Replicating the Photo-DNA algorithm

Introduction

Photo-DNA is a tool developed by Microsoft, which is most notably used to detect child pornography. Although the exact algorithm is kept secret for obvious reasons, some people have made attempts to hypothesise how it could work. Among those is Dr. Neal Krawetz, who published an article in his blog to explain his theory[1]. The goal of this project is to replicate his approach and evaluate the results.

How does the algorithm work?

I will only briefly summarise the approach, for those wanting more details and reasoning behind the approach, I suggest taking a look at Dr. Krawetz' work.

First the image is converted to grayscale and reduced to 26 x 26 pixels. After that the image is divided into blocks of 6x6, which overlap by 2. To calculate a hash the sobel gradient is applied to the 6x6 grid. The gradient is applied in x- and y-direction resulting in two matrices, an example for which can be found in fig. 1.

0 1 1 1 1 0	**0 0 0 0 0 0**
0 2 2 2 2 0	**1 2 3 4 5 6**
0 3 3 3 3 0	**1 2 3 4 5 6**
0 4 4 4 4 0	**1 2 3 4 5 6**
0 5 5 5 5 0	**1 2 3 4 5 6**
0 6 6 6 6 0	**0 0 0 0 0 0**

Fig. 1 Resulting matrices after sobel gradient.

The hash consists of four values, horizontally increasing/decreasing and vertically increasing/decreasing. The horizontal values are calculated by adding all the positive numbers for the increasing part and adding the negative numbers for the decreasing part.

Note: I tried to replicate what Dr. Krawetz did, but my values are vastly different from his. If someone wants to exactly replicate the results, this part of the calculation should be the first thing they look at.

After the hashes have been calculated the difference is calculated in three ways, although I only will focus on *linear distance* for my evaluation, which is being calculated as follows:

$$\text{Ld} = \sqrt{\sum_{n=0}^{144} (h_1(n) - h_2(n))^2}$$

with hashes h_1, h_2 and n as the position in the hash

[1] https://hackerfactor.com/blog/index.php?/archives/931-PhotoDNA-and-Limitations.html

Results

To ensure that the algorithm works, I will test it for robustness. To achieve this three images will be compared against themselves with different scaling and cropping. In fig. 2 there is an exemplary graph for scaling and cropping.

The graphs show that the discrepancy between the hashes increases exponentially with the scaling. It also increases with cropping, a worrying sign is that cropping starts at 2293.94, which is way higher than any value for scaling. This will be discussed further after the foundation for comparison has been established.

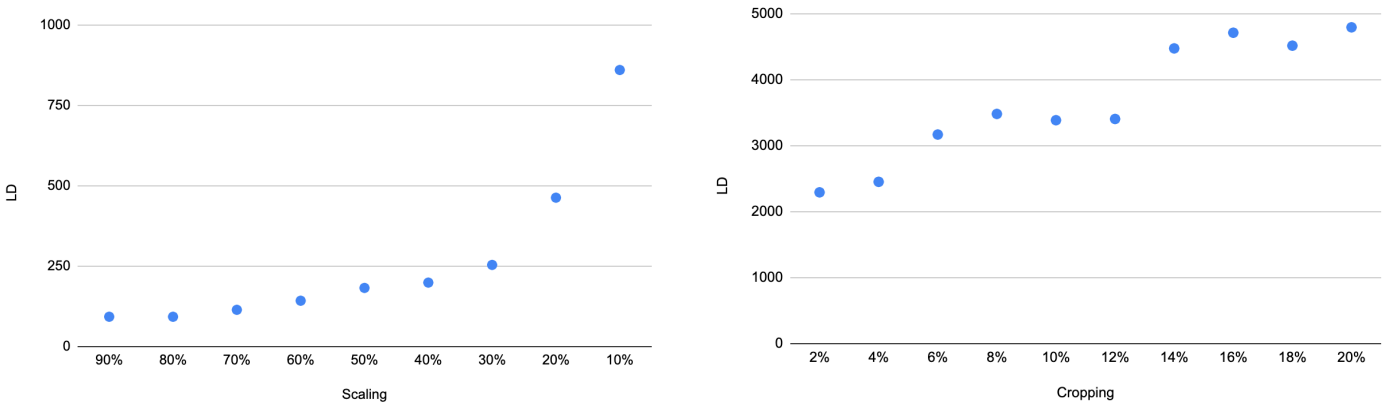

Fig. 2 Results of the windmill image

Creating a foundation for comparison

Next step is a comparison of a set of images to one another and evaluate the results. There were 39 images in this particular set. To find a frame of reference to judge the results, the minima and maxima are examined.

The absolute minimum is the comparison between *Zelda2* and *ZeldaColor*, which is 871.97. Looking at the pictures this is unsurprising since it is the same picture, the only difference being the latter being coloured and larger in size. Because the images are so similar one can conclude that a value for a slight difference is approximately smaller than 1000. While this is good to know, it is not good enough to establish a baseline, therefore the second smallest value is being looked at, which is *Truck* and *Tank2* (fig. 3) with a value of 2011.25. On first glance this makes sense, the two being transport vehicles after all. On closer examination the surroundings are somewhat similar, due to both images being taken in a desert environment. To further cement our hypothesis the relationship between *Truck*, *Tank* and *Tank2* is being looked at, since they have a similar background and subject matter. All the values are between the aforementioned minimum and 2479.58, therefore a value between 2000 and 2500 means the images are very similar.

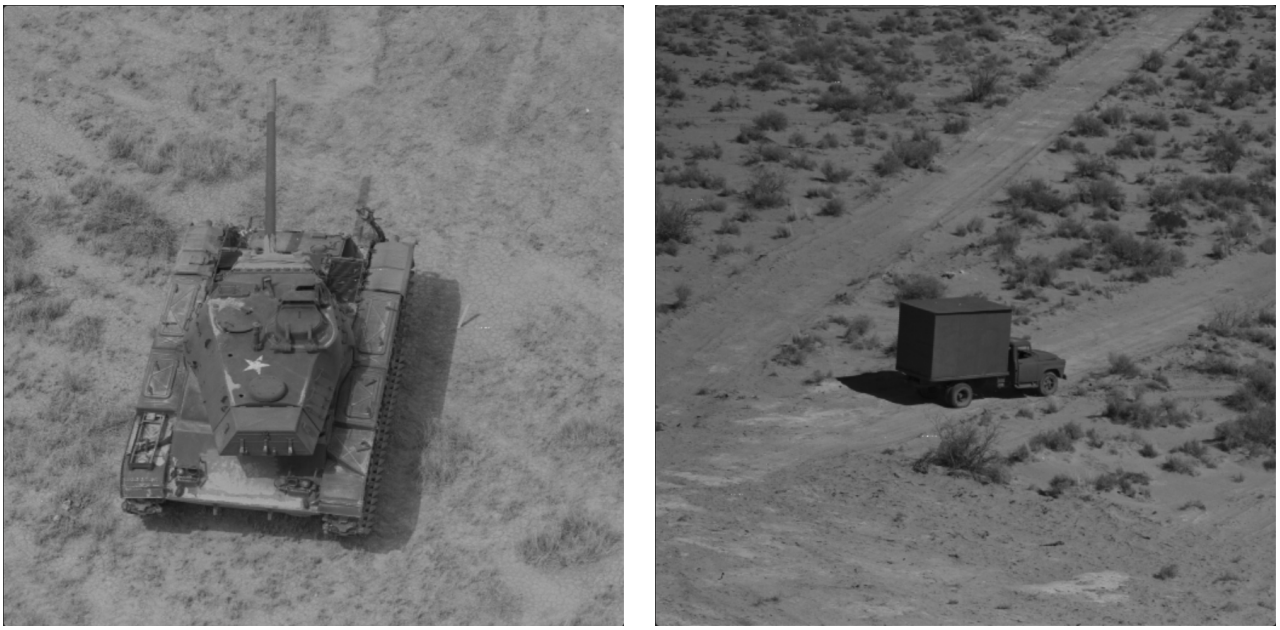

Fig. 3 Images *Tank* (left) and *Truck* (right)

On the other side of the spectrum there is the maximum for the images Windmill and Pens, which is 8048.83. On inspection the value seems reasonable and will be used as reference for images that are not alike at all.

Fig. 4 Images *Pens* (left) and *Windmill* (right)

A criteria is introduced, which shall be named *certainty*. This is calculated for value *v* as follows:

$$c(v) = 1 - \frac{v - offset}{max - offset} \; with \; max = 8048.83 \; and \; offset = 800$$

800 has been chosen as offset arbitrarily, since the value for almost identical was 871.97. By dividing with the max value, we get a percentage. This percentage can be read as *"I am …% sure, that x is similar to y"*. For the examples above this would lead to *"I am 99 % sure that Zelda2 and ZeldaColor are similar."* and *"I am 83.3 % sure that a tank is similar to a truck."*.

Now that there is a foundation, the data for cropping is evaluated once more. Under the assumption that a value above 70 % is fine, then the following statement can be made: when the original image is cropped by more than 4%, the algorithm doesn't recognize the original image. Therefore it can be said that the algorithm is robust against scaling, but not robust against cropping.

Comparing and discussing the results

For starters values that are greater than 70 % certainty are evaluated. There are two groups emerging, which are the *Zelda* images and the military vehicles (*truck*, *tank*, *tank2* and *Tiffany)*.

Fig. 4 Images *Zelda, Zelda2 and ZeldaColor (left to right)*

We discussed earlier the similarity between the coloured and uncoloured *Zelda* image, yet the certainty goes down to 71.1 % and 74.8 % respectively when comparing *Zelda* to *ZeldaColor* and *Zelda2*.

As for the second group, there is definitely an odd one out. The algorithm seems to be 78.6 % sure that she is either a truck or a tank, which has to be a false positive.

To go further the similarity between the adolescent women is evaluated, specifically the *Zelda* images, *Tiffany, Lenna, Barbara,* and *Girlface*. The expected outcome is that they are all detected to be similar. The results can be seen in table 1.

Barbara	
Image name	**Certainty**
crowd.bmp	57
flowers.bmp	54.3
couple.bmp	54
lenna.bmp	52.7
soccer.bmp	52.6

Girlface	
Image name	**Certainty**
zelda.bmp	48.6
baboon.bmp	42.1
couple.bmp	41.9
flowers.bmp	41.8
lenna.bmp	41.8

Lenna	
Image name	**Certainty**
couple.bmp	54.8
barbara.bmp	52.7
tank2.bmp	50.4
baboon.bmp	49.4
tiffany.bmp	48

Tiffany	
Image name	**Certainty**
tank2.bmp	78.6
truck.bmp	78.6
tank.bmp	73.2
zelda2.bmp	65.3
airplaneU2.bmp	65.3

Zelda	
Image name	**Certainty**
zelda2.bmp	74.8
ZeldaColor.bmp	71.1
tank2.bmp	63.8
tiffany.bmp	62.3
truck.bmp	60.7

Zelda2	
Image name	**Certainty**
ZeldaColor.bmp	99
zelda.bmp	74.8
tank2.bmp	68.3
trucks.bmp	66.7
tiffany.bmp	65.3

ZeldaColour	
Image name	**Certainty**
zelda2.bmp	99
zelda.bmp	71.1
trucks.bmp	61.7
tank2.bmp	58.9
tiffany.bmp	56.3

Table 1: Comparison of adolescent women

If we evaluate this strictly then every result that wasn't in our set is a miss, resulting in a detection rate of 42.85%. If the evaluation would be more forgiving, humans could be included, resulting in a detection rate of 57.14%.

Lossy compression

To test how the algorithm deals with lossy compression, two images are selected: tank.bmp and *zelda.bmp*. Both are being compressed to .jpg-format with a decrementing quality factor. The resulting charts can be seen in fig. 5

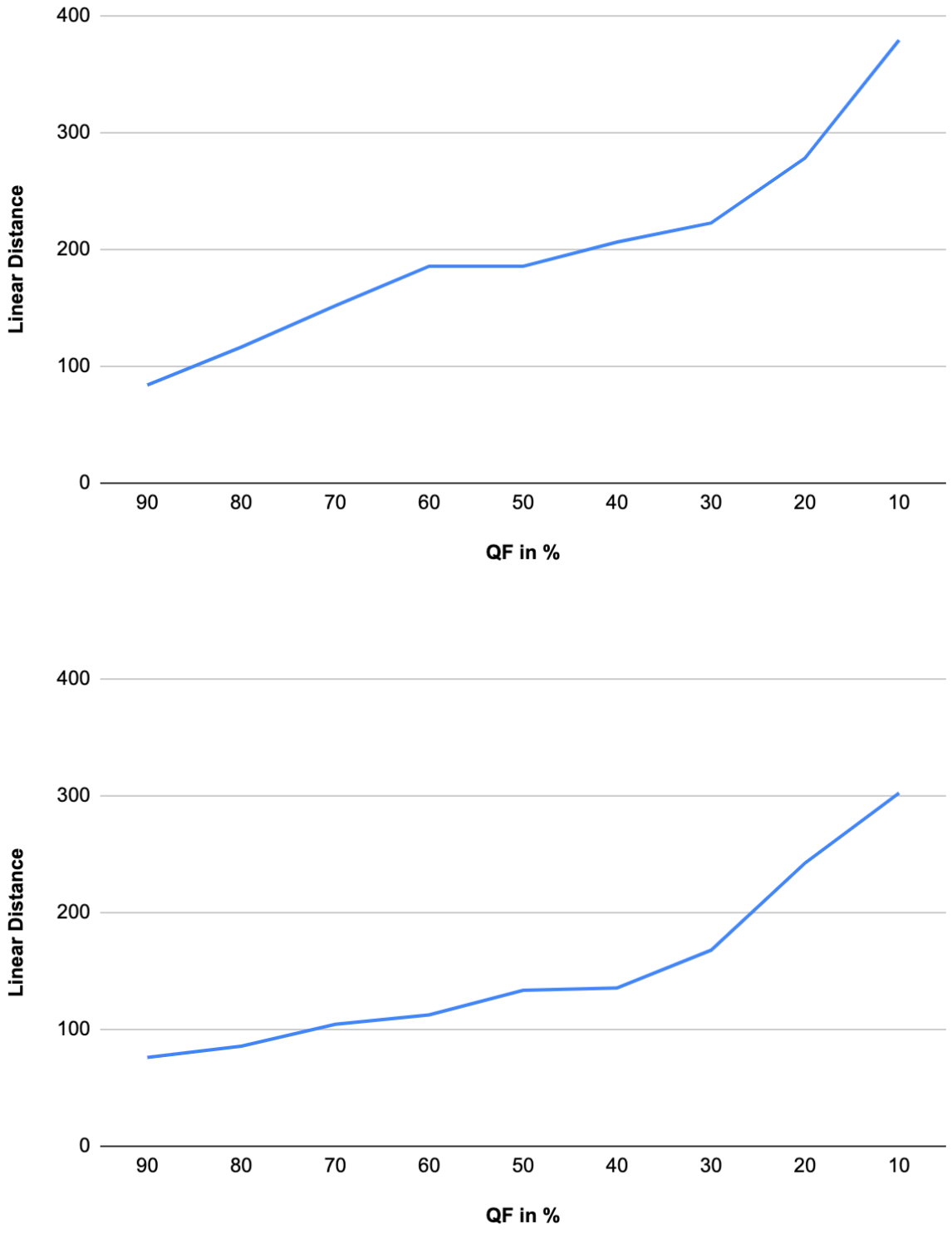

Fig. 6 Charts depicting the change in linear distance when compressing the original image

As one would expect the smaller the images get the less certain the algorithm is that it is in fact the original image. Yet putting these numbers into perspective then even the largest value is smaller than our smallest value in the evaluation. In terms of our introduced criteria *certainty* that would mean that in the worst case the algorithm is 105.8% sure that after reducing the size to 10% of the original's size, it is still the original *tank*.

The new assumption is that since the smaller image is almost identical to the original image, surely the comparison we made before should come out with similar results. And indeed the biggest change of *certainty* was -1.23% for the comparison to *pepper*, meaning that *pepper* is even less likely to be a tank. The algorithm is even more sure with our previous true positives (*truck* and *tank2*), yet it has to be noted that it is also slightly more sure that *tiffany* is a tank.

Summary

The algorithm is working as intended with scaling, cropping and lossy compression. And while some hits are objectively correct, namely the *tank, tank2* and *truck* comparison, there are false positives that cannot reasonably be justified, specifically the assumption that *tiffany* is a *tank*. This leads me to the conclusion that while this algorithm shows promise it is not on par with the widely used product by Microsoft.

Outlook

To improve this implementation one could take another look at the calculation of the hashes, since some assumptions were made, which was explained at the beginning. Perhaps the processing of the image when scaling it down to 26 x 26 pixels is also worth looking into.