Image Censoring Tool: Mutify

Mert Can Balcı

Istanbul Technical University

Information Systems Engineering

Abstract: On the Internet we see lots of images and sometimes we don't want to see those. In this paper an image mute tool: Mutify is proposed. Mutify based on Template Matching to find similarities and to censor those Gaussian Blur, Fuzzying or just deleting is used. Success rate and constraints are demonstrated and discussed.

I. INTRODUCTION

Internet and PC is a part of our lives, just like the media, every single thing we seen on the Internet have an influence on us. Even though a couple of words can change our perspective think about images, which are stronger than words. Sometimes, a trend may be disturbing or uninteresting for us and because it is a trend we can see it that image so much time**.** We can mute words or people on the Internet or we can just filter words on our PC. But sometimes in our pc there are images that we don't want to see anymore or we just want to blur, hide or censor some images that we don't want to see. But dealing with images is not easy as dealing with words, you cannot search for certain images in a web-page or in a folder. If there is so many images in a folder that process can be annoying and time-wasting.

An image censoring tool Mutify solves that kind of problems, by using Template Matching and Rotation to identify similarity and uses Gaussian Blur, Interpolation and resizing to censor images. In section II Image types and constraints will be discussed. Section III will be about Template Matching and Gaussian Blur alongside, resizing and interpolation. Section IV will include success rate and results while Section V will include conclusion.

II. BACKGROUND

JPG files are widely common on the Internet and most of data, game documents and personal documents includes JPG files mostly. While other image types like .PNG exist in Mutify only JPG comparison and censoring can be done. Besides those, identifying similarity between two objects is a tricky and hard process when you aim for near perfect results. Even Google cannot give best results in its image searching algorithm. Since it is hard to accomplish such an advanced success Mutify's main aim is doing best with the most efficient solution in a simple manner. Since it's a simple and efficient algorithm, it can be applied to many platform which was the first goal when I thought about this Project. Muting, or censoring images in every platform. As I introduced, this is a common problem for many of us, and better and advanced solutions for this problem will be one of the best discovery for multimedia systems.

According to my researches, I didn't find any approach or tool similar to Mutify on the Internet. Since it is very complicated process, there should be unpopular approaches which I couldn't find.

a)Original Image

- b)Modified Image

III. PROPOSED APPORACH

In this paper, approach to problem is based on comparisons and several modifications in images. First an image to compare with target folder(images) is taken by user, then to apply Template Matching, that image is converted into Gray Scale image. Then after images from target taken file by file, a couple of processes applied.

A. Template Matching

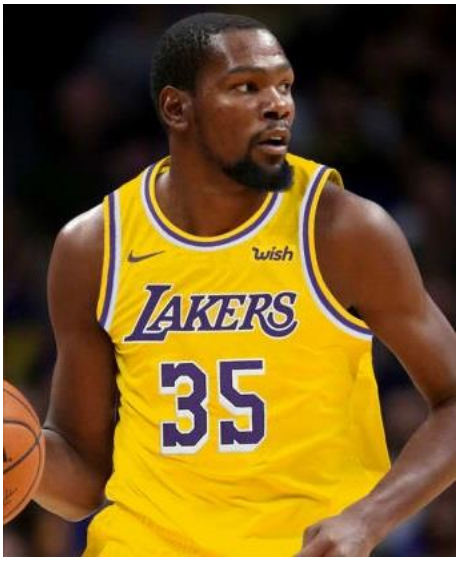
Template Matching is applied to both source and target image to compare and found similarity between them. Template Matching does the comparison by going through pixel by pixel and comparing them, since it is a simple approach, it is fast but not that successful. It only detects near same or same pictures, but sometimes it can fail on same pictures too due to resolution for example Figure (a) and Figure(b) couldn't be detected as similar by Template Matching. Comparison does by sliding window. When a similarity found it return where the similarity is. While Template Matching returns where is the similarity, our approach isn't interested with where the similarity is. After converting both source and target into gray scale image, comparing give us a result. Selecting a threshold is a tricky job in here. After finding the most suitable threshold comparing it with the output will give us is there any similarity found or not. Since Template Matching is a sliding window algorithm. It cannot detect rotated images, since its pixel values different

but before that Template Matching prevents us to compare them because after a rotation the size of the photo will change and that change may lead to incomparable images. To deal with that problem and to check if our image is rotated version of target image or vice-versa, I simply rotated the image until a similarity found and to prevent any error both images are checked if they are comparable.

B. Gaussian Blur

After finding similarity, depending on user input, images from target folder, is blurred, fuzzied or deleted. Blurring is done with Gaussian Blur, At first, Gaussian blur creates adjacency matrix on target image. After that,values in that matrix is applied to each pixel to remove noise from image, which also blurs it. Because of that matrix, in program a code is added to increase Blur level or decrease, it is done with multiplying kernel size. Blurred image replaces target image.

c) Original Image



d) Modified image

C. Resizing and Interpolation

If Fuzzy option is selected, program applies almost same process until resizing, with resize function, Image's resolution is decreased by function, which leads to a fuzzy image which is used by many people to censor images, interpolation is applied to maintain images color quality.

In delete option program simply deletes the target image if it is similar to our source image.

IV. EVALUATION

The success criteria is subjective and difficult to determine, since everyone's expectation from this tool can vary. Also it is very successful to find same images or any images includes another, also according to my trials, it is also successful on modified images in certain conditions. Since in this paper an unsuccessful example has been given, but also there are some successful modified entries like Figure (c) and (d) above.

And censored form of those images is on the next page as (e) and (f).

The success rate of the program(g) is very good on the simple tasks. Among 60 images program gives 55 successful result to us approximately 92%, which is impressive for Template Matching. On the other hand for hard tasks it's success rate significantly drops at 30% levels. Simple tasks are generally same pictures with little to no modification besides cropping, rotating, etc. Hard tasks are meanwhile again very similar but not same images with somewhat more modification. It can be implied that in hard tasks if resoultion is same, Template Matching can be successful but otherwise, not. Resolution dependancy also applies to Simple tasks, even in simple tasks with different resolution can fail. But different resolution images aren't included in hard tasks generally.

V. CONCULUSION

This paper presented an image muting, censoring tool Mutify with its components. The way mutify handles tasks is demonstrated under Background and Proposed Approach. Results are evaluated under evaluation part,

results show that Template Matching is a simple, efficient but highly insufficient algorithm when it comes to detect similarity. Tool is successful at simple tasks. But for similarity detection it needs some improvement.

REFERENCES

Template Matching. (n.d.). Retrieved June 9, 2020, from https://docs.adaptive-vision.com/4.7/studio/machine_vision_guide/TemplateMatching.html

Gaussian Blur - an overview | ScienceDirect Topics. (n.d.). Retrieved from https://www.sciencedirect.com/topics/engineering/gaussian-blur

Geometric Image Transformations — OpenCV 2.4.13.7 documentation. (n.d.). Retrieved from https://docs.opencv.org/2.4/modules/imgproc/doc/geometric_transformations.html

Type	Input	Success	Success Rate
Simple	60	55	91%
Hard	10	3	30%