Objective

The aim of the assignment is to locate and recognise paintings in a gallery. Once the paintings have been identified, the paintings found by the program should evaluate the performance of the system in terms of the Accuracy, Precision and Recall.

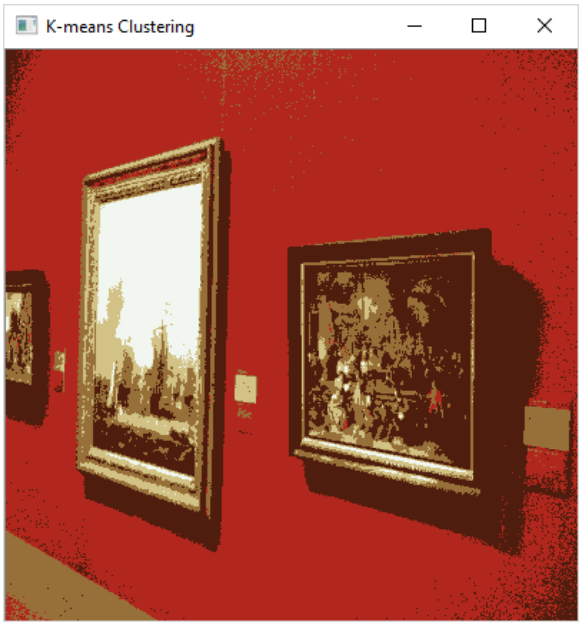
Method

- Resize image

The images were downsampled to a smaller image as it was found that the solution worked better across all the images if they were of a similar size.

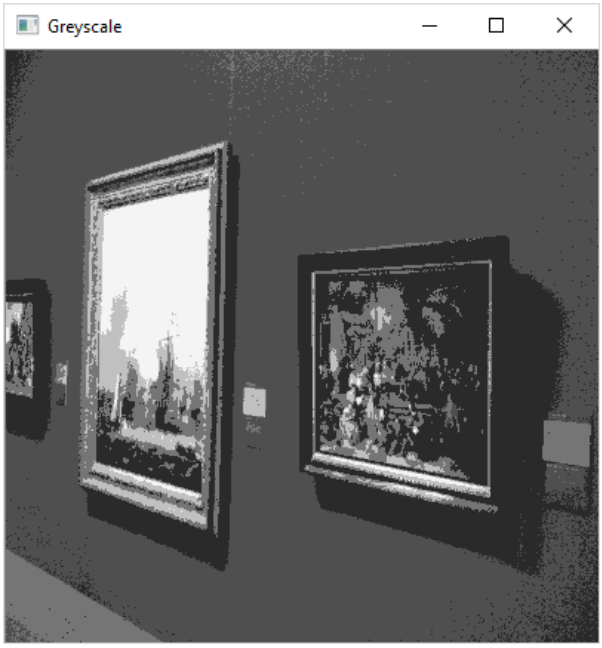
- K-Means Clustering

K-Means is a clustering algorithm. That means you can "group" points based on their neighbourhood. This allows for colour segmentation and makes for easier distinguishing as the number of colours in the image has been reduced



- Greyscale

The image is converted to grayscale using the cvtColor function and the CV_BGR2GRAY code. Grayscale is used predominately in image processing and edge detection because of the intensity of the converted image. Grayscale allows for edges to be distinguished and does not suffer from the complexity issues of processing colours. It is also needed when thresholding the image.



- Threshold

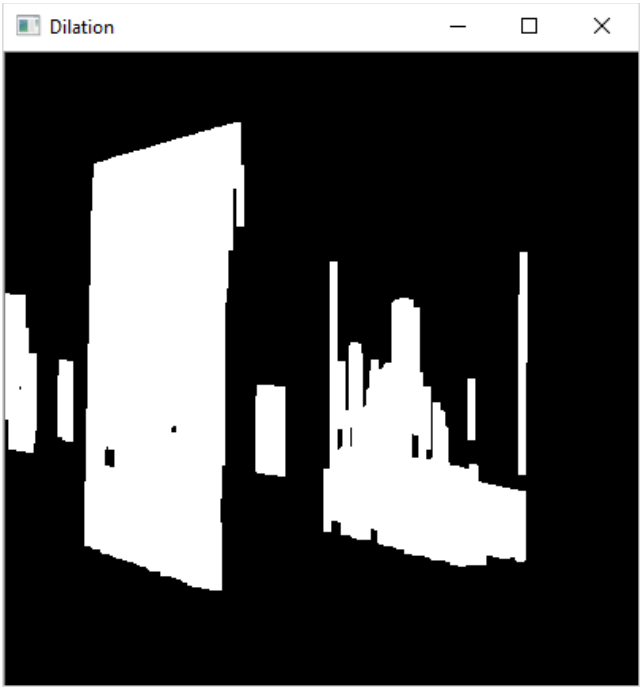
The threshold is done next, thresholding is accomplished by searching for pixels above a certain value and if the pixel fits the criteria, then the pixel is assigned a new value, if the pixel is below then it is black. The threshold value is 150 and any pixels above this are changed white, this allows for the image to be binarised and easier to connect regions.

- Inversion (If needed)

It’s easier to find the regions on a white image on a black background, some images might need to be inverted so that the background is black and white region brought to the foreground. This was accomplished by comparing the white pixels to the black pixels using bitwise_not which inverts every pixel in an image, if more white than black were found.

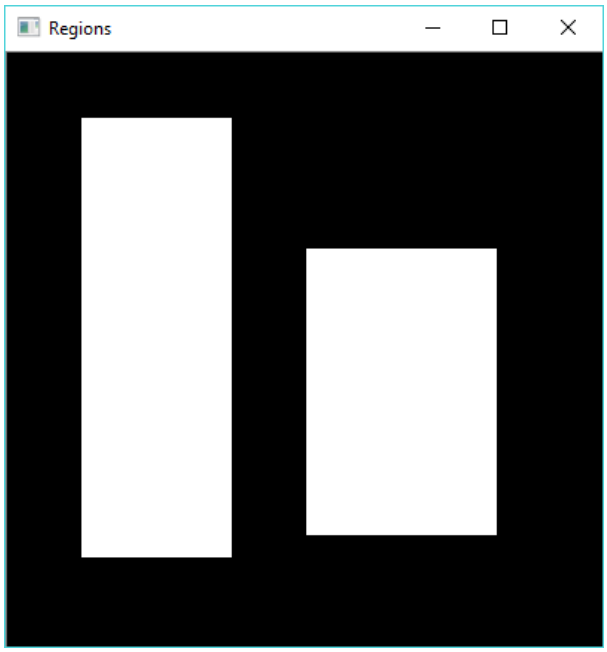
- Dilation

Two dilations are performed once the image is thresholded and inverted, if needed, this was done to increase the size of the white regions so as to better detect and bound the regions.



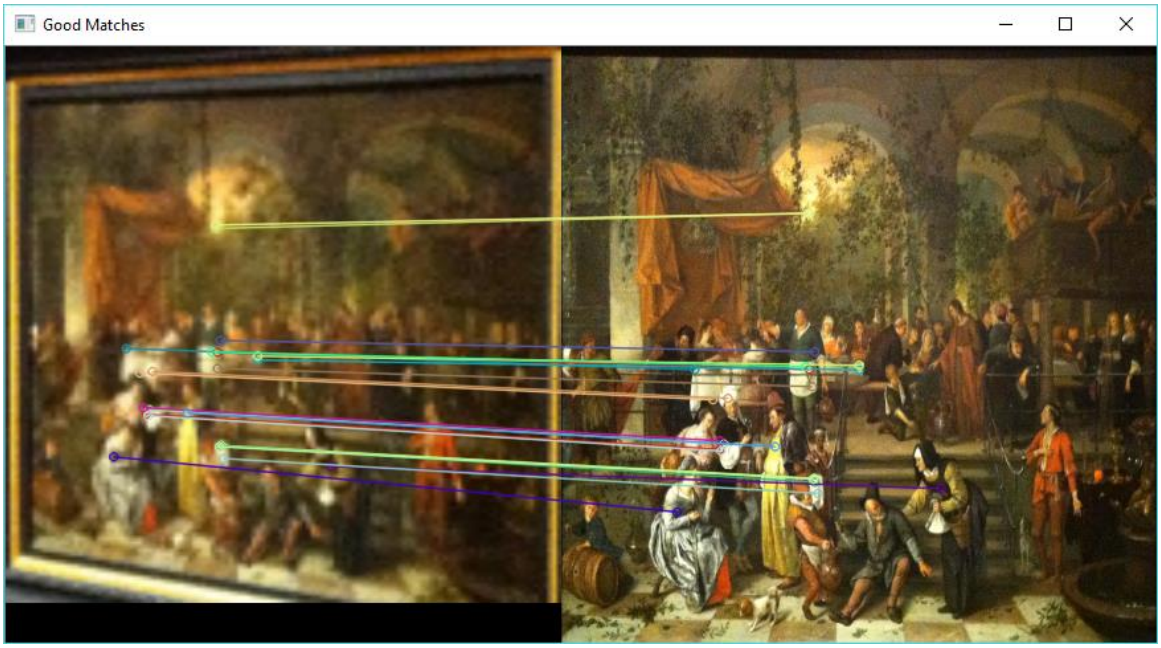
- Region detection

After all these operations are performed on the image, OpenCV's findContours function is used to find regions in the image that are of a certain size. The function finds the points of the region as stores them. The region is bounded by a rectangle at its edges, if the area of the rectangle found is greater than or less than certain values then it is added to the rectangle vector. The stored rectangle points are then drawn on the images.



- Feature detection/matching

Feature detection was used to find the correct paintings from the detected regions, the bounding rectangles were used to crop out the images from the gallery images and each painting was iterated through against the cropped image and similar features were connected across the two images. The painting with the best matches were saved and later a number was drawn indicating which painting the cropped image best matched. Feature matching was accomplished using ORB feature detector which would find the features in the paintings and image. BFM was used to match the similar features of each image to each other.



- Performance calculation

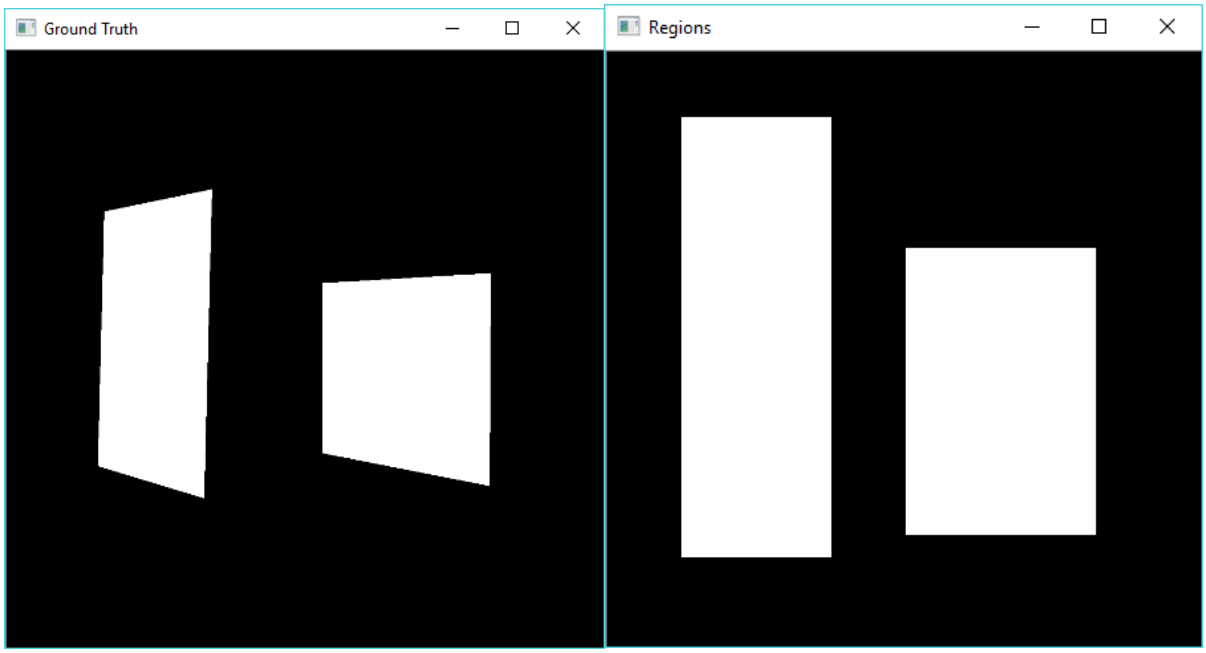
For the performance evaluation, the dice coefficient was used to calculate how accurate the regions were to the ground truth regions. Depending on how accurate the dice was to the ground truth would be used to calculate Accuracy, Precision and Recall.

Dice Coefficient

$$Dice\ Score = 2(A \cap B)\ /A + B$$

- A is the total number of pixels that are white in my image.
- B is the number of pixels that are white in the ground image.
- A∩B is the total number of pixels that are white in both A and B. So it the intersection of the regions of ones in A and B.

The ground truth images are made using the given points of the rectangles and placed onto the images. The images are converted to black and the bounding rectangle boxes filled in with white. This is also done using the bounding boxes found by my solution. The white pixels in these two images are counted and used to calculate the dice coefficient as per the equation above.



Accuracy, Precision and Recall

$$\text{Accuracy} = \text{TP}+\text{TN}\ /\text{TP}+\text{FP}+\text{FN}+\text{TN}$$

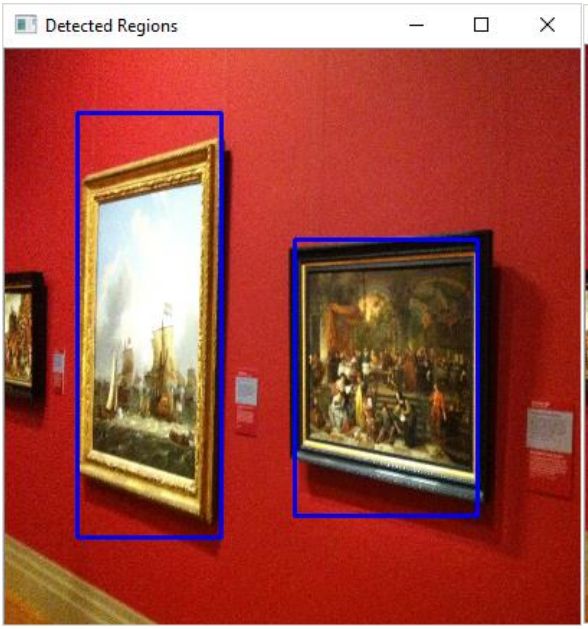
$$\text{Precision}= \text{TP}/\text{TP}+\text{FP}$$

$$\text{Recall} = \text{T}/\text{TP}+\text{FN}$$

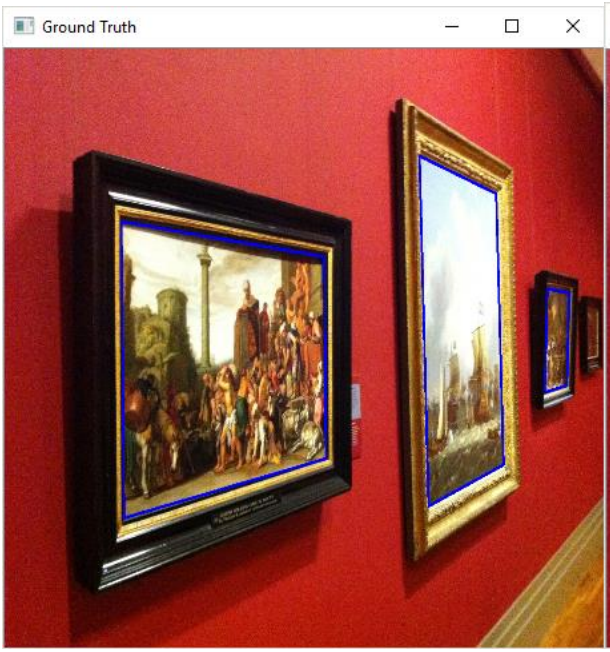
- A true positive(TP) test result is one that detects the condition when the condition is present.
- A true negative(TN) test result is one that does not detect the condition when the condition is absent.
- A false positive(FP) test result is one that detects the condition when the condition is absent.
- A false negative(FN) test result is one that does not detect the condition when the condition is present.

Results

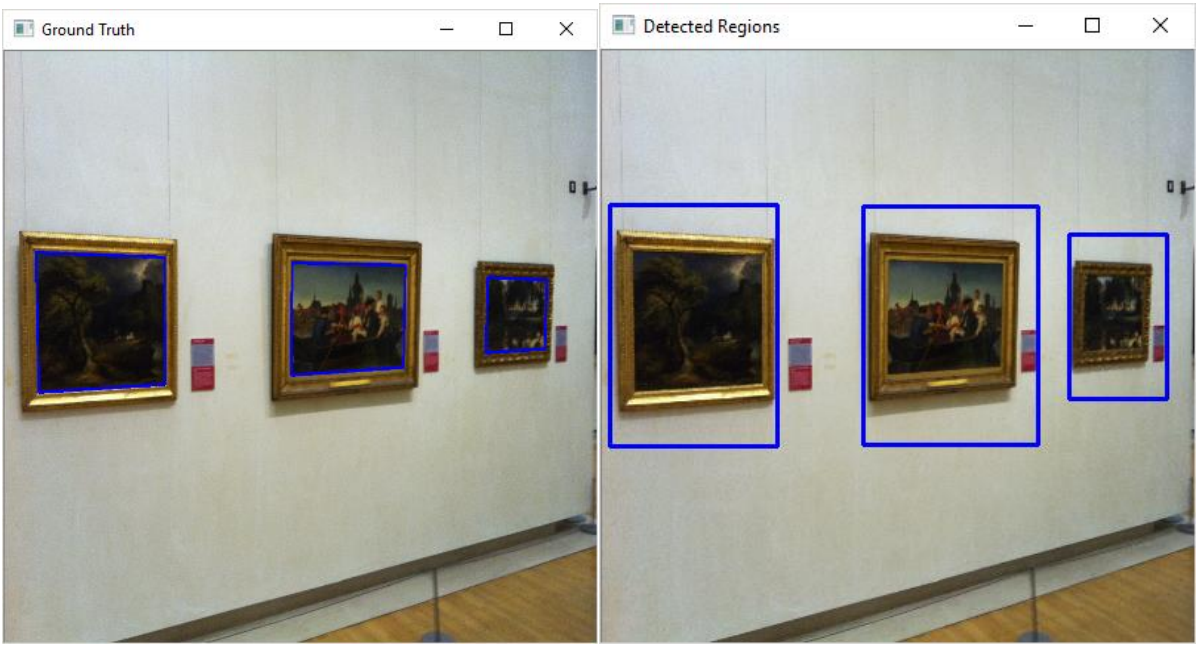
Gallery 1. Paintings Expected: Painting 1 and 2. Paintings Detected: Painting 1 and 2.

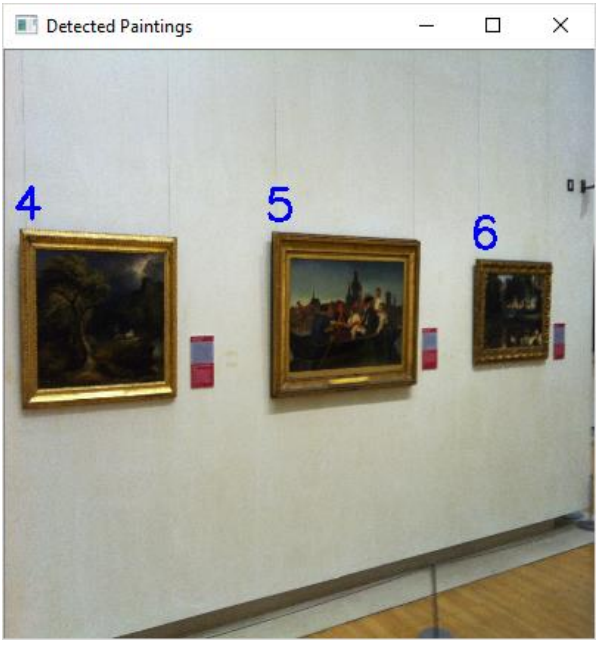


Gallery 2. Paintings Expected: Painting 1, 2 and 3. Paintings Detected: Painting 2 and 3.

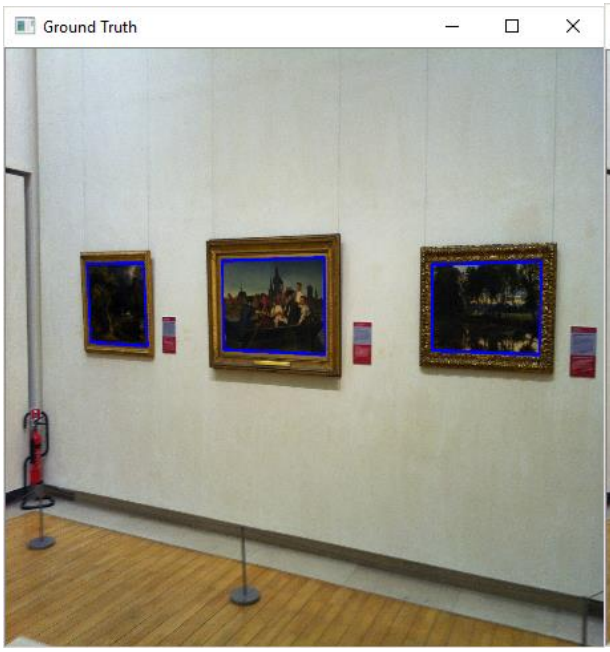


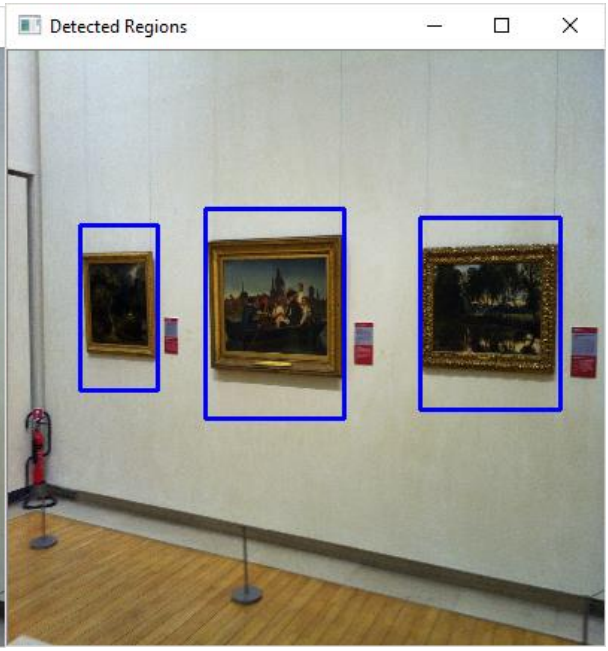
Gallery 3. Paintings Expected: Painting 4, 5 and 6. Paintings Detected: Painting 4, 5 and 6.





Gallery 4. Paintings Expected: Painting 4, 5 and 6. Paintings Detected: Painting 4, 5 and 6.



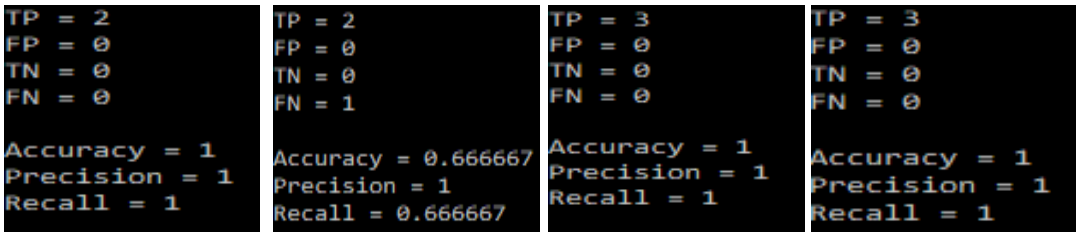


Conclusion

As far as detection and feature matching the results of the program are rather good it matches 10/11 paintings and only misses painting 1 in Gallery 2 which was exceedingly difficult to get and I am doubtful whether my feature matching technique could have identified the painting either.

One significant improvement would have been to better bound the regions using Hough Lines instead of bounding the regions with rectangles as this made the pictures very large when doing region detection. I offset this by cropping the frame a small bit but this is unlikely to work every time.

The solution is fairly appropriate for the assignment as it is simple but gives decent results. It can obviously be improved but it completes the job to a fairly high standard. The program does take a while to run and as to be done one gallery at a time for optimal results.



The performance calculation was quite difficult as the bounding regions were quite a bit bigger than the ground truth regions and although the features matched and 91% of the right paintings were identified the performance calculation while correct was likely poorly implemented by using the dice coefficient.

Regarding the metric, while TP is fairly simple, it was difficult to ascribe meaning to TN as my program never detects anything it shouldn't, it does however not find one painting (FN) and never finds something that it shouldn't (FP).