Computer Vision extended assignment

Task 1:

Aim: we are provided with MRI data and our aim for task one is to Develop and apply different segmentation algorithms, based on any technique you have learnt to each slice of the MRI data.

Method:

Initially, we have created a function, namely 'segmentation' which is a simplistic approach to segmenting the image which gave us an idea that the segmentation would be much more complicated than what we had anticipated, in this function, we first we applied 'mat2gray' function that converts the image to a grayscale image, then we passed that image through multi-thresholding function to segment gray image into different regions. Later on, we quantize the image and label rgb to it using label2rgb and then convert it to gray once again.

Then to evaluate the accuracy for my function, we use dice score, which unfortunately turned out to be less then what we had expected (0.74). Therefore we leave this method, and approach it differently.

The second method we used for segmentation is mixture of our knowledge gained from labs, lecture videos and online research.

In this, we identify aspects of the image in histogram with different intensities which we can get from histogram), later we worked on separating different aspects of the image such as skin, skull, grey matter, air, gray matter and white matter., so that we get their separate values in multiple different variables.


fig 1,2,3 & 4: skull ,skin, gray and white matter


Towards the end, we combined the all the variables consisting of different aspects of the image to form a segmented and combined image

Result: the result of our final approach provided us with an image that comes to the expected output, we later confirm it in task 2.

fig 5: segmented and combined image

Conclusion: this task gave us an in-depth understanding and experience of applying multiple approaches to image segmentation based on the knowledge gathered.

Task 2:

Aim: after applying multiple approaches to MRI data, we then have to compare your segmented results for each algorithm to the ground-truth label provided. Justify and explain the metric used to assess accuracy.

Method: conventionally, the methods used for evaluation of similarity are jaccard and dice scores, in our case we used dice similarity to assess the accuracy of the image because dice method provides us with how many positives you find, but it also penalizes for the false positives that the method finds.

$$\text{Dice score} = \frac{2 \cdot \text{number of true positives}}{2 \cdot \text{number of true positives} + \text{number of false positives} + \text{number of false negatives}}$$

Result: the dice score gave an accuracy of 0.98

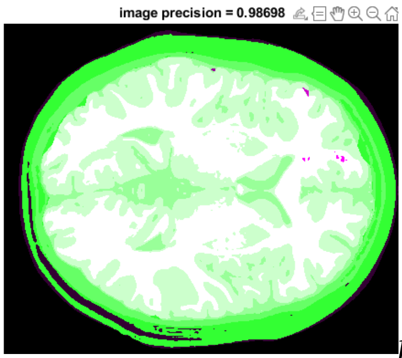


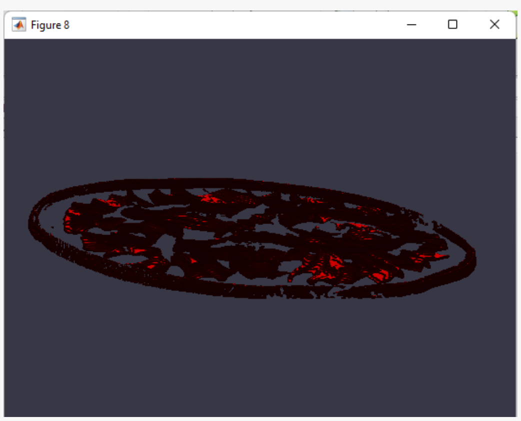
fig5: image precision for task 1

Conclusion: this task allowed us with the option of choosing between various approaches of evaluating precision of our algorithms, in our case we chose dice that helped us gain the accuracy of our image.

Task 3:

Aim: In task three we have to think about and implement a 3D image segmentation algorithm that can be applied to all slices, simultaneously. Discuss why the use of the proposed algorithm is much more reasonable than task 1 and same evaluation method is applied as that of task 2.

Method: In this case, we have used `imsegkmeans3 function for segmentation and then used dice function to evaluate the accuracy.`



`fig2: output for task 3`

Result: `The accuracy for the algorithm is 0.77.`

Conclusion: having imsegkmeans3 made it simpler to implement 3D segmentation, and the evaluation method (dice score) helped us gain accuracy of the proposed algorithm, `Conclusion:`

Supplementary data

Code for task 3:-

```
slices = load('Brain.mat');
label = slices.label;
T1 = slices.T1;
L = imsegkmeans3(T1,6, MaxIterations=100);
volshow(L==2);

Mtx=zeros(size(L1));
Mtx(L==1) =0;
Mtx(L==2) =1;
Mtx(L==3) =2;
Mtx(L==4) =3;
Mtx(L==5) =4;

label_new = cast(label, "logical");
final_image = cast(Mtx,"logical");
similarity_dice_new = dice(final_image, label_new);
figure();
title(['image precision = ' num2str(similarity_dice_new)]);
```

Code for task 2:-

```
label_new = cast(label(:,:,9), "logical");
final_image = cast(final_image,"logical");
similarity_dice_new = dice(final_image, label_new);
figure();
imshowpair(final_image, label(:,:,1));
title(['image precision = ' num2str(similarity_dice_new)]);
```

Task 1 code:-

```
load("Brain.mat") %loading brain file
%calling the segmentation function
[res,score,final] = Segmentation(T1(:,:,9),label(:,:,9));
imshowpair(res,label(:,:,1),"montage");
title(score);

% creating a Histogram for identify different intensities
imhist(final);
figure;


%% Skull + Grey with Skull
final_one = final >= 0.7 & final <= 0.8;
imshow(final_one);
mask = zeros(size(final_one));
mask(45:end-45, 40:end-40)=1;
binary_weighted=activecontour(final_one, mask,300);
binary_weighted = imcomplement(binary_weighted);
binary_weighted = cast(binary_weighted,"double");
skullvar = segmentImageOne(bw); %calling the segmentImageOne function
imshow(final_one);
imshow(skullvar);
imshow(final_one - skullvar); %separating skull from brain
```

```
binary_weighted = cast(binary_weighted,"double");
skullvar = segmentImageOne(bw); %calling the segmentImageOne function
imshow(final_one);
imshow(skullvar);
imshow(final_one - skullvar); %separating skull from brain
skull= final_one - skullvar;
skull_1 = zeros(size(skull));
skull_1(skull>0)=51;
imshow(skull_1);
title("Skull");


%% Skin Label 1
final_two = final_one - skullvar;
imshow(final_two);
mask_2 = zeros(size(final));
mask_2(77:end-77, 88:end-88)=1;
binary_weighted_2=activecontour(final, mask_2, 300);
binary_weighted_2 = imcomplement(binary_weighted_2);
binary_weighted_2 = cast(binary_weighted_2,"double");
skinvar = segmentImageOne(bw_2);
imshow(skinvar);
imshow(final_two - skinvar);
```

```
imshow(final_two - skinvar);
skin = final_two - skinvar;
skin = skin > 0;
final_1 = zeros(size(skin));
final_1(skin>0)=153;
imshow(final_1);
title("Skin New 153 values");


%% CSF Label 3
testvar4 = testvar >= 0.0 & testvar <= 0.1;
imshow(testvar4);
mask_3 = zeros(size(testvar));
mask_3(106:end-106, 114:end-114)=1;
binary_weighted_3=activecontour(testvar, mask_3, 300);
binary_weighted_3 = imcomplement(binary_weighted_3);
binary_weighted_3 = cast(binary_weighted_3,"double");
csfvar = segmentImageOne(binary_weighted_3);
imshow(csfvar);
imshow(testvar4 - csfvar);
csfvar_one = testvar4 - csfvar;
csfvar_one = csfvar_one > 0;
final_2 = zeros(size(skin));
```

```
csfvar_one = testvar4 - csfvar;
csfvar_one = csfvar_one > 0;
final_2 = zeros(size(skin));
final_2(skin>0)=255;
imshow(final_2);
title("CSF New 255 values");

%% Grey Matter label 4
greyvar = segmentImageTwo(bw);
greyvar = final_one - skullvar - greyvar;
imshow(greyvar);
mask_4 = zeros(size(greyvar));
mask_4(15:end-15, 10:end-10)=1;
binary_weighted_4=activecontour(greyvar, mask_4,300);
figure();
imshow(mask_4,[]);
title("Skull")
figure();
imshow(binary_weighted_4,[], colormap=jet)
title("Skull Binary")
imshow(binary_weighted_4);
binary_weighted_4 = cast(binary_weighted_4,'double');
skull_1=binary_weighted_4-greyvar;
imshow(skull_1);
```

Main.mlx | test_3d.mlx | finale.mlx | +

```
imshow(binary_weighted_4,[], colormap=jet)
title("Skull Binary")
imshow(binary_weighted_4);
binary_weighted_4 = cast(binary_weighted_4,'double');
skull_1=binary_weighted_4-greyvar;
imshow(skull_1);
skull_1 = imcomplement(skull_1);
skull_1 = skull_1 - csfvar;
imshow(skull_1)
grey_matter = skull_1;
grey_matter = grey_matter > 0;
final_3 = zeros(size(grey_matter));
final_3(grey_matter>0)=102;
imshow(final_3);
title("Grey Matter New 102 values");



%% White Matter Label 5
testvar3 = final >= 0.9 & final <= 1;
imshow(testvar3);
testvar3 = testvar3 - skinvar;
imshow(testvar3);
```

Live Editor - C:\Users\user\Desktop\Main.mlx

Main.mlx | test_3d.mlx | finale.mlx | +

```


%% White Matter Label 5
testvar3 = final >= 0.9 & final <= 1;
imshow(testvar3);
testvar3 = testvar3 - skinvar;
imshow(testvar3);
white_matter = testvar3;
white_matter = white_matter >0;
final_4 = zeros(size(white_matter));
final_4(white_matter>0)=204;
imshow(final_4);
title("White Matter New 204 values");



%% Combining all aspects
final_image = (skin)+(skull)+(csfvar_one)+(grey_matter)+(final_white_m
title("final image");
imshow(final_image);
```

Three functions used for task1, task2 and task3 are given below:-

```
%%
function [result,score,final] = Segmentation(img,label)
    image_gray = mat2gray(img);      %coverting the image to gray
    threshold_image = multithresh(image_gray,3);  %multi-thresholding
    segmented_image = imquantize(image_gray,threshold_image); %quantizing image
    segmented_rgb = label2rgb(segmented_image);
    final = im2gray(segmented_rgb);
    final = mat2gray(final);
    imshow(final); %o/p the final image
    figure;
    %Now, we find the dice score
    label = cast(label, "logical");
    result = cast(segmented_image,"logical");
    score = dice(result, label);
    result = segmented_rgb;
end
%%
```

```
function [BW,maskedImage] = segmentImageOne(X)
% Adjust data to span data range.
X = imadjust(X);

% Create empty mask.
BW = false(size(X,1),size(X,2));

% Flood fill
row = 75;
column = 87;
tolerance = 5.000000e-02;
addedRegion = grayconnected(X, row, column, tolerance);
BW = BW | addedRegion;

% Create masked image.
maskedImage = X;
maskedImage(~BW) = 0;
end
```

```
function [BW,maskedImage] = segmentImageTwo(X)

Xmin = min(X(:)); % Normalizing data to range in [0,1].
Xmax = max(X(:));
if isequal(Xmax,Xmin)
    X = 0*X;
else
    X = (X - Xmin) ./ (Xmax - Xmin);
end

BW = false(size(X,1),size(X,2)); % Creating empty mask

% Flood fill
tolerance = 5.000000e-02;
column = 185;
row = 30;
addedRegion = grayconnected(X, row, column, tolerance);
BW = BW | addedRegion;

% making masked image.
maskedImage = X;
maskedImage(~BW) = 0;
end
```