

Name- Vishakha Baviskar

Roll. No. – B220

SAP-Id 70022100337

Submitted On- 25-10-2023

**Automatic Sensitive Information Redaction (protect Privacy) in Images.**

1. **Problem Description:**

Developing a Solution that automatically blurs and redacts Sensitive information in images such as faces or license plate to protect Privacy.

With the increasing use of images and videos in various applications, there is a growing concern regarding the privacy of individuals captured in these media. Sensitive information, such as faces, license plates, and personal identifiers, should be protected to ensure the privacy and security of individuals. The problem is to develop a solution that can automatically identify and redact or blur sensitive information in images to safeguard privacy.

In today's digital age, images are used in a wide range of applications, including social media, surveillance, news reporting, and more. However, sharing or publishing these images often involves exposing individuals' sensitive information, such as faces and license plates, which can lead to privacy concerns. Automated solutions for identifying and redacting this sensitive information are crucial to ensure compliance with privacy regulations and ethical standards.

**2. Proposed Solution:**

- Image Analysis: Develop an image analysis module that can detect sensitive information within images, specifically faces and license plates.

- Redaction Techniques: Implement redaction techniques such as blurring, pixelation, or other privacy-preserving methods to protect the identified sensitive information.

- Real-time Processing: If applicable, ensure that the solution can process images in real-time or near-real-time without significant latency.

- Customization: Allow users to customize redaction parameters (e.g., intensity of blurring) to balance privacy protection and usability.

- Scalability: Ensure that the solution can scale to process a large number of images efficiently.

**3. Stakeholders:**

- Developers and engineers

- Privacy advocates

- Government organizations

- Companies using surveillance systems

- Media and news agencies

- Anyone who handles and shares images with sensitive information.

Addressing this problem requires advanced image processing techniques, ethical considerations, and a deep understanding of privacy laws and regulations. The solution should provide a balance between privacy protection and practical usability in a wide range of applications.

Develop a solution that automatically blurs or redacts sensitive information in images such as faces or license plates to protect privacy.

**4. code:**

% Load the image

image = imread('face5.jpeg');

% ... (code for skin color detection and region properties) ...

gray\_image = rgb2gray(image);

% Define the skin color range in the RGB color space (adjust as needed)

lower\_skin\_color = [100, 40, 20];

upper\_skin\_color = [255, 175, 130];

% Create a binary mask for skin color regions

skin\_mask = (image(:,:,1) >= lower\_skin\_color(1)) & (image(:,:,1) <= upper\_skin\_color(1)) ...

& (image(:,:,2) >= lower\_skin\_color(2)) & (image(:,:,2) <= upper\_skin\_color(2)) ...

& (image(:,:,3) >= lower\_skin\_color(3)) & (image(:,:,3) <= upper\_skin\_color(3));

% Apply morphological operations to clean up the mask (adjust as needed) close small gaps in the binary mask.

skin\_mask = imclose(skin\_mask, strel('disk', 7));

skin\_mask = imfill(skin\_mask, 'holes');

% Compute region properties (i.e., bounding boxes) of the skin regions

stats = regionprops(skin\_mask, 'BoundingBox');

% Apply Gaussian blur to each detected face with a smaller size and blur

for i = 1:numel(stats)

bbox = stats(i).BoundingBox;

% Extract x, y, width, and height from the bounding box

x = max(1, round(bbox(1)));

y = max(1, round(bbox(2)));

% Reduce the width and height to make the face smaller

w = min(size(image, 2) - x, round(bbox(3)) \* 0.8); % Adjust the factor to control size

h = min(size(image, 1) - y, round(bbox(4)) \* 0.4); % Adjust the factor to control size

% Extract the face region

face = image(y:y+h, x:x+w, :);

% Apply Gaussian blur with a smaller sigma value

sigma = 60; % Adjust the value for a smaller blur

blurredFace = imgaussfilt(face, sigma);

% Replace the original face with the blurred face

image(y:y+h, x:x+w, :) = blurredFace;

end

% Display the image with redacted faces

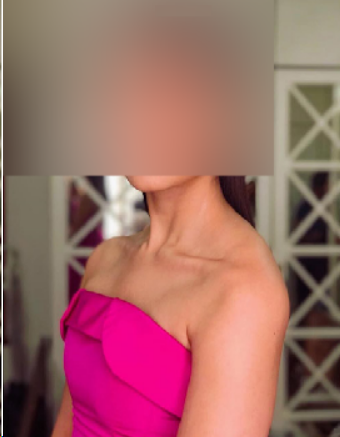
imshow(image);

**Output:**

Original images.



Blurred or redacts images



**Bluring the license plate region.**

% Load the image

image = imread('blur1.png');

% Convert the image to grayscale

gray\_image = rgb2gray(image);

% Apply edge detection (e.g., using Canny)

edges = edge(gray\_image, 'Canny');

% Find contours in the edge-detected image

stats = regionprops(edges, 'BoundingBox', 'Area');

% Determine the largest bounding box (assumed to be the license plate)

[~, index] = max([stats.Area]);

boundingBox = stats(index).BoundingBox;

% Extract and blur the license plate region

x\_start = floor(boundingBox(1));

y\_start = floor(boundingBox(2));

x\_end = x\_start + boundingBox(3);

y\_end = y\_start + boundingBox(4);

% Apply Gaussian blur to the license plate region

sigma = 10; % Adjust the value for the amount of blur

blurred\_license\_plate = imgaussfilt(image(y\_start:y\_end, x\_start:x\_end, :), sigma);

% Replace the original license plate with the blurred license plate

image(y\_start:y\_end, x\_start:x\_end, :) = blurred\_license\_plate;

% Display the image with the redacted license plate

imshow(image);

Original Images



Blurred or redacts images



**Automatic License Plate Detection in Images.**

**1. Problem Description:**

License plate detection is a crucial component in various applications, including traffic management, parking systems, law enforcement, and security. The problem at hand is to develop a solution that can automatically detect and extract license plates from images and videos, enabling the identification of vehicles for purposes such as traffic monitoring, toll collection, and public safety.

In the modern world, the effective and efficient management of traffic and security is essential. License plate detection plays a central role in automating processes like toll collection, parking management, traffic violation monitoring, and even in aiding law enforcement investigations. An automated license plate detection system can significantly improve the accuracy and speed of these processes.

**2. Solution Requirements:**

- License Plate Detection: Develop an image analysis module that can accurately detect license plates within images and video frames.

- Localization: The solution should not only detect the presence of a license plate but also accurately locate its position within the image.

- Real-time Processing: When applicable, the solution should process images and video frames in real-time or near-real-time with minimal latency.

- Adaptability: Ensure that the system can adapt to varying environmental conditions, lighting, and types of license plates.

- Privacy Compliance: If required by law, provide the option to blur or redact other parts of the image to protect privacy.

**3. Stakeholders:**

- Developers and engineers

- Traffic management authorities

- Law enforcement agencies

- Parking management companies

- Private security firms

- Anyone using license plate data for various applications

Addressing this problem requires advanced image processing techniques, ethical considerations, and a deep understanding of privacy laws and regulations. The solution should provide a balance between functionality, accuracy, and privacy compliance, catering to the needs of various applications and industries.

**4.code:**

Plate\_detection.m

close all;

clear all;

im = imread('/MATLAB Drive/plate1.jpg');

imgray = rgb2gray(im);

imbin = imbinarize(imgray);

im = edge(imgray, 'prewitt');

%Below steps are to find location of number plate

Iprops=regionprops(im,'BoundingBox','Area', 'Image');

area = Iprops.Area;

count = numel(Iprops);

maxa= area;

boundingBox = Iprops.BoundingBox;

for i=1:count

if maxa<Iprops(i).Area

maxa=Iprops(i).Area;

boundingBox=Iprops(i).BoundingBox;

end

end

im = imcrop(imbin, boundingBox);%crop the number plate area

im = bwareaopen(~im, 500); %remove some object if it width is too long or too small than 500

[h, w] = size(im);%get width

imshow(im);

Iprops=regionprops(im,'BoundingBox','Area', 'Image'); %read letter

count = numel(Iprops);

noPlate=[]; % Initializing the variable of number plate string.

for i=1:count

ow = length(Iprops(i).Image(1,:));

oh = length(Iprops(i).Image(:,1));

if ow<(h/2) & oh>(h/3)

letter=Letter\_detection(Iprops(i).Image); % Reading the letter corresponding the binary image 'N'.

noPlate=[noPlate letter] % Appending every subsequent character in noPlate variable.

end

end

Letter\_detection.m

function letter=readLetter(snap)

load NewTemplates

snap=imresize(snap,[42 24]);

rec=[ ];

for n=1:length(NewTemplates)

cor=corr2(NewTemplates{1,n},snap);

rec=[rec cor];

end

ind=find(rec==max(rec));

display(ind);

% Alphabets listings.

if ind==1 || ind==2

letter='A';

elseif ind==3 || ind==4

letter='B';

elseif ind==5

letter='C';

elseif ind==6 || ind==7

letter='D';

elseif ind==8

letter='E';

elseif ind==9

letter='F';

elseif ind==10

letter='G';

elseif ind==11

letter='H';

elseif ind==12

letter='I';

elseif ind==13

letter='J';

elseif ind==14

letter='K';

elseif ind==15

letter='L';

elseif ind==16

letter='M';

elseif ind==17

letter='N';

elseif ind==18 || ind==19

letter='O';

elseif ind==20 || ind==21

letter='P';

elseif ind==22 || ind==23

letter='Q';

elseif ind==24 || ind==25

letter='R';

elseif ind==26

letter='S';

elseif ind==27

letter='T';

elseif ind==28

letter='U';

elseif ind==29

letter='V';

elseif ind==30

letter='W';

elseif ind==31

letter='X';

elseif ind==32

letter='Y';

elseif ind==33

letter='Z';

%\*-\*-\*-\*-\*

% Numerals listings.

elseif ind==34

letter='1';

elseif ind==35

letter='2';

elseif ind==36

letter='3';

elseif ind==37 || ind==38

letter='4';

elseif ind==39

letter='5';

elseif ind==40 || ind==41 || ind==42

letter='6';

elseif ind==43

letter='7';

elseif ind==44 || ind==45

letter='8';

elseif ind==46 || ind==47 || ind==48

letter='9';

else

letter='0';

end

end

**Output:**



