Assignment 2: Painting Recognition
CS4053: Computer Vision
Sean Durban (14320715)
Due: 12/12/2017

Method

Note: Examples of the method steps will use Gallery 3 image shown below unless otherwise stated.

The first subtask of this assignment was to be able to find the frames and paintings. An assumption was made that the background of the wall the paintings are hung on is a single colour. With this assumption meanshift segmentation is used, the openCV methods pyrMeanShiftFiltering and floodFill in particular. Meanshift is used as it provides clustering based on colour and clustering based on spatial locality. This is especially useful when with our background assumption and the assumption that the frame of a painting will be clearly differentiable from the background. With meanshift with also don't need to a specific number of clusters like k-means segmentation. But it does still have 'magic numbers' in both the difference between pixels in a region and the maximum size of a region to be considered. After some trial and error between the test images, I concluded the best parameters would be a relatively low-mid colour window and high spatial radius. This makes sense since we have to account for many colours in the paintings and we're looking for potentially large regions in the image. Below are examples of first the meanshift filtering and the following output of floodFill:

After meanshift filtering

After floodFill

As seen in the above output image of floodFill it has successfully differentiated the different segments on the wall and has the background of the wall as a single segment. It also picks up other elements of the image such as the floor and the information plaques. FloodFill returns a Rect for each segment. These are further classified in order to remove the unwanted elements mentioned before. When a rect is returned by floodFill of a segment it is first checked to see if it is over a certain size (10*10 pixels), this removes segments which may be noise or inner segments of the painting. It is also checked to ensure the segment isn't the background wall. Below shows an example of all the rects (in red) returned by the floodFill process. Here clearly there's the floor and the bottom of the wall segments which we want to ignore. More classification is done to remove these segments, it checks to see if a rect is connected to an edge of the image. This makes the assumption that segments such as the floor and possibly adjacent walls will be originate or end at a boundary of the image. The next example image shows the remaining rects after this classification

Rects returned from floodFill

After filtering boundary rects

Next there's the inner segments of the paintings which we'd like to merge. First we go through all the rects and merge rects which overlap. This is done by using the & (AND) operation on both the rects and comparing the overlap area to 0. If they overlap then they're merged and added back to the list of rects. Below an example image is shown which shows the rects returned after merged. There's also segments which are obviously not the paintings such as the information plaques and the security camera on the wall. An additional filtering method removes these, it simply removes rects which are below a threshold of height and width which could be valid paintings. As this step is after all the rects have been merged this is a reasonable assumption. The final rects following these filtering steps are shown in an image below.

After merging rects

Final rects

Once the final rects have been determined this method deals with each separately. The image is cropped just to include that segment, an example of this is shown below.

The previous step was to find the painting and frame so next step is to remove the frame. I attempted to use meanshift segmentation again to remove the frames and this was successful in most cases but not all cases., particularly it was unable to remove the frame for painting 2 and 6. I came to the realisation that all but one cropped image (out of 10) still had their full frame and these frames took up similar area of the image. From this realisation I changed my approach to simply zooming the image by 14%, this figure was found through trial and error. This zooming method outperformed the meanshift for the metrics explained next on average. Therefore it was chosen over meanshift.

Once we've determined the rect which we believe to be the painting we crop the image further. Then for each of our template paintings we crop them to the same size of the cropped image. An example of these are shown below.

Before zoom

After zoom

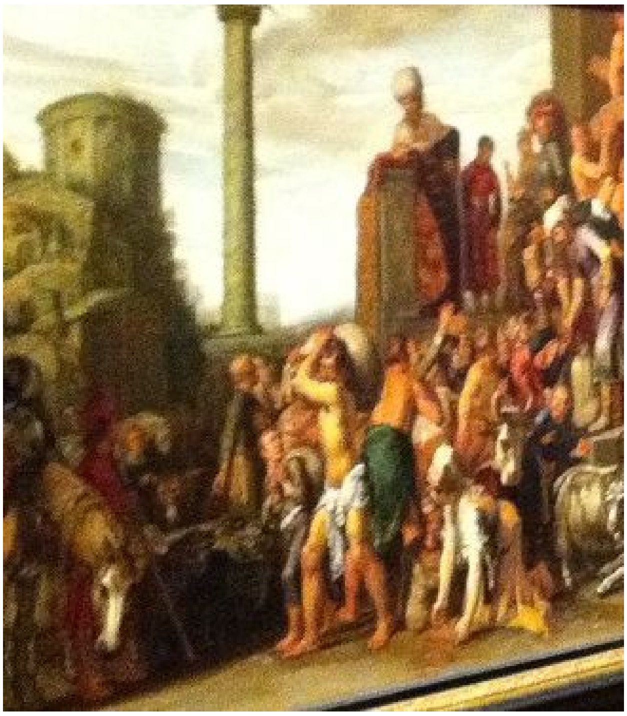
Cropped Image

Cropped Template

The above images are a good example of the issues with direct comparison of paintings. To the human eye we can say that these paintings are the same but they're actually quite different. The brightness of both is the obvious difference and the cropped image still has a few pixels of the golden frame. This made it particularly hard to pick a single comparison strategy that would be successful for all painting recognition. Another issue was paintings were the zoom removes some of the painting which would negatively effect template matching. Therefore I decided to use template matching in conjunction with HSV histogram comparison.

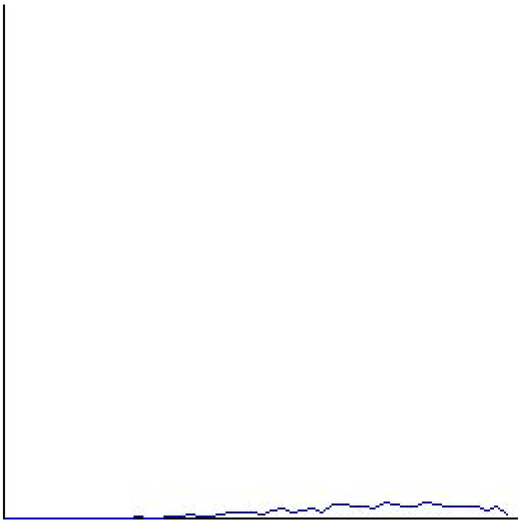
Template matching makes use of greyscale versions of both the image and the template. It essentially slides the template across the image and compares the template position to the relative position in the image. As the cropped image and the template are the same size only a single comparison needs to be completed. I used the normalized correlation coefficient method (CCOEFF_NORMED in openCV) which returns the correlation between the template and the image. This correlation is between [-1,1] with -1 being a perfect mismatch, 0 being no correlation and 1 being perfect match. The max value for this correlation which is extracted from the result object returned by the method.

Histogram comparison in HSV colour space is done to remove the effect of difference in brightness/luminance between the image and the template. As both the image and template are the same size we can compare them directly, they're both converted from RGB to HSV. The histogram only takes into account the hue and saturation channels. The openCV ranges were used hue - [0, 179] and saturation - [0,255]. We allow for 50 hue bins and 60 saturation bins. After the histogram is calculated (using calcHist()) it's normalised to give values of either 0 or 1, this allows us to simple binary comparison. Correlation metric is used again to compare the histograms which gives a value between [0,1] with 1 being a perfect match. Below is the resulting histograms for the shown cropped image and resized template from gallery 2.

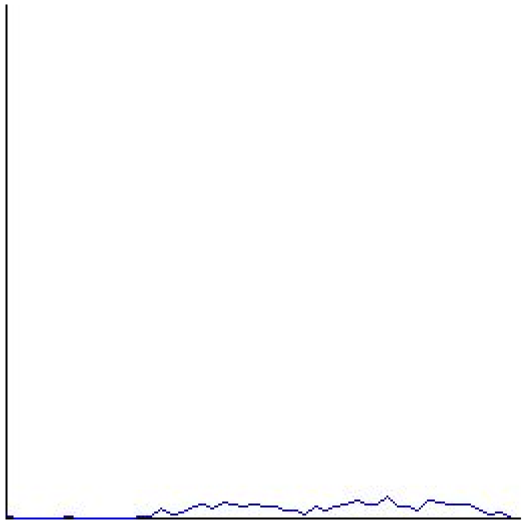


Cropped Image

Resized template



Cropped Image H-S Histogram



Template H-S Histogram

The above histogram comparison results in correlation of 0.86 but template matching correlation of 0.44. Template matching is negatively impacted by the frame and zoom method cropping some of the painting out. But the cropped image still accounts for majority of the pixels in the template allowing the histograms to have strong correlation. In the reverse there's also examples of paintings which perform poorly in histogram comparison but have strong correlation in template matching. In all cases the highest template matching correlation was for the correct painting when the cropped segment was actually a painting. There's a case in gallery 4 where a segment of the floor is not filtered out, also there could be cases where a painting is not in the templates provided. This introduced the need for a threshold.

The max result for template matching correlation is recorded for each cropped image along with the template/painting number and the corresponding histogram comparison result. After computing all the results for this image, the average of the max template matching result and corresponding histogram comparison result is compared to a threshold. If it's above this threshold then the program has successfully recognise a painting and the relevant rect is added to a vector along with the painting. After all the segments have been processed for the current gallery picture then the recognised painting's relevant rect is drawn on the image along with the painting number above it. This is the resulting image and these are show in a subsequent part of this report.

Method Discussion

This method worked quite well for the image set provided but unfortunately there are some clear cases where it will fail.

Some segments make it through all the filtering stages and are not paintings. One example of this is shown below where a segment of the floor is cropped and compared to all the templates.

But this is why the threshold was introduced, in this case this segment gets a max average of 0.1704 which is far below the threshold of 0.4. The threshold itself is concerning as it's quite low meaning if a segment happens to have very similar pixel values to a template painting it could pass this threshold. But in my opinion this would be a rare case and getting the average correlation of template matching and HS histogram comparison in two different colour spaces should reduce the risk of this occurring. For all the paintings there was a clear max value in the template matching result. Regardless I wish the threshold could be higher than the current value of 0.4.

The method of zooming could also introduce errors in cases where the frame has already been removed and it ends up zooming in on the painting. This will negatively affect the results of comparison with it's correct template and could possibly be below the threshold or be incorrectly recognised as another similar template. As stated previously on average the zoom method gained better results than meanshift segmentation. But if performance was similar I would employ the meanshift segmentation approach to avoid this issue. It's worth noting that the meanshift approach has the inverse issue in that it fails to remove the frame from every painting. Example of the zoom being detrimental is shown below.

Before zoom

After zoom

Using Rects to keep track of the different segments is an issue too. The Rect object in openCV is upright and 2D meaning it does not take into account different geometric positions. Images can be taken at different angles and therefore the paintings will be at different orientations. This led to a loss in accuracy at each stage of cropping the image which is one reason the correlation was never very high. Geometric operations could have been added in order to fix the orientation of the paintings found back to straight on similar to the template. But this is not a trivial step, one possible way would be by using the line of the inner frame and computing the angle or slope of the line from one side to the other. This would be done both vertically and horizontally. Being able to successfully apply this step would have greatly increased the template matching results and improved area of overlap between the painting identified by the program and the ground truth.

Another edge case is if a painting is part of the boundary of an image (it has pixels at any edge of the image matrix). It will not be recognised as it will be filtered out in an early step by the filterBoundaryRects method. This was introduced to remove floor/ceiling segments and I believe it's use is necessary in the program.
The program will likely fail in cases where the assumptions do not hold. Where the wall in not the same colour or contrasting the painting frames. Or if there are multiple different colour walls in the image with painting on each wall.

Results

Result Gallery 1

Ground truth Gallery 1

Result Gallery 2

Ground truth Gallery 2

Result Gallery 3

Ground truth Gallery 3

Result Gallery 4

Ground truth Gallery 4

Results of correlation

Gallery No	Segment No	Max TempMatch Corr	Hist Corr	Avg Corr	Predicted Painting No	Actual Painting No
1	0	0.8588	0.2018	0.5303	2	2
1	1	0.2147	0.860326	0.537513	1	1
2	0	0.8238	0.2938	0.5588	2	2
2	1	0.443	0.8644	0.6537	3	3
3	0	0.5218	0.356163	0.4389815	4	4
3	1	0.2716	0.6835	0.47755	5	5
3	2	0.4411	0.5011	0.4711	6	6
4	0	0.3005	0.6044	0.45245	5	5
4	1	0.3621	0.4792	0.42065	6	6
4	2	0.4875	0.356445	0.4219725	4	4
4	3(Floor)	-0.0581	0.3989	0.1704	Below threshold	Below threshold

The metrics were calculated with the following assumptions:

True Positive (TP) - If area of overlap between a predicted painting and the ground truth results for the same painting is above a threshold of 0.6.
False Positive (FP) - If program wrongly identifies a painting which is either not present in the ground truth or the area of overlap is below the threshold.
True Negative (TN) - If program correctly doesn't identify a painting if it is not present in the ground truth image or the result image.
False Negative (FN) - If program fails to recognise a painting which is present in the ground truth.

*Note: In this case TN value is always 0. I believe this value would be redundant as it doesn't provide any additional information on the performance of the program.

These are counted for each gallery image and following metrics are calculated:

Precision = TP / (TP+FP)

Recall = TP / (TP+FN)

Accuracy = TP+TN / (TP+FP+FN+TN)

F1 Score = 2 (Recall * Precision) / (Recall + Precision)*

These results are automatically calculated and outputted by the program. Below is a screenshot of the output and the identical table of results.

```
Gallery No : 1
Recognised Painting2
Recognised Painting1
TP  FP  TN  FN  |  Precision   |  Recall  |  Accuracy  |  F1 Score
2  0  0  0  |  1  |  1  |  1  |  1

Gallery No : 2
Recognised Painting2
Recognised Painting3
TP  FP  TN  FN  |  Precision   |  Recall  |  Accuracy  |  F1 Score
2  0  0  1  |  1  |  0.666667  |  0.666667  |  0.8

Gallery No : 3
Recognised Painting4
Recognised Painting5
Recognised Painting6
TP  FP  TN  FN  |  Precision   |  Recall  |  Accuracy  |  F1 Score
3  0  0  0  |  1  |  1  |  1  |  1

Gallery No : 4
Recognised Painting5
Recognised Painting6
Recognised Painting4

TP  FP  TN  FN  |  Precision   |  Recall  |  Accuracy  |  F1 Score
3  0  0  0  |  1  |  1  |  1  |  1


Average over all Gallerys
Precision | Recall | Accuracy | F1 Score
1  |  0.916667  |  0.916667  |  0.95
```

Gallery No	TP	TN	FP	FN	Precision	Recall	Accuracy	F1 Score
1	2	0	0	0	1	1	1	1
2	2	0	0	1	1	0.666	0.666	0.8
3	3	0	0	0	1	1	1	1
4	3	0	0	0	1	1	1	1
Average					1	0.9166	0.9166	0.95

Results Discussion

The program performed well for the input images given and I believe would work well in general if avoids the cases discussed previously in method discussion. 1/11 Paintings is not recognised. This is the Painting 1 in Gallery 2. This is the single false negative painting and decreases the average recall, accuracy and F1 score. But not the precision as there's not incorrectly recognised paintings, this shows the need for multiple metrics. Take for example a program which randomly returns a painting number this would get a perfect recall result of 1 but perform poorly in the other 3 metrics. Therefore taking all four metrics and producing an average for each gives a better overall picture of the performance of the program.

It's clear from the resulting images that the program does not locate the paintings at the exact same points as the ground truth. The ground truth were done by hand /eye so this makes sense and in my opinion they're close to one another. It is particularly noticeable in the images where the paintings are shot at an angle. This is due upright Rects being used in the method, the limitations of this was discussed previously in the method discussion. It's also worth noting that these limitations are also a reason for the poor correlation scores.

I considered including a DICE coefficient score but the goal of this task was to locate and recognise the paintings in the images given and not locate their exact position in the image. This is reflected in the metrics used The program already ensures a high threshold for the area of overlap between the painting and the ground truth. This threshold needs to be passed in order for the painting to be considered a true positive result.