به نام خدا

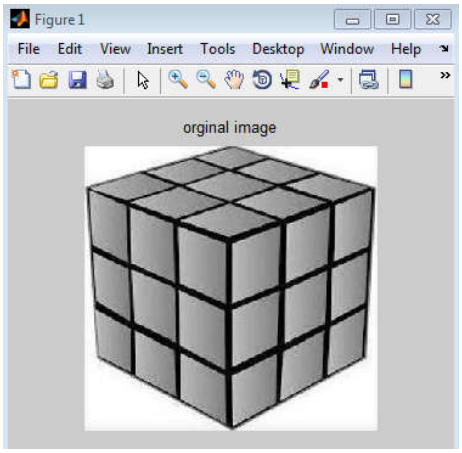
مرجان مودّت **گزارشکار پیاده سازی مقاله**

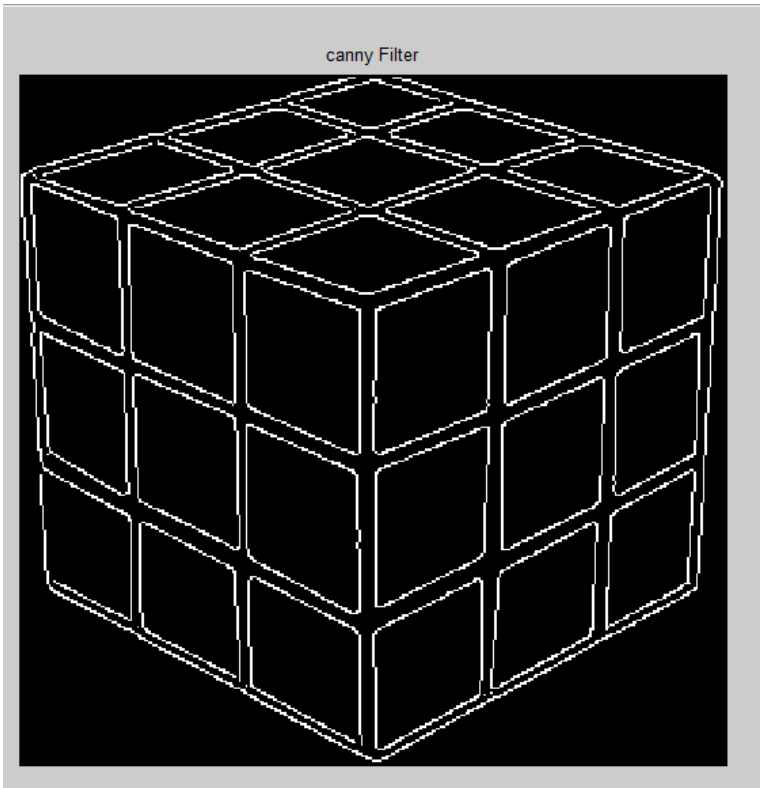
فایل Symmetrical.m

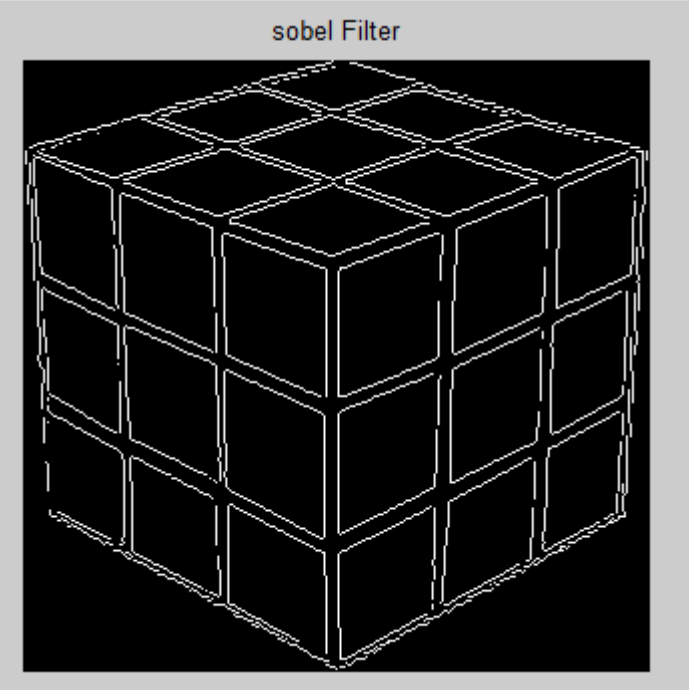
```
I = imread('C:\Users\Marjan\Documents\MATLAB\sio.jpg');
m=rgb2gray(I);
figure,imshow(I);title('orginal image');
x = edge(m,'canny');
figure,imshow(x);title('canny Filter');
w = edge(m,'sobel');
figure,imshow(w);title('sobel Filter');
y = edge(m,'prewitt');
figure,imshow(y);title('prewitt Filter');
z = edge(m,'roberts');
figure,imshow(z);title('roberts Filter');
```

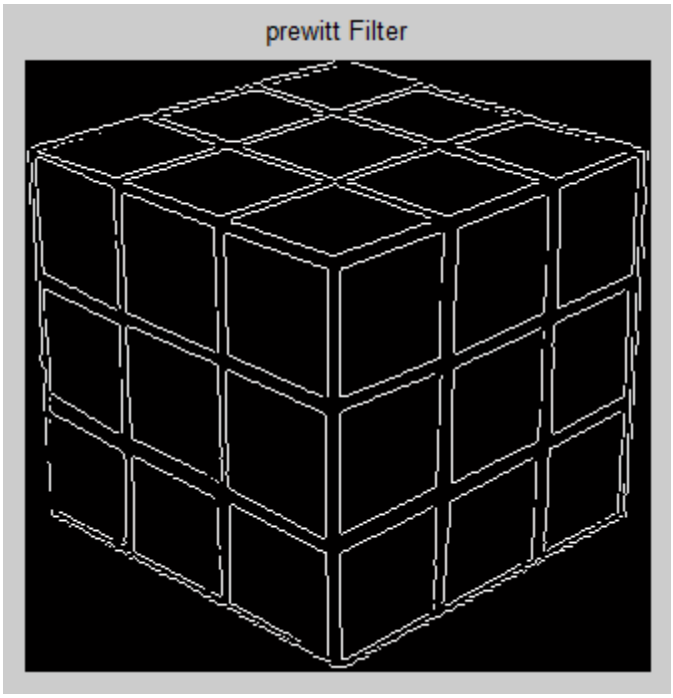
با استفاده از دستور imread تصویر را می خواند و با استفاده از rgb2gray آن را به مقیاس خاکستری تبدیل می کند و با imshow آن را نمایش می دهد در یک figure و با یک عنوان به وسیله ی title. با دستور edge تصویر مقیاس خاکستری و عملگر لبه یابی موردنظر را گرفته و لبه یابی را روی آن اعمال می کند.

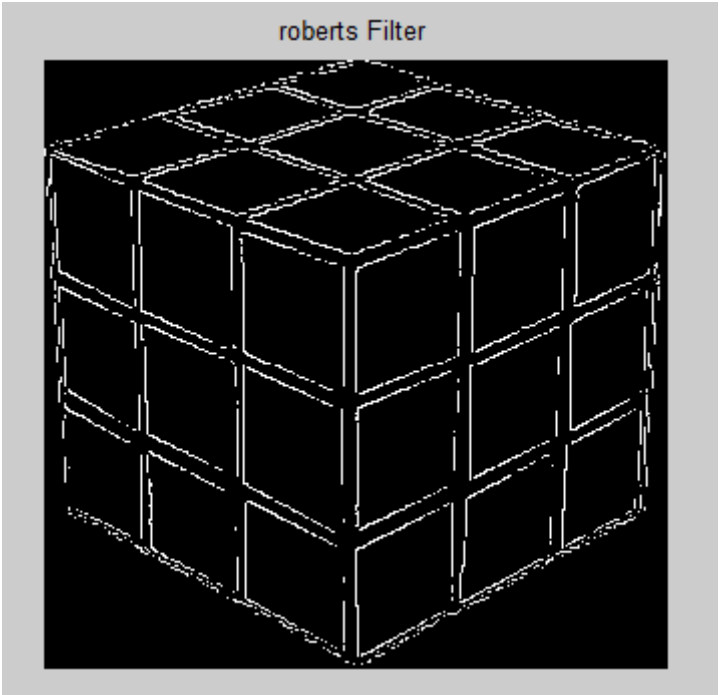
خروجی: کنی بهتر از همه لبه یابی را انجام می دهد چون تصویر متقارن است.




canny Filter


sobel Filter


prewitt Filter

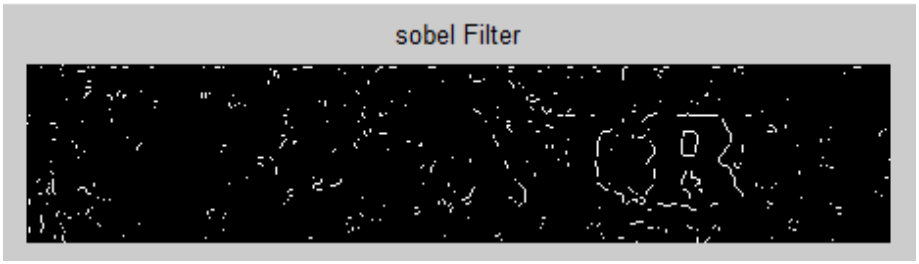

roberts Filter

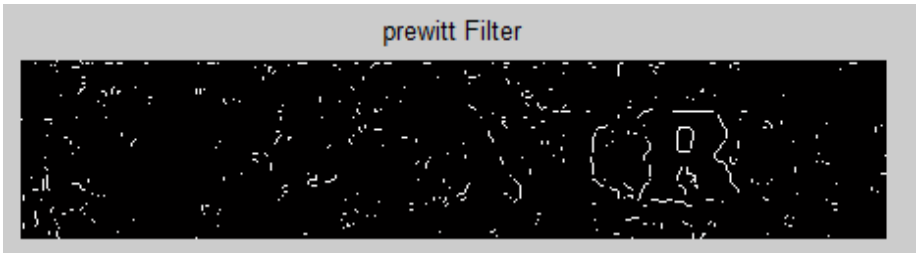
فایل UnSymmetrical.m

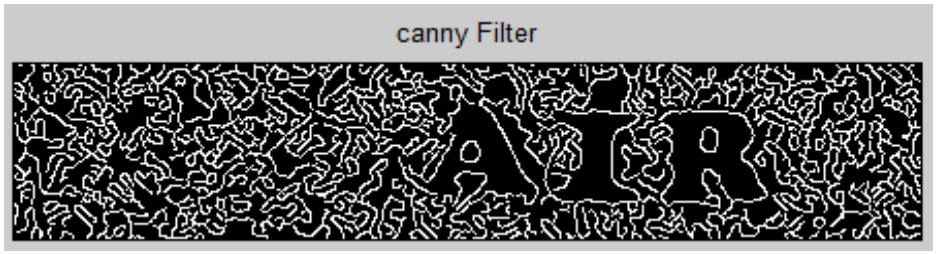
```
I = imread('C:\Users\Marjan\Documents\MATLAB\AIR.jpg');
m=rgb2gray(I);
figure,imshow(I);title('original Image');
figure,imshow(m);title('Converting RGB to grayscale');
y = edge(m,'prewitt');
figure,imshow(y);title('prewitt Filter');
z = edge(m,'roberts');
figure,imshow(z);title('roberts Filter');
w = edge(m,'sobel');
figure,imshow(w);title('sobel Filter');
x = edge(m,'canny');
figure,imshow(x);title('canny Filter');
```

با استفاده از دستور imread تصویر را می خواند و با استفاده از rgb2gray آن را به مقیاس خاکستری تبدیل می کند و با imshow آن را نمایش می دهد در یک figure و با یک عنوان به وسیله ی title. با دستور edge تصویر مقیاس خاکستری و عملگر لبه یابی موردنظر را گرفته و لبه یابی را روی آن اعمال می کند.

خروجی: کنی همه لبه ها را نمایش می دهد حتی لبه های چمن که مورد نیاز نیست، برای تصاویر نامتقارن الگوریتم های سنتی لبه یابی خوب جواب نمی دهد.







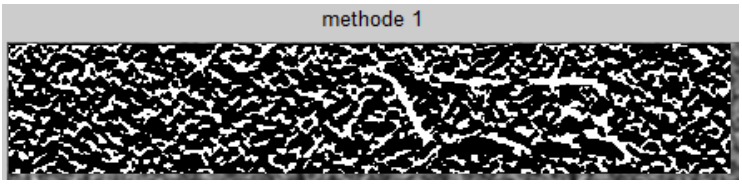
فایل methode1.m

```
m =
rgb2gray(imread('C:\Users\Marjan\Documents\MATLAB\AIR.j
pg'));
% figure
% imhist(m);
G = fspecial('gaussian',[10 10],1);
Ig = imfilter(m,G,'same');
figure,imshow(Ig);title('with gaussian');
[row,col]=size(Ig);
for y=2:col-6
    for x=2:row-6
```

```
        difference_of_intensity = abs(Ig(x,y) - Ig(x+1,
y))+abs(Ig(x,y) - Ig(x+2, y))+abs(Ig(x,y) - Ig(x+3,
y))+abs(Ig(x,y) - Ig(x+4, y))+abs(Ig(x,y) - Ig(x+5,
y));
        if difference_of_intensity < 120
            Ig(x,y) = 0;
        else
            Ig(x,y) = 255;
        end
    end
end
figure,imshow(Ig);title('methode1');
```

با استفاده از دستور imread تصویر را می خواند و با استفاده از rgb2gray آن را به مقیاس خاکستری تبدیل می کند و با fspecial ماسک گوسین با میانگین 1 و واریانس 10 می سازد و با imfilter ماسک ساخته شده را روی تصویر اعمال کرده و فیلتر گوسی روی تصویر اعمال می شود و با imshow آن را نمایش می دهد در یک figure و با یک عنوان به وسیله ی titleو با size اندازه تعداد پیکسل های سطر و ستون را مشخص کرده و در متغیرهای موردنظر ذخیره می کند سپس میانگین 5 پیکسل های همسایگی یا adjacent5pixels به کمک دو حلقه for اعمال می کند به این صورت که اختلاف شدت پیکسل با 5 پیکسل همسایگی بدست آمده و باهم جمع می شود و بامقدار آستانه مقایسه می شود و نهایتا سیاه یا سفید جایگزین شدت پیکسل فعلی میشود.

خروجی:

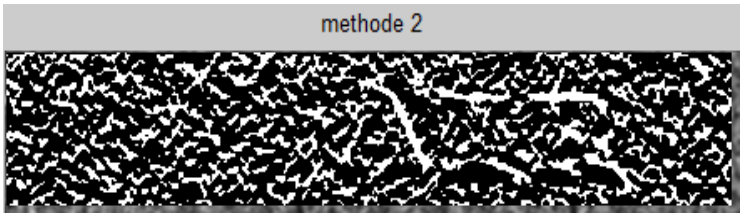


فایل methode2.m

```
m = imread('C:\Users\Marjan\Documents\MATLAB\AIR.jpg');
G = fspecial('gaussian',[10 10],1);
Ig = imfilter(m,G,'same');
figure,imshow(Ig);title('with gaussian');
g1=rgb2gray(Ig);
[row,col]=size(g1);
for y=2:col-6
    for x=2:row-6
        difference_of_intensity = abs(g1(x,y) - g1(x+1,
y))+abs(g1(x,y) - g1(x+2, y))+abs(g1(x,y) - g1(x+3,
y))+abs(g1(x,y) - g1(x+4, y))+abs(g1(x,y) - g1(x+5,
y));
        if difference_of_intensity < 120
            g1(x,y) = 0;
        else
            g1(x,y) = 255;
        end
    end
end
figure,imshow(g1);title('methode2');
```

دقیقا همان توضیحات بالا فقط ترتیب اجرا عوض می شود. اول فیلتر گوسی اعمال می شود و سپس به مقیاس خاکستری تبدیل می شود و نهایتا میانگین 5 پیکسل همسایگی گرفته می شود.

خروجی:

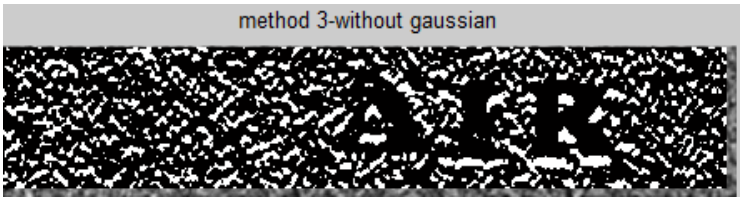


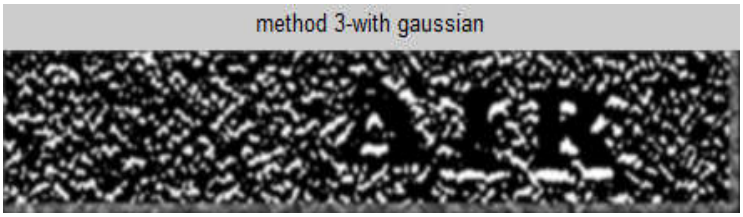
فایل methode3.m

```
m =
rgb2gray(imread('C:\Users\Marjan\Documents\MATLAB\AIR.j
pg'));
[row,col]=size(m);
for y=2:col-6
    for x=2:row-6
        difference_of_intensity = abs(m(x,y) - m(x+1,
y))+abs(m(x,y) - m(x+2, y))+abs(m(x,y) - m(x+3,
y))+abs(m(x,y) - m(x+4, y))+abs(m(x,y) - m(x+5, y));
        if difference_of_intensity < 120
            m(x,y) = 0;
        else
            m(x,y) = 255;
        end
    end
end
figure,imshow(m);title('method 3-without gaussian');
G = fspecial('gaussian',[10 10],1);
Ig = imfilter(m,G,'same');
figure,imshow(Ig);title('method 3-with gaussian');
J = imclearborder(Ig);
figure,imshow(J);title('with clear border image');
se=strel('disk',4);
img1 = imopen(J,se);
img2 = imerode(img1,se);
img3 = imclose(img2,se);
bw_image=im2bw(img3,0.1);
figure,imshow(bw_image);title('with erode');
y = edge(img3,'canny');
figure,imshow(y);title('Canny Filter');
```

ابتدا تصویر به مقیاس خاکستری تبدیل می شود و بعد آستانه گذاری روی آن اعمال می شود و نهایتا فیلتر گوسی اعمال می شود و با imclearborder الگوریتم حاشیه شفاف برای حذف پس زمینه به تصویر اضافه شده و سپس برای حذف نویز پس زمینه با strel دیسکی با شعاع 4 پیکسل و imopen تصویر باز می شود و سپس imerode فرسایش میابد و با imclose بسته می شود و با im2bw تصویر را باینری کرده و با edge(img3,'canny'); الگوریتم کنی را روی پیش پردازش اعمال می کنیم.

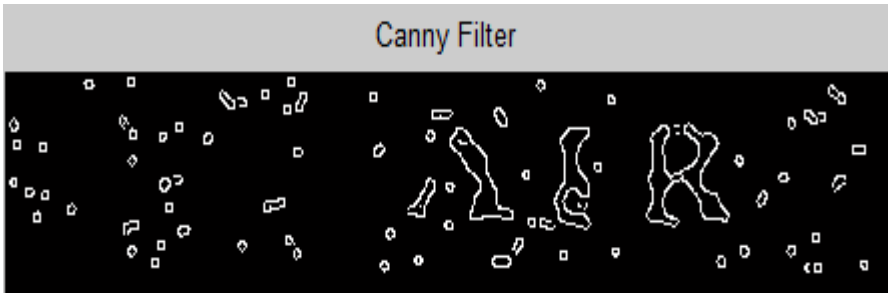
خروجی:


method 3-without gaussian


method 3-with gaussian


with clear border image


with erode


Canny Filter