not be crucial.



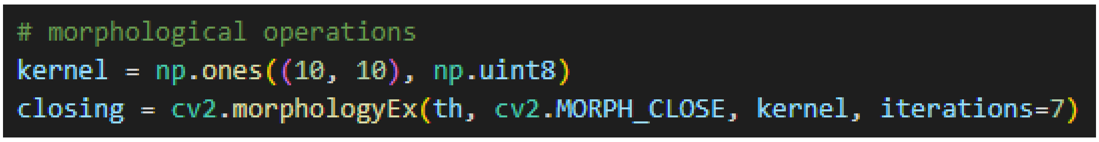
```
# apply Gaussian blur
blur_img = cv2.GaussianBlur(gray_img, (31, 31), 0)
```

Following the grayscale conversion, a Gaussian blur is applied to the grayscale image (gray_img). This blurring operation serves the purpose of smoothing the image by reducing high-frequency noise and fine details. This is particularly important for subsequent thresholding operations and edge detection, as it helps eliminate small variations in pixel values that may not be relevant for the task at hand.



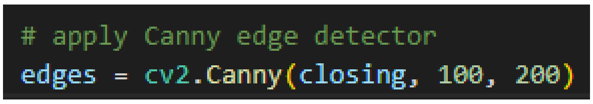
```
# global thresholding
_, th = cv2.threshold(blur_img, 0, 255, cv2.THRESH_BINARY_INV + cv2.THRESH_OTSU)
```

The next step involves global thresholding using Otsu's method on the blurred image (blur_img). This thresholding operation binarizes the image, classifying pixels into foreground or background. Otsu's method dynamically determines an optimal threshold based on the image's histogram, adapting to variations in pixel intensities. This adaptability is crucial for segmenting medical images with varying levels of contrast.



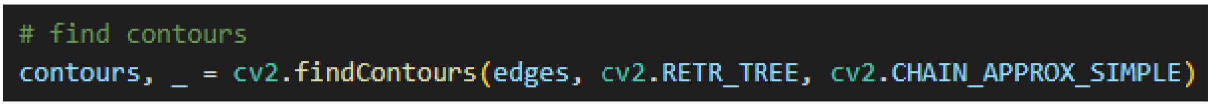
```
# morphological operations
kernel = np.ones((10, 10), np.uint8)
closing = cv2.morphologyEx(th, cv2.MORPH_CLOSE, kernel, iterations=7)
```

To further refine the binary image, morphological closing is applied using a square-shaped kernel. Closing is a morphological operation that helps fill small gaps and connect nearby contours in the binary image. This ensures that segmented regions are more cohesive and free from small interruptions, contributing to improved accuracy in subsequent contour detection.



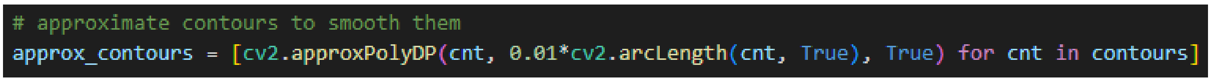
```
# apply Canny edge detector
edges = cv2.Canny(closing, 100, 200)
```

The Canny edge detector is then employed on the closed binary image (closing). This edge detection technique identifies edges in the image, which can be critical for medical imaging where edges often represent boundaries of structures or features. Detecting these edges is a fundamental step in the subsequent contour analysis.



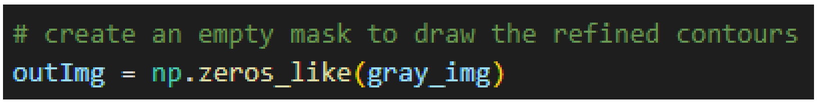
```
# find contours
contours, _ = cv2.findContours(edges, cv2.RETR_TREE, cv2.CHAIN_APPROX_SIMPLE)
```

Contours are detected in the edge-detected image (edges) using the cv2.findContours function. Contours represent the boundaries of objects or regions, and their detection is a key step in identifying and analysing specific structures within the medical image. These contours are later sorted based on their area, prioritising larger regions that may be more significant for the task.



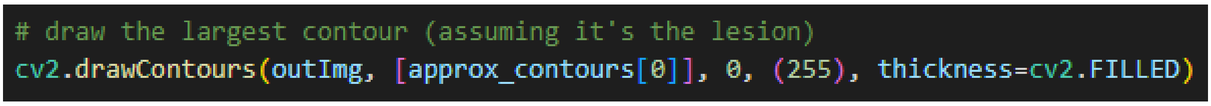
```
# approximate contours to smooth them
approx_contours = [cv2.approxPolyDP(cnt, 0.01*cv2.arcLength(cnt, True), True) for cnt in contours]
```

The next step involves contour approximation, which reduces the number of vertices while retaining the overall shape of the contours. This simplification is beneficial for subsequent analysis, making the contours smoother and more manageable in terms of data points.



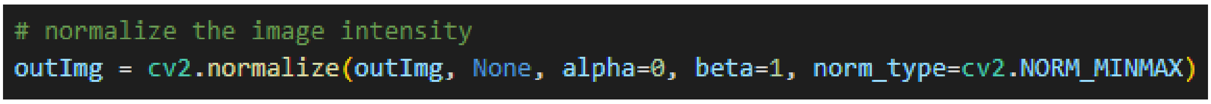
```
# create an empty mask to draw the refined contours
outImg = np.zeros_like(gray_img)
```

To visualise the refined contours, an empty black image (outImg) with the same dimensions as the grayscale image is created. This image serves as a canvas for drawing the contours and providing a clear visual representation of the identified regions of interest.



```
# draw the largest contour (assuming it's the lesion)
cv2.drawContours(outImg, [approx_contours[0]], 0, (255), thickness=cv2.FILLED)
```

The code then proceeds to draw the largest approximated contour on the empty image (outImg). This assumes that the largest contour corresponds to the most significant object or region in the image, such as a lesion in medical images.



```
# normalize the image intensity
outImg = cv2.normalize(outImg, None, alpha=0, beta=1, norm_type=cv2.NORM_MINMAX)
```

Finally, the intensity values of the resulting image (outImg) are normalised to the range [0, 1]. This normalisation step ensures consistency in visualisation or further processing, making the image suitable for display or input to algorithms that require standardised intensity ranges.

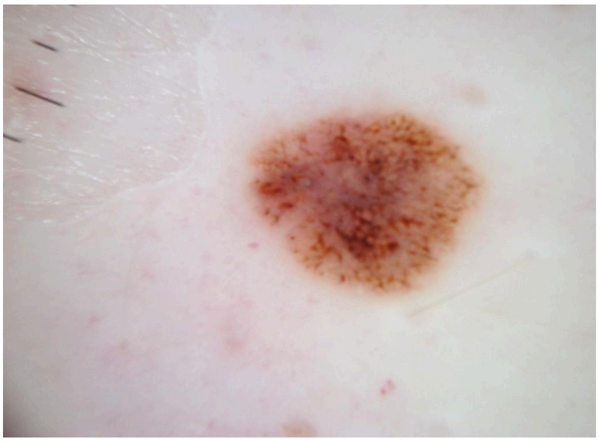


Figure 1: Skin Lesion Images

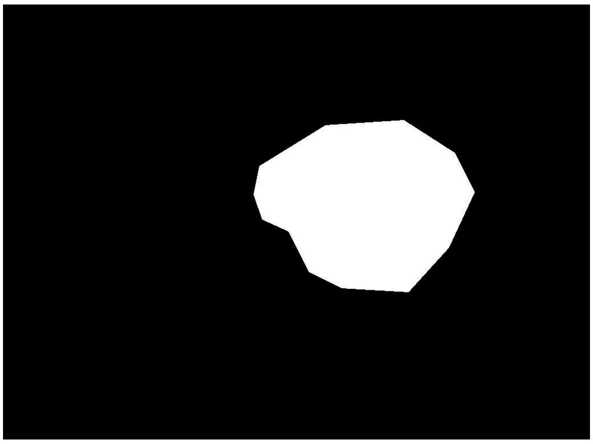


Figure 2: Segmented Skin Lesion Images

4. Result and Analysis

4.1. Main Dataset

Metrics	Scores
Adapted Rand Error	9%
Precision	95%
Recall	89%
IoU	85%

Figure 3: Main Dataset Result

The Adapted Rand Error, a measure of the dissimilarity between ground truth and predicted segmentations, was observed to be at 9%. This is probably due to the small details at the edge and shape of the segmented image are not captured well. Thus, there is some dissimilarity between ground truth and segmented images.

Precision, a metric that quantifies the accuracy of positive predictions, was observed to be at 95%. This indicates that the algorithm effectively identified and segmented true positive instances of skin lesions. The possible factor would be because the lesion area has been segmented well even though there are some missing details.

The Recall, measuring the algorithm's ability to capture all positive instances, was recorded at 89%. While slightly lower than precision, a recall score above 80% suggests that the algorithm successfully identified a significant portion of actual lesion areas. The choice of parameters and type of Morphology operation applied might affect the algorithm's ability to capture all lesion instances and get higher scores.

The Intersection over Union (IoU) score represents the overlap between the predicted and actual lesion areas and it was measured at 85%. This score shows a good overlap between predicted and ground truth regions since it is above 80%. The IoU score is directly influenced by the balance between precision and recall. If the algorithm achieves a good balance, capturing true positive lesions while minimising false positives and false negatives, the IoU score is likely to be higher.

Analysis of these scores suggests that the algorithm performs well in terms of precision, ensuring a high accuracy of identified lesions in this main dataset. The slightly lower recall score might indicate that there is potential for improvement in capturing all lesion instances. Factors influencing these scores could include variations in lesion types, image quality, and the diversity of the dataset.

4.2. Additional Dataset

Metrics	Scores
Adapted Rand Error	36%
Precision	76%
Recall	69%
IoU	55%

Figure 4: Additional Dataset Result

When compared to the primary dataset, the results for the extra dataset show some differences. This is the analysis done using the data that was given:

With the additional dataset, the Adapted Rand Error is greater at 36%. This indicates that, in comparison to the main dataset, there may be a greater difference between the ground truth and the anticipated segmentations. Potential limitations in the segmentation performance could be indicated by the higher error, which could be attributable to difficulties capturing fine details at the edges and forms of the segmented images.

The observed precision for the additional dataset is 76%. Although this indicates a fairly high accuracy of positive predictions, it is slightly less accurate than what can be observed in the primary dataset. The precision of the algorithm's identification and segmentation of true positive occurrences of skin lesions appears to be sufficient, however there might be some missing features.

Moreover, the recall score is 69%. A recall score above 60% indicates that the algorithm detected a significant portion of actual lesion spots, although significantly less than the precision score. The slightly decreased recall, however, may indicate that there is still potential for improvement in terms of fully capturing lesion instances in this dataset.

Next, the Intersection over Union (IoU) score is 55%. The IoU score is lower when compared to that of the main dataset, yet it still shows a certain amount of overlap between the predicted and ground truth regions. This points to a possible discrepancy between recall and precision that may be greater.

In summary, the analysis of the additional dataset reveals some challenges in segmentation performance, as evidenced by higher Adapted Rand Error and lower IoU scores. Precision remains relatively high, indicating accuracy in positive predictions, but there is room for improvement in recall. Factors influencing these scores may include variations in lesion types in this additional dataset, noises, image quality, and the unique characteristics of the additional dataset.

4.3. Output Comparison

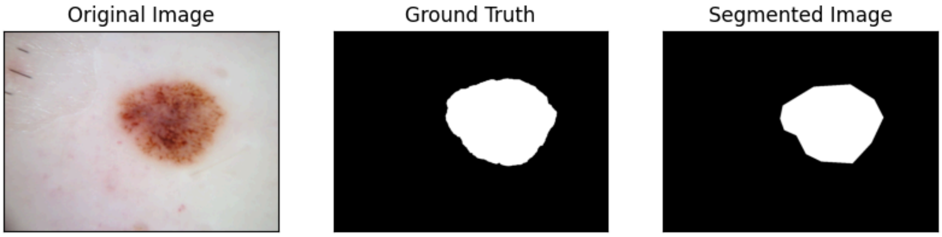


Figure 5: Comparison of results in main dataset

Figure 4 shows the results compared with ground truth images. Based on the result above, we can observe that the lesion area has been segmented well, and the noise has been removed successfully. However, the edge details are not captured well. The shape of the segmented image also is not 100% the same as ground truth, since the details are not successfully captured.

5. Suggestion for Improvement

In order to **improve this algorithm on additional datasets**, it is suggested to remove the Canny Edge Detector from the algorithm. To prove this suggestion, the algorithm has been tested with and without Canny Edge Detector. The results shown below in Figure 6.

Metrics	**With Canny Edge Detector**	**Without Canny Edge Detector**
Adapted Rand Error	36%	24%
Precision	76%	79%
Recall	69%	77%
IoU	55%	66%

Figure 6: Comparison of algorithm results for Canny Edge Detector

Based on the figure above, we can observe that the score is slightly better for the additional dataset in terms of Adapted Rand Error, Precision, Recall, and IoU when we remove the Canny Edge Detector. The Canny edge detector is sensitive to changes in intensity, identifying edges in the image. In certain cases, it might detect edges that are not part of the actual lesion boundaries, leading to false positives. By removing the Canny edge detector, it can help to restrict the inclusion of false positives produced by edge detection and the algorithm may now focus more on the internal details of the lesions rather than relying on potentially noisy edge information.

Furthermore, as we observed earlier in the main dataset, the edge and shape details of the segmented images are not captured well, it also might be affected by Morphology operations, Canny Edge Detector, and Contour detection. To improve this algorithm, we can adapt the size and shape of the structuring elements in Morphology operations, Canny Edge detector, Contour detection based on the dataset characteristics that can enhance the algorithm's performance. This fine-tuning may involve experimenting with different kernel sizes and types to improve the connectivity of segmented regions. Other than that, we can use different approaches to ensure we obtain the best result, such as another method of morphological operation, and edge or boundary detection.

References

1. ChatGPT: https://chat.openai.com
2. CV2 documentation: https://docs.opencv.org/3.4/d6/d00/tutorial_py_root.html
3. Stack Overflow: https://stackoverflow.com

Responsible Collaborators

Below is the list of students I discussed with during the assignment:

1. Lovesh Anand A/L A.Kumereshwaran

I acknowledge that this assignment was done by myself with peers discussion and reference to the lab exercise.