Brian Leung UofM ’22

image_adjustments

b01lers_ctf 2020

CATEGORY: Images

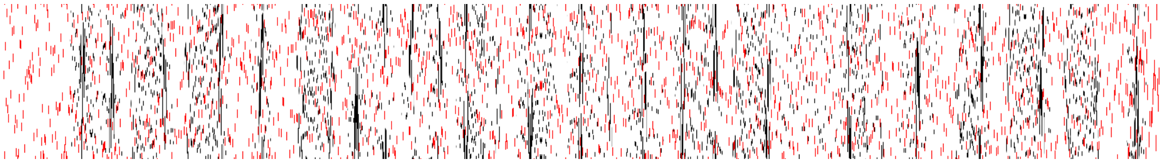
PTS: 100

DESC: We are given the scrambled flag image with red and black lines scattered around a white background. Extract the flag from manipulating this image back to normal.

TOOLS: MATLAB, Snipping Tool

DETAILED APPROACH (Solution Summary at end):

We are given a 1500x200 ‘image_edited.png’ and opening it reveals:



Hella chaotic right? Well, with further inspection, the black blobs on this image look a lot like characters. After all, they’re spaced fairly evenly and are vertically aligned.

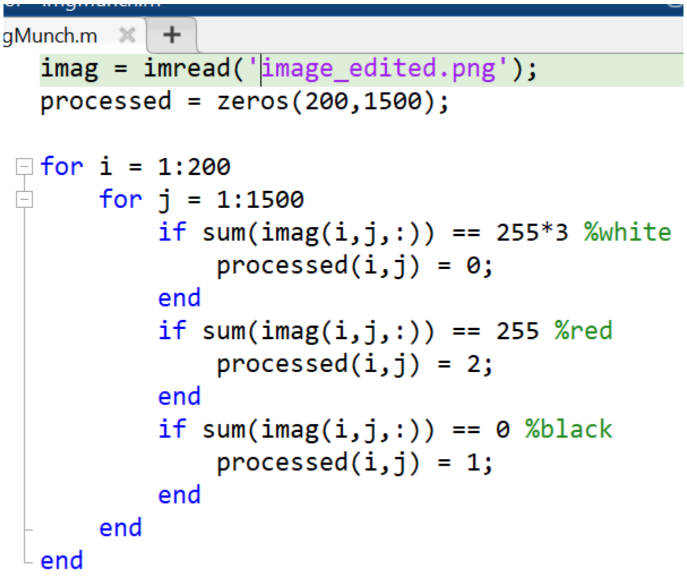
Well, since the characters are spaced out evenly and seemingly not scrambled horizontally, it leads me to believe that the columns of the images were only moved up and down to achieve this.

I’m thinking I can move the columns up and down like one of those word locks to align everything together.

This also leads me to wonder, what is the significance of the red segments? I have a feeling that they must be something for aligning the blobs into readable characters.

To confirm this, I want to check that the red segments are of similar length and that there is only one segment per column of pixels in the image. This ensures that they can be aligned, w/o problems.

I loaded the image into MATLAB since windows photo viewer doesn't allow you to really examine individual pixels.

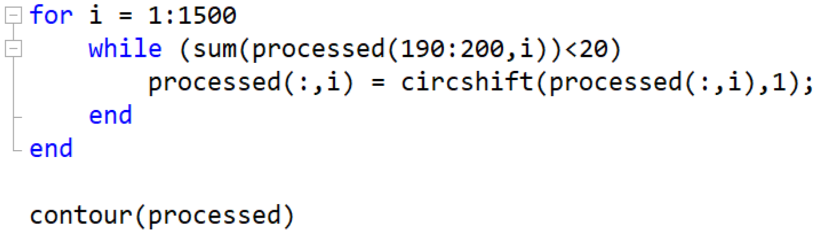


```
imag = imread('image_edited.png');
processed = zeros(200,1500);

for i = 1:200
    for j = 1:1500
        if sum(imag(i,j,:)) == 255*3 %white
            processed(i,j) = 0;
        end
        if sum(imag(i,j,:)) == 255 %red
            processed(i,j) = 2;
        end
        if sum(imag(i,j,:)) == 0 %black
            processed(i,j) = 1;
        end
    end
end
```

This block of code basically says, "Make a big matrix named 'processed' with the same dimensions of the image. For each pixel in this image, replace each white pixel with a 0, each red with a 2, and each black with a 1."

Crawling through the variable viewer and examining each segment of 2s confirmed that they were unique for each column, and that they were all about the same length.

Well, let's align them then and see what happens.



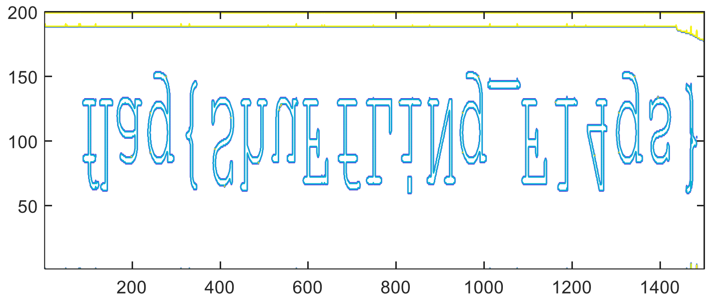
```
for i = 1:1500
    while (sum(processed(190:200,i))<20)
        processed(:,i) = circshift(processed(:,i),1);
    end
end

contour(processed)
```

My intention for this block is to align all the red segments at the bottom of the image. Since we have the ‘processed’ matrix, we don’t have to worry about have to iterate through pixel after pixel, treat each column as a nice and simple array, and just do math stuff on it.

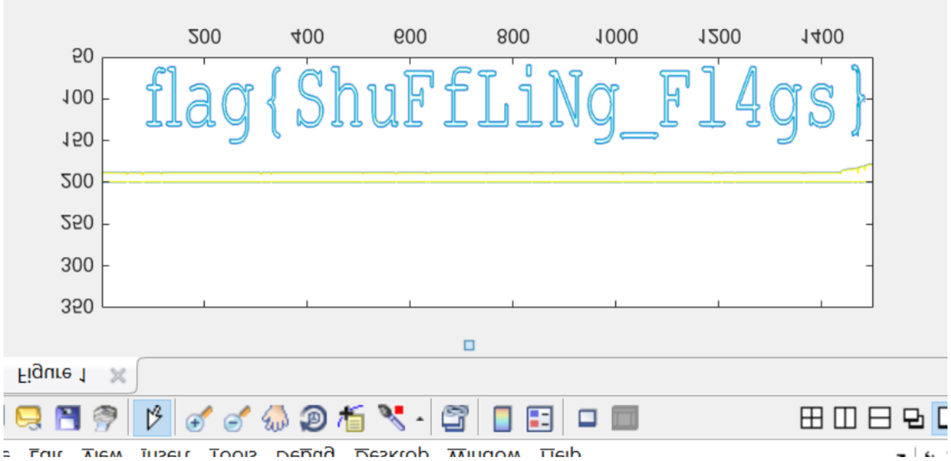
(Skip this paragraph if you don’t need math stuff)

In order to make sure that the red segments are all aligned at the bottom, I want the sum of the numbers in indices 190 through 200 greater than equal to 20. The only way for that to happen is if all the 2s (reds) are placed in those indices (and each 2 segment is about 10 long). circshift(matrix,up) shifts the matrix circularly, like the columns in the word lock above. It keeps shifting it by 1 until the condition in the while fulfills.

By now, all the red segments are aligned and we can simply contour() ‘processed’ to see what shapes came out.



Yep that's a flag. I could do some fancy matrix tricks on 'processed' to get the graph to look right, but snipping tool and some mirroring can do the same.



flag{ShuFfLiNg_Fl4gs}

SOLUTION SUMMARY:

Align the red segments of each column in the image together and the flag will appear.

P.S.: *Why MATLAB?*

For manipulating images as matrices, MATLAB is an incredibly strong tool. Python is not bad, but MATLAB's development environment is a one-stop shop for viewing images, inspecting individual values of cells, and handling big matrices in general (after all it was built with that purpose in mind).

CODE:

```
imag = imread('image_edited.png');
processed = zeros(200,1500);

for i = 1:200
    for j = 1:1500
        if sum(imag(i,j,:)) == 255*3 %white
            processed(i,j) = 0;
        end
        if sum(imag(i,j,:)) == 255 %red
            processed(i,j) = 2;
        end
        if sum(imag(i,j,:)) == 0 %black
            processed(i,j) = 1;
        end
    end
end

for i = 1:1500
    while (sum(processed(190:200,i))<20)
        processed(:,i) = circshift(processed(:,i),1);
    end
end

contour(processed)
```