CMPE 362 - HW3

Burak Berk Özer - 2016400015 - 27 June 2020

Question - 1

These question ask us to design kernel to apply some filters to given joker image. The filter is basically convolution of a kernel matrix with the image. Convolution operation is implemented in the code. Kernels are found from the web.

1.1) Design a kernel that adds blur to your image.

The kernel: gaussian_blur = [1 2 1; 2 4 2; 1 2 1].*(1/16);

1.2) Design a kernel that sharpens your image found in 1.1 to get rid of the blur.

The kernel: sharpen = [0 -1 0; -1 5 -1; 0 -1 0];

1.3) Design a kernel that highlights edges in your image.

The kernel for x axis: edge_detection_x = [1 0 -1; 2 0 -2; 1 0 -1];

The kernel for y axis: edge_detection_y = [1 2 1; 0 0 0; -1 -2 -1];

Edge detected for x axis

Edge detected for y axis

1.4) Design a kernel that makes your image embossed.

The kernel: embose = [1 1 -1; 1 1 -1; 1 -1 -1];

Code for Question 1)

```
f = imread('jokerimage.png');
f2 = imread('blurred.png');

% Required kernels with some extra options
gaussian_blur = [1 2 1; 2 4 2; 1 2 1].*(1/16);
edge_detection = [-1 -1 -1; -1 8 -1; -1 -1 -1];
edge_detection_2 = [0 -1 0; -1 4 -1; 0 -1 0];
edge_detection_x = [1 0 -1; 2 0 -2; 1 0 -1]; % sobel operator
edge_detection_y = [1 2 1; 0 0 0; -1 -2 -1];
sharpen = [0 -1 0; -1 5 -1; 0 -1 0];
embose = [1 1 -1; 1 1 -1; 1 -1 -1];

filtered = filter(f, gaussian_blur, 'blurred.png');
%filtered = filter(f, edge_detection, 'edge_detected.png');
filtered = filter(f, edge_detection_x, 'edge_detected_x.png');
filtered = filter(f, edge_detection_y, 'edge_detected_y.png');
filtered = filter(f2, sharpen, 'sharpened.png');
filtered = filter(f, embose, 'embosed.png');

function filtered = filter(f, kernel, file_name)
    % get channels from rgb image
    red= f(:,:,1);
    green= f(:,:,2);
    blue = f(:,:,3);
    % pad with zeros
    red_2 = zeros(size(red)+2);
    red_2(2:end-1,2:end-1) = red;
    green_2 = zeros(size(green)+2);
    green_2(2:end-1,2:end-1) = green;
    blue_2 = zeros(size(blue)+2);
    blue_2(2:end-1,2:end-1) = blue;


    % Initialize new matrices
    filtered = zeros(size(f));
    channel_r = zeros(size(red));
    channel_g = zeros(size(blue));
    channel_b = zeros(size(green));
    [rowsx, colsx] = size(red_2);
```

```
% Convolution operation for channel red
for row = 1 : rowsx-3
  for col = 1 : colsx-3
    sub = red_2([row row+1 row+2], [col col+1 col+2]);
    [rowsx_s, colsx_s] = size(sub); %sub matrix values
    [rowsy_k, colsy_k] = size(kernel); %kernel values
    theSum = 0;
    for row_s = 1 : rowsx_s
      for col_k = 1 : colsy_k
        theSum = theSum + sub(row_s, col_k) * kernel(row_s, col_k);
      end
    end
    channel_r(row, col) = theSum;

  end
end

% Convolution operation for channel green
for row = 1 : rowsx-3
  for col = 1 : colsx-3
    sub = green_2([row row+1 row+2], [col col+1 col+2]);
    [rowsx_s, colsx_s] = size(sub); %sub matrix values
    [rowsy_k, colsy_k] = size(kernel); %kernel values
    theSum = 0;
    for row_s = 1 : rowsx_s
      for col_k = 1 : colsy_k
        theSum = theSum + sub(row_s, col_k) * kernel(row_s, col_k);
      end
    end
    channel_g(row, col) = theSum;
  end
end

% Convolution operation for channel blue
for row = 1 : rowsx-3
  for col = 1 : colsx-3
    sub = blue_2([row row+1 row+2], [col col+1 col+2]);
    [rowsx_s, colsx_s] = size(sub); %sub matrix values
    [rowsy_k, colsy_k] = size(kernel); %kernel values
    theSum = 0;
    for row_s = 1 : rowsx_s
      for col_k = 1 : colsy_k
        theSum = theSum + sub(row_s, col_k) * kernel(row_s, col_k);
```

```
        end
      end
      channel_b(row, col) = theSum;
    end
  end

  % concat all 3 channels to get rgb image
  filtered = cat(3, channel_r, channel_g, channel_b);
  filtered = uint8(filtered);
  % write to file
  imwrite(filtered, file_name);


end
```

Question 2)

These question asks us to detect the cigarette and replace it with a flower. This is done by utilizing Matlab's computer vision library. First, images are being converted to gray scale. Then, SURF points are detected for the scene image which is the Joker's whole picture and object image which is the cigarette. Then their features are being extracted. The features are then being matched with matchFeatures function. By using estimateGeometricTransform outliers are eliminated and with transformPointsForward we get the polygon around the cigarette. An important point to note here is that, the given image of the joker was not high enough resolution (this means there are not enough SURFFeatures to identify given object) so this script first identify a larger area than cigarette and then scales it down to its center to pinpoint the cigarette.

The cigarette.

Detected cigarette.

Cigarette replaced with a flower.

The code)

```
f1 = imread('jokerimage.png');
f2 = imread('cropped.png');
f3 = imread('flower.png');

replaced = replace(f1, f2, f3, 'replaced.png');

function replaced = replace(f1, f2, f3, file_name) % replace f2 in f1 with f3.

   % cast images to gray scale and get SURF features
   sceneImage = rgb2gray(f1);
   objectImage = rgb2gray(f2);
   objectPoints = detectSURFFeatures(objectImage);
   scenePoints = detectSURFFeatures(sceneImage);

%    figure;
%    imshow(objectImage);
%    title('100 Strongest Feature Points');
%    hold on;
%    plot(selectStrongest(objectPoints, 100));
%
%    figure;
%    imshow(sceneImage);
%    title('300 Strongest Feature Points');
%    hold on;
%    plot(selectStrongest(scenePoints, 300));

   % extract features from points
   [objectFeatures, objectPoints] = extractFeatures(objectImage, objectPoints);
   [sceneFeatures, scenePoints] = extractFeatures(sceneImage, scenePoints);

   % match features
   objectPairs = matchFeatures(objectFeatures, sceneFeatures);
```

```
    % get matched points
    matchedObjectPoints = objectPoints(objectPairs(:, 1), :);
    matchedScenePoints = scenePoints(objectPairs(:, 2), :);
%    figure;
%    showMatchedFeatures(objectImage, sceneImage, matchedObjectPoints, ...
%        matchedScenePoints, 'montage');
%    title('Putativelty Matched Points w/ Outliers)');


    [tform, inlierObjectPoints, inlierScenePoints] = ... % Eliminate outliers
        estimateGeometricTransform(matchedObjectPoints, matchedScenePoints,
'affine');

    objectPolygon = [1, 1; ... % Bounding polygon of objectImage
        size(objectImage, 2), 1; ...
        size(objectImage, 2), size(objectImage, 1); ...
        1, size(objectImage, 1); ...
        1, 1];

    newObjectPolygon = transformPointsForward(tform, objectPolygon); %
Transform into coordinate system of image

    % Display detected object
%    figure;
%    imshow(sceneImage);
%    hold on;
%    line(newObjectPolygon(:, 1), newObjectPolygon(:, 2), 'Color', 'y');
%    title('Detected Object');
%    newObjectPolygon;
    % Polygon needs to be narrowed to only include the cigarette.
    cigarettePolygon = newObjectPolygon;

    % Since cigarette is much smaller than the object used to locate the
    % cigarette, the polygon needs to be scaled down in size to only
    % include the cigarette. This was required because of the low
    % resolution of the original image.
```

```
    x_diff = (newObjectPolygon(2, 1) - newObjectPolygon(1, 1))/4;
    y_diff = (newObjectPolygon(3, 2) - newObjectPolygon(1, 2))/4;
    xLeft = sub2ind(size(cigarettePolygon),[1 4 5], [1 1 1]); % x left coordinates
    xRight = sub2ind(size(cigarettePolygon),[2 3], [1 1]); % x right coordinates
    yUp = sub2ind(size(cigarettePolygon),[3 4], [2 2]); % y up coordinates
    yDown = sub2ind(size(cigarettePolygon),[1 2 5], [2 2 2]); % y down coordinates
    cigarettePolygon(xLeft) = cigarettePolygon(xLeft) + x_diff;
    cigarettePolygon(xRight) = cigarettePolygon(xRight) - x_diff;
    cigarettePolygon(yDown) = cigarettePolygon(yDown) + y_diff;
    cigarettePolygon(yUp) = cigarettePolygon(yUp) - y_diff;

    % Display detected cigarette
    figure;
    imshow(sceneImage);
    hold on;
    line(cigarettePolygon(:, 1), cigarettePolygon(:, 2), 'Color', 'y');
    title('The Cigarette');

    % Replace cigarette with a flower
    finalImage = f1;
    % Get bottom left part of the crop coordination
    x = int64(cigarettePolygon(1, 1));
    y = int64(cigarettePolygon(1, 2));
    % Get size of the replace image
    imSize = size(f3);
    imSizeX = imSize(1);
    imSizeY = imSize(2);
    % Replace the area with the flower
    finalImage(x:x+imSizeX-1, y:y+imSizeY-1, :) = f3;
    % Display the finalImage
    figure;
    replaced = finalImage; % return value
    imshow(finalImage);
    imwrite(finalImage, file_name); % write image to file

end
```