IMAGE PROCESSING

LAB 5

AIM: Understanding the use of various filters.

EXERSICE:

1. Take any of your gray scale photo and blur it with standard box filter of size 3x3, 5x5 ,7x7 and 9x9. Comment on amount of blurring and filter size. Assume padding of zeros.

Code:

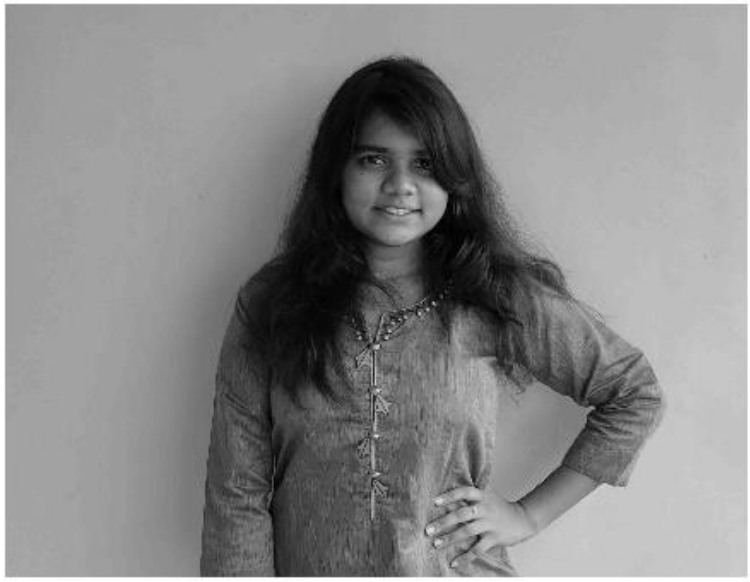
```
img = imread('images/my_img.jpg');
imshow(img);
title("Original Image");
figure;
subplot(2,2,1);
res1 = standard_box(img,3,3);
imshow(res1);
title("3 x 3 image");
subplot(2,2,2);
res2 = standard_box(img,5,5);
imshow(res2);
title("5 x 5 image");
subplot(2,2,3);
res3 = standard_box(img,7,7);
imshow(res3);
title("7 x 7 image");
subplot(2,2,4);
res4 = standard_box(img,9,9);
imshow(res4);
title("9 x 9 image");
```

Function:

```
function s = standard_box(r,m,n)
  [M,N] = size(r);
  x = (m-1)/2;
  y = (n-1)/2;
  new_imsize = zeros(M + 2*x,N + 2*y);
  new_imsize(1+x:M+x,1+y:N+y) = r;
  sub_image = ((1/(m*n)).*ones(m,n));
  s = zeros(size(r));
  for i = 1+x:M+x,
    for j = 1+y:N+y,
      k = new_imsize(i-x:i+x,j-y:j+y);
      s(i-x,j-y) = sum(sum(k.*sub_image));
    endfor
  endfor
  s = uint8(s);
endfunction
```

Output:

Original Image



3 x 3 image

5 x 5 image

7 x 7 image

9 x 9 image

2. Observe border of image for results in (a). Justify the reason for dark borders. Comment on thickness of the border and filter size. Suggest a way to solve the issue. Implement your suggestion and show the code and results.

Code:

```
img= imread('images/my_img_resize.jpg');
subplot(1,2,1);
s1=standard_box(img,7,7);
imshow(s1);
title("Blur Image with border");
subplot(1,2,2);
s2 = border_solved(img,7,7);
imshow(s2);
title("Blur Image with no border");
```

Function:

```
function s = border_solved(r,m,n)
  [M,N] = size(r);
  x = (m-1)/2;
  y = (n-1)/2;
  new_imsize = zeros(M + 2*x,N + 2*y);
  new_imsize(1+x:M+x,1+y:N+y) = r;

  left_part = r(:,1:y);
  right_part = r(:,N-y:N);
  upper_part = r(1:x,:);
  down_part = r(M-x:M,:);

  [M1,N1] = size(new_imsize);
  new_imsize(1+x:M1-x,1:y) = left_part;
  new_imsize(1+x:M1-x,N1-y:N1) = right_part;
  new_imsize(1:x,1+y:N1-y) = upper_part;
  new_imsize(M1-x:M1,1+y:N1-y) = down_part;

  new_imsize(1:x,1:y) = sum(sum(r(1:x,1:y)))/(x*y);
  new_imsize(M1-x+1:M1,1:y) = sum(sum(r(M-x+1:M,1:y)))/(x*y);
  new_imsize(1:x,N1-y+1:N1) = sum(sum(r(1:x,N-y+1:N)))/(x*y);
  new_imsize(M1-x+1:M1,N1-y+1:N1) = sum(sum(r(M-x+1:M,N-y+1:N)))/(x*y);

  sub_image = ((1/(m*n)).*ones(m,n));
  s = zeros(size(r));
  for i = 1+x:M+x,
    for j = 1+y:N+y,
      k = new_imsize(i-x:i+x,j-y:j+y);
      s(i-x,j-y) = sum(sum(k.*sub_image));
    endfor
  endfor
  s = uint8(s);

endfunction
```

Output:

3. Take any of your gray scale photo and blur it with weighted average filter. Compare amount of blurring with the standard box filter of the same size.

Code:

```
img = imread('images/my_img.jpg');
imshow(img);
figure;
res = weighted_average(img,7,7);
imshow(res);
title("Image(weighted average filter 7 x 7)");
```

Function:

```
function s = weighted_average(r,m,n)
  [M,N] = size(r);
  x = (m-1)/2;
  y = (n-1)/2;
  new_imsize = zeros(M + 2*x,N + 2*y);
  new_imsize(1+x:M+x,1+y:N+y) = r;
  weight_image = ones(m,n);
  weight_image(x+1,y+1) = 4;
  weight_image(x+1,y)=2;
  weight_image(x+1,y+2)=2;
  weight_image(x,y+1)=2;
  weight_image(x+2,y+1)=2;
  sub_image = ((1/(sum(sum(weight_image)))).*weight_image);
  s = zeros(size(r));
  for i = 1+x:M+x,
    for j = 1+y:N+y,
      k = new_imsize(i-x:i+x,j-y:j+y);
      s(i-x,j-y) = sum(sum(k.*sub_image));
    endfor
  endfor
  s = uint8(s);
endfunction
```

Output:

4. Assume that you are working on some image enhancement application which gives following functionality to user.

1) Anti-aging: Removes the wrinkles on the input face image.

2) Beautify: Removes facial marks.

Take any of the color photo of a face and implement any (or both) of the above functionality.

Solution:

1) Anti-aging

Code:

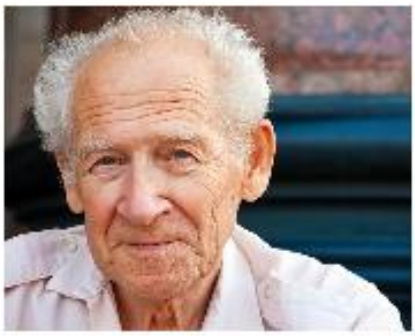
```
img = imread("images/wrinkle_img.jpg");
subplot(1,2,1);
imshow(img);
title("Wrinkled face");

s = uint8(zeros(size(img)));
s(:,:,1) = antiaging(img(:,:,1));
s(:,:,2) = antiaging(img(:,:,2));
s(:,:,3) = antiaging(img(:,:,3));
subplot(1,2,2);
imshow(s);
title("wrinkleless face");
```

Function:

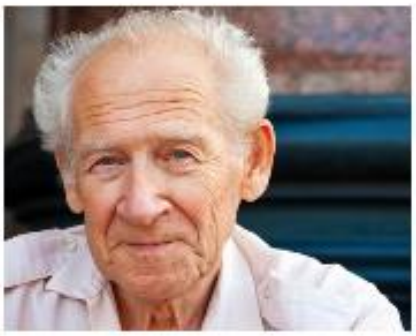
```
function s = antiaging(r)
  [m,n] = size(r);
  x=1;
  y=1;
  new_imsize = zeros(m + 2*x,n + 2*y);
  new_imsize(1+x:m+x,1+y:n+y) = r;
  filter = zeros(3,3);
  filter_box(1,1) = 5;
  filter_box(1,2) = 7;
  filter_box(1,3) = 5;
  filter_box(2,1) = 7;
  filter_box(2,2) = 9;
  filter_box(2,3) = 7;
  filter_box(3,1) = 5;
  filter_box(3,2) = 7;
  filter_box(3,3) = 5;
  filter_box = filter_box/sum(sum(filter_box));
  ans = zeros(size(r));
  for i = 1+x:m+x,
    for j = 1+y:n+y,
      k = new_imsize(i-x:i+x,j-y:j+y);
      ans(i-x,j-y) = sum(sum(k.*filter_box));
    endfor
  endfor
  ans = uint8(s);
endfunction
```

Output:

Wrinkled face



wrinkleless face



2) Beautify: Removes facial marks.

Code:

```
img = imread('images/facemark.jpg');
subplot(1,2,1);
imshow(img);
title("Original Image");

nose = img(97:240,530:640,:);
for i=1:5
  nose(:,:,1) = border_solved(nose(:,:,1),7,7);
  nose(:,:,2) = border_solved(nose(:,:,2),7,7);
  nose(:,:,3) = border_solved(nose(:,:,3),7,7);
endfor
img(97:240,530:640,:) = nose;

l_cheek = img(180:370,270:440,:);
for i=1:5
  l_cheek(:,:,1) = border_solved(l_cheek(:,:,1),7,7);
  l_cheek(:,:,2) = border_solved(l_cheek(:,:,2),7,7);
  l_cheek(:,:,3) = border_solved(l_cheek(:,:,3),7,7);
endfor
img(180:370,270:440,:) = l_cheek;

r_cheek = img(171:370,680:840,:);
for i=1:5
  r_cheek(:,:,1) = border_solved(r_cheek(:,:,1),7,7);
  r_cheek(:,:,2) = border_solved(r_cheek(:,:,2),7,7);
  r_cheek(:,:,3) = border_solved(r_cheek(:,:,3),7,7);
endfor
img(171:370,680:840,:) = r_cheek;
subplot(1,2,2);
imshow(img);
title("Beautified Image");
```

Function:

```
function s = border_solved(r,m,n)
  [M,N] = size(r);
  x = (m-1)/2;
  y = (n-1)/2;
  new_imsize = zeros(M + 2*x,N + 2*y);
  new_imsize(1+x:M+x,1+y:N+y) = r;

  left_part = r(:,1:y);
  right_part = r(:,N-y:N);
  upper_part = r(1:x,:);
  down_part = r(M-x:M,:);

  [M1,N1] = size(new_imsize);
  new_imsize(1+x:M1-x,1:y) = left_part;
  new_imsize(1+x:M1-x,N1-y:N1) = right_part;
  new_imsize(1:x,1+y:N1-y) = upper_part;
  new_imsize(M1-x:M1,1+y:N1-y) = down_part;

  new_imsize(1:x,1:y) = sum(sum(r(1:x,1:y)))/(x*y);
  new_imsize(M1-x+1:M1,1:y) = sum(sum(r(M-x+1:M,1:y)))/(x*y);
  new_imsize(1:x,N1-y+1:N1) = sum(sum(r(1:x,N-y+1:N)))/(x*y);
  new_imsize(M1-x+1:M1,N1-y+1:N1) = sum(sum(r(M-x+1:M,N-y+1:N)))/(x*y);

  sub_image = ((1/(m*n)).*ones(m,n));
  s = zeros(size(r));
  for i = 1+x:M+x,
    for j = 1+y:N+y,
      k = new_imsize(i-x:i+x,j-y:j+y);
      s(i-x,j-y) = sum(sum(k.*sub_image));
    endfor
  endfor
  s = uint8(s);

endfunction
```

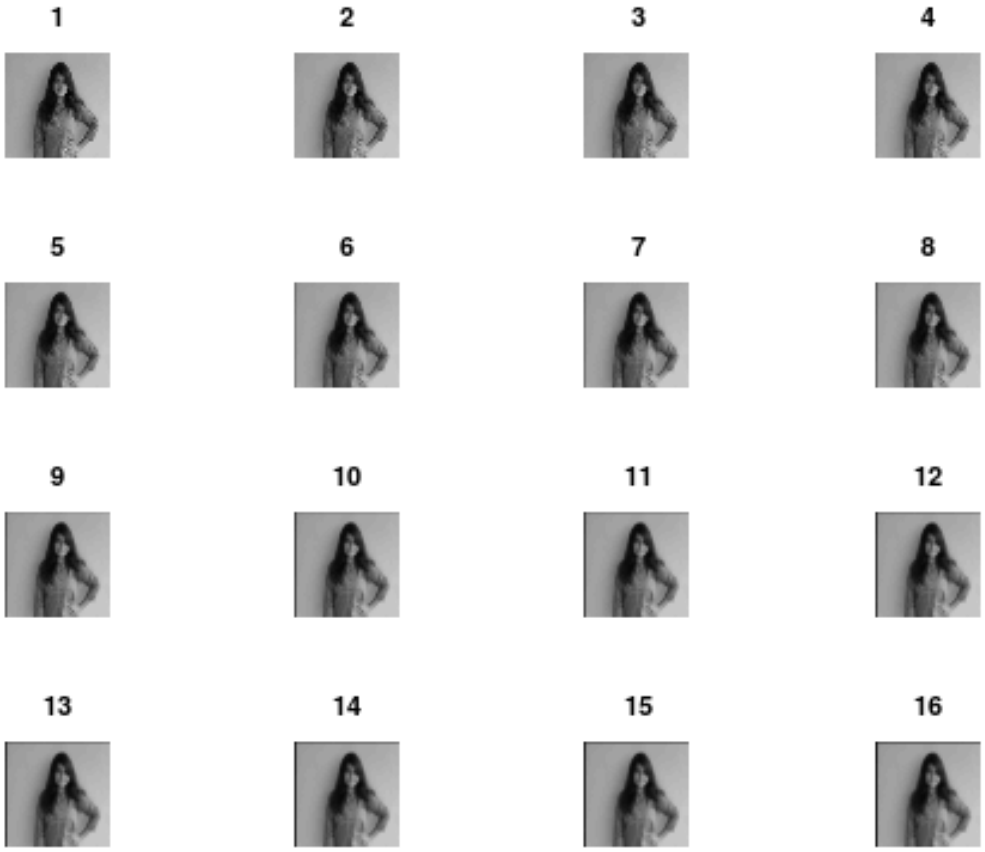
Output:

5. Show the impact of multiple passes of the smoothing filter of same size. Derive your conclusion on image quality and maximum number of passes of filter? What happens if infinite(read very high!) number of passes are applied? Will it change image quality?

Code:

```
img = imread('images/my_img_resize.jpg');
imshow(img);
title("Original Image");
figure;
s = standard_box(img,4,4);
subplot(4,4,1);
imshow(s);
title("1");
for i=2:16
  s = standard_box(s,4,4);
  subplot(4,4,i);
  imshow(s);
  title(i);
endfor
```

Output:



- Here 16 passes for the smoothing is applied and output is shown for every 4 passes.
- As we can see from the output, if the same smoothing filter is applied again and again, then each time smoothing will be increased.
- After a few passes we can't even recognize the image, as the details will be gone.
- So, we can conclude that at infinite passes, the image will turn gray