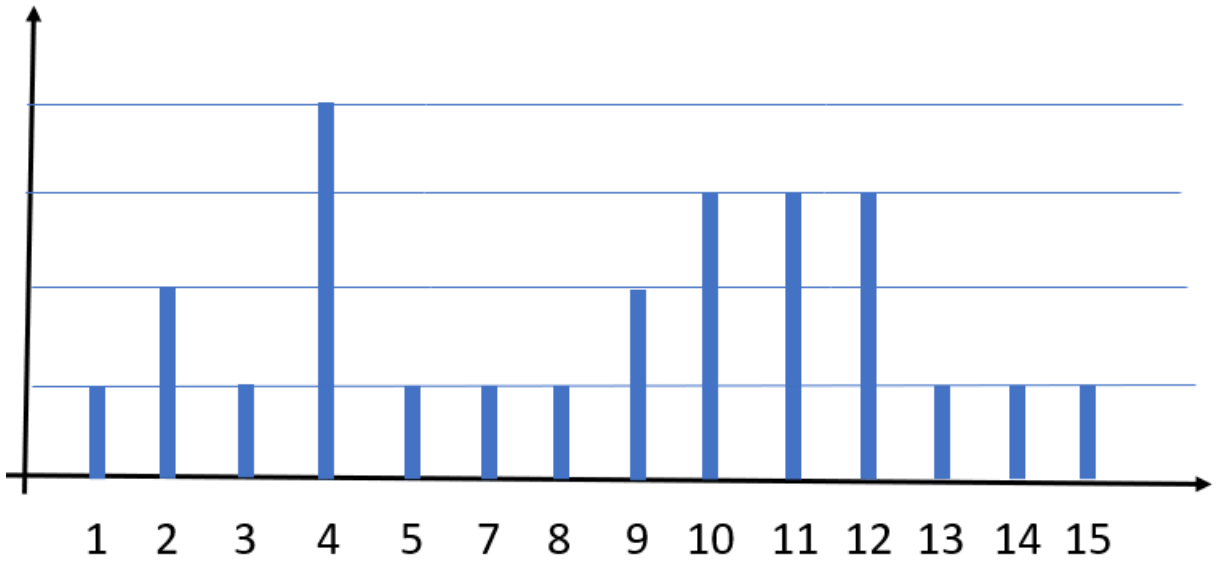
.1

الف)



AVRAGE = $\frac{1*1+2*2+3*1+4*4+5+7+8+9*2+10*3+11*3+12*3+13+14+15}{26}$ = 8.12

Median = 9

Mode = 4

Variance = $\sqrt{\frac{\sum(xi-\mu)^2}{N}}$ = $(1-8.12)^2$+2*$(2-8.12)^2$ +$(3-8.12)^2$+4 * $(4-8.12)^2$ + $(5-8.12)^2$+ $(7-8.12)^2$+$(8-8.12)^2$+2*$(9-8.12)^2$+3*$(10-8.12)^2$+3*$(11-8.12)^2$+3*$(12-8.12)^2$+$(13-8.12)^2$+$(14-8.12)^2$+$(15-8.12)^2$ = $\sqrt{280} = 16.74$

ب)

Threshold = 9.5

W1 = 13/25 = 0.52

Var1 = 6.63 Var2 = 2.35

Otsu = w1 * var1 + (1-w1)*var2 = 4.58

Threshold = 11.5

W1 = 19/25 = 0.76

Var1 = 11.71 Var2 = 1.33

Otsu = w1 * var1 + (1-w1)*var2 = 9.22

میدانیم هرچقدر مقدار otsu کمتر باشد پس سطح استانه بهتری داریم و براساس مقادیر بالا threshold = 9.5 بهتر است.

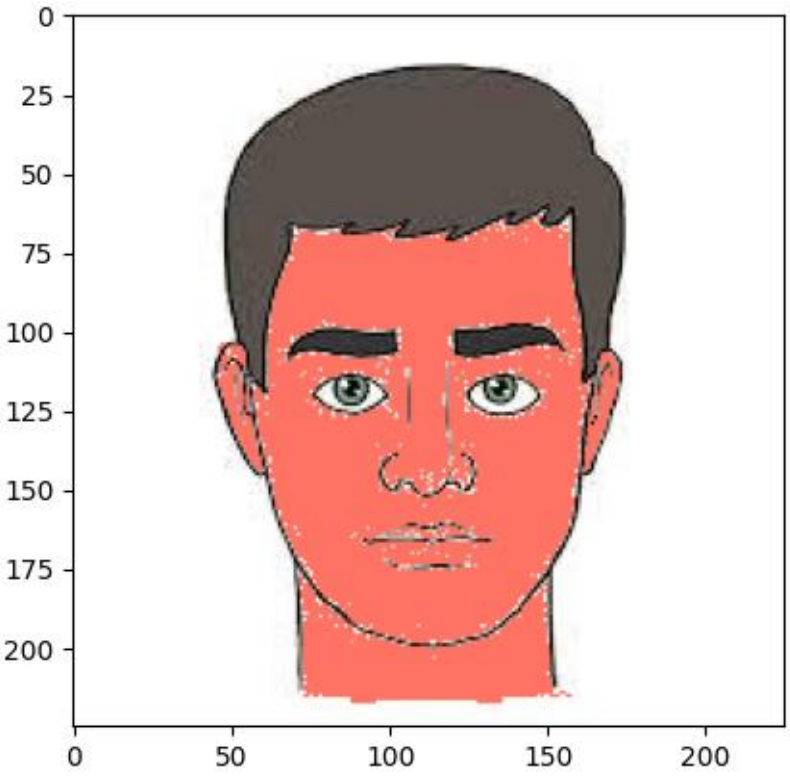
الف) سرعت روش gaussian otsu بیشتر از otsu است طبق متن بالا به دلیل اینکه محاسبات بیشتری دارد. همچنین دقت روش gaussian otsu به دلیل بررسی تفاوت کلاس پس زمینه و جلوی تصویر بیشتر از روش otsu است.

ب) خیر با بیشینه کردن واریانس بین کلاسی میخواهیم بین کلاس ها تفاوت ایجاد کنیم و در واقع تصویر جلو بهتر دیده شود تا تصویر پس زمینه.

با کمینه کردن واریانس درون کلاسی هم میخواهیم اجزای داخل کلاس تفاوت کمتری داشته باشند.

.3

در این قسمت ابتدا چند نقطه به عنوان seed در نظر میگیریم.(نقاط بینی و گوش ها و گردن) سپس به ازای هر یک از این نقاط رنگ تمام عکس را از مقدار این نقطه کم میکنیم با این کار نقاطی که رنگشان نزدیک به نقطه ی seed است به سمت سیاهی میروند. سپس عکس را باینری کرده با استفاده از یک threshold و تابع cv2.threshold() . بعد از آن با شروع از نقطه ی seed روی عکس dfs میزنیم. نقاط همسایه ی هر نقطه شامل 8 نقطه ی اطرافش که در یک کرنل 3 در 3 قرار میگیرند است. اگر این نقاط 0 بودند(در باینری کردن 0 شدند چون نزدیک به سیاه و در نتیجه نزدیک به رنگ نقطه seed بوده) و اگر قبلا دیده نشده بودند در استک اضافه می شود. خروجی:



4.الف)

سایش:

60	60	60	60	60	60	60	60
60	60	60	60	60	60	60	60
60	60	70	60	70	70	60	60
60	60	60	60	60	70	70	70
60	60	60	60	60	70	60	60
60	60	60	60	60	60	60	60
60	60	60	60	60	60	60	60
60	60	60	60	60	60	60	60

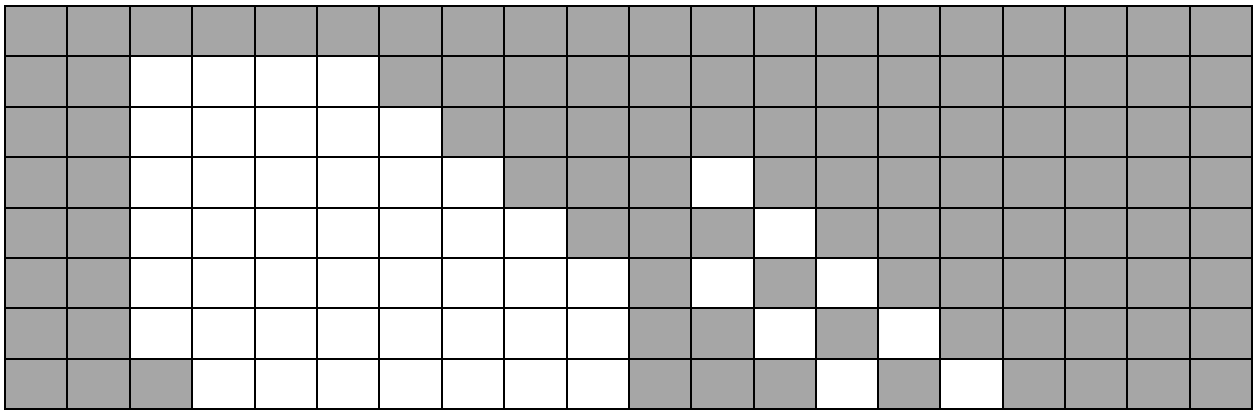
گسترش:

70	70	70	70	70	70	70	70
70	70	70	70	70	70	70	70
80	80	80	80	80	80	80	70
70	80	70	70	80	70	70	70
70	80	80	80	80	80	70	60
70	80	80	80	80	80	80	60
70	80	80	80	80	80	80	70
70	70	80	80	80	80	80	70

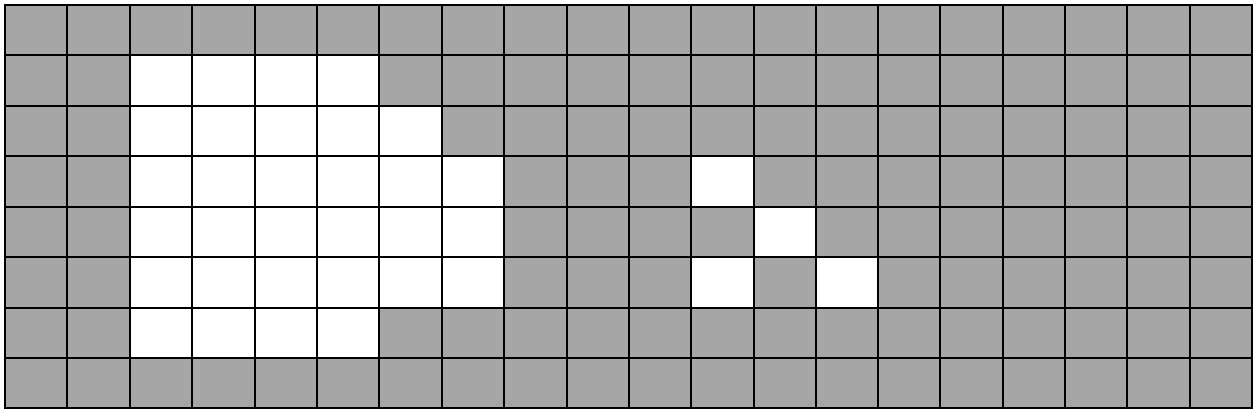
ب)

Closing:

dilate

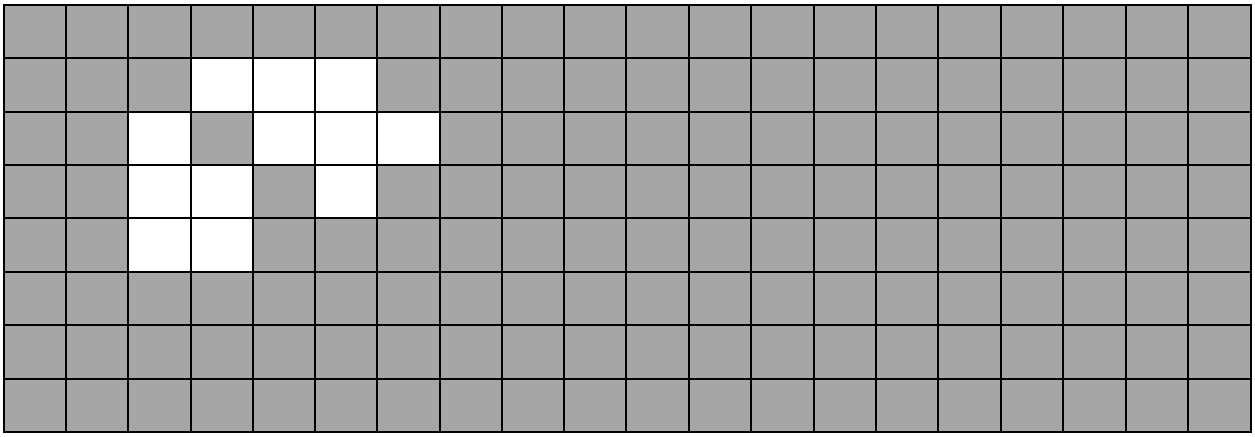


Erode

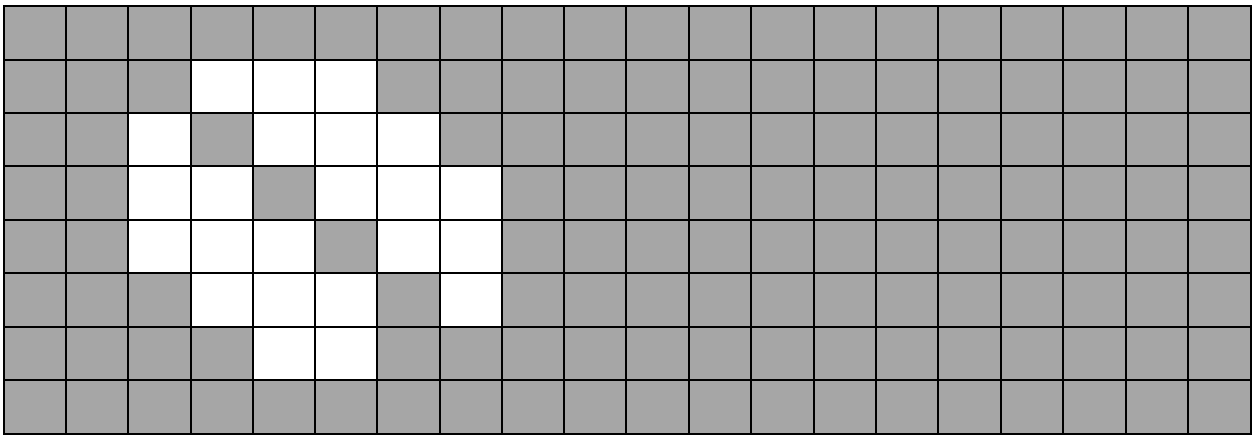


opening:

erode



Dilate



5.الف)از این 4 کرنل برای تایین مرز استفاده میکنید که نشان دهنده مرز ها در 4 جهت است.

0	-1	0
0	1	0
0	0	0

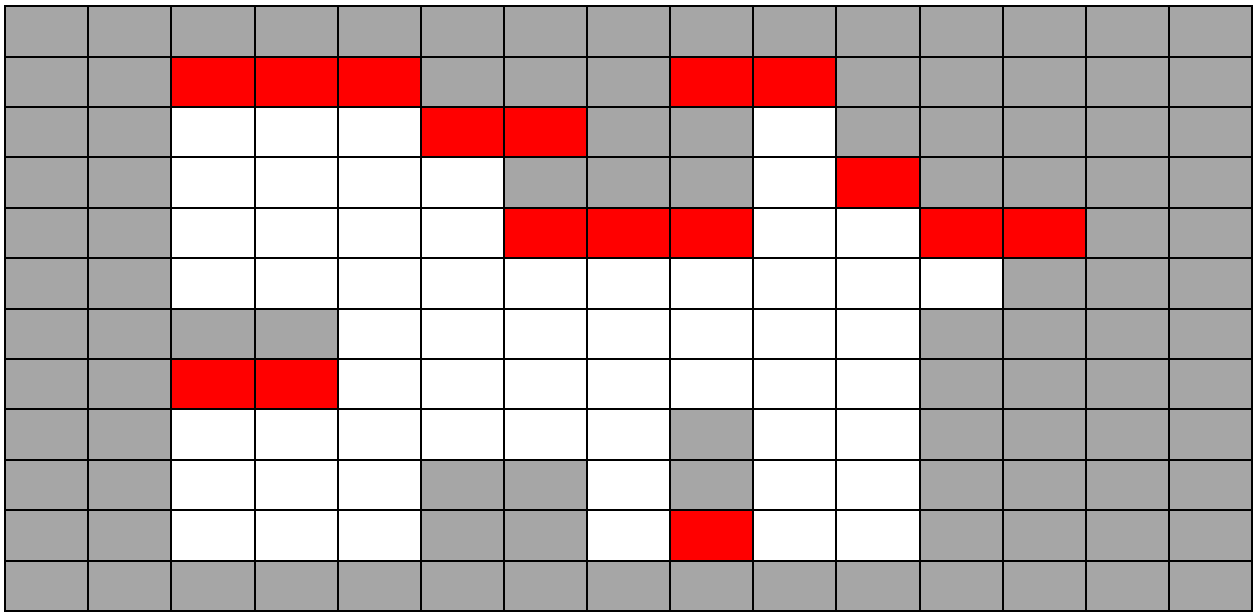
0	0	0
-1	1	0
0	0	0

0	0	0
0	1	-1
0	0	0

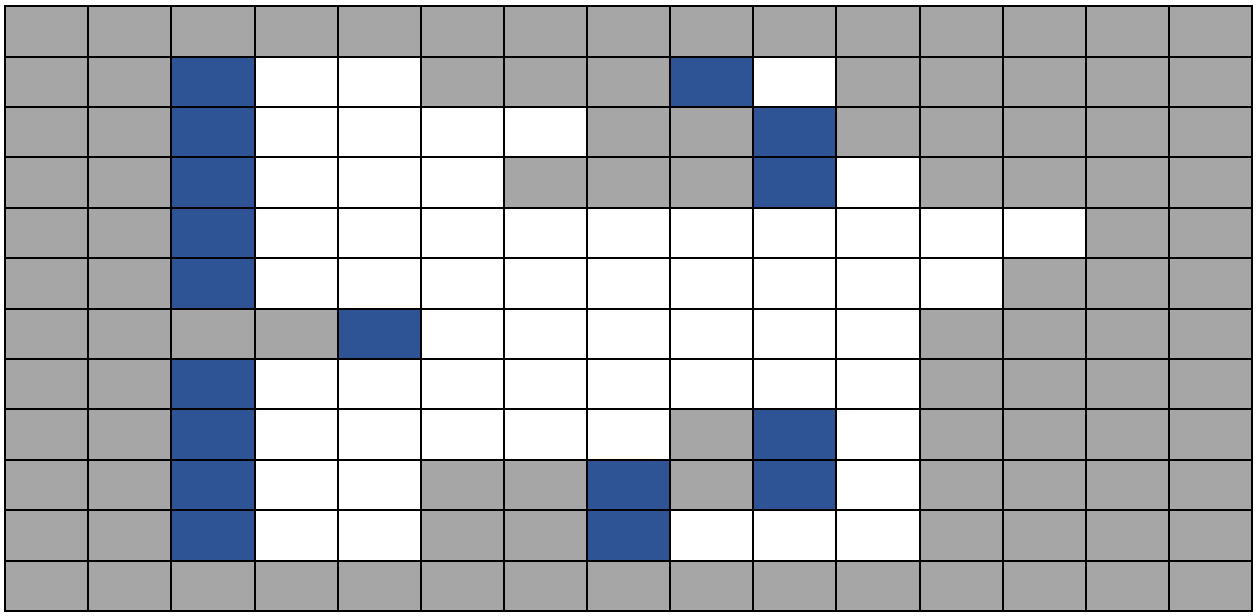
0	0	0
0	1	0
0	-1	0

برای کرنل ها به ترتیب از سمت راست داریم:

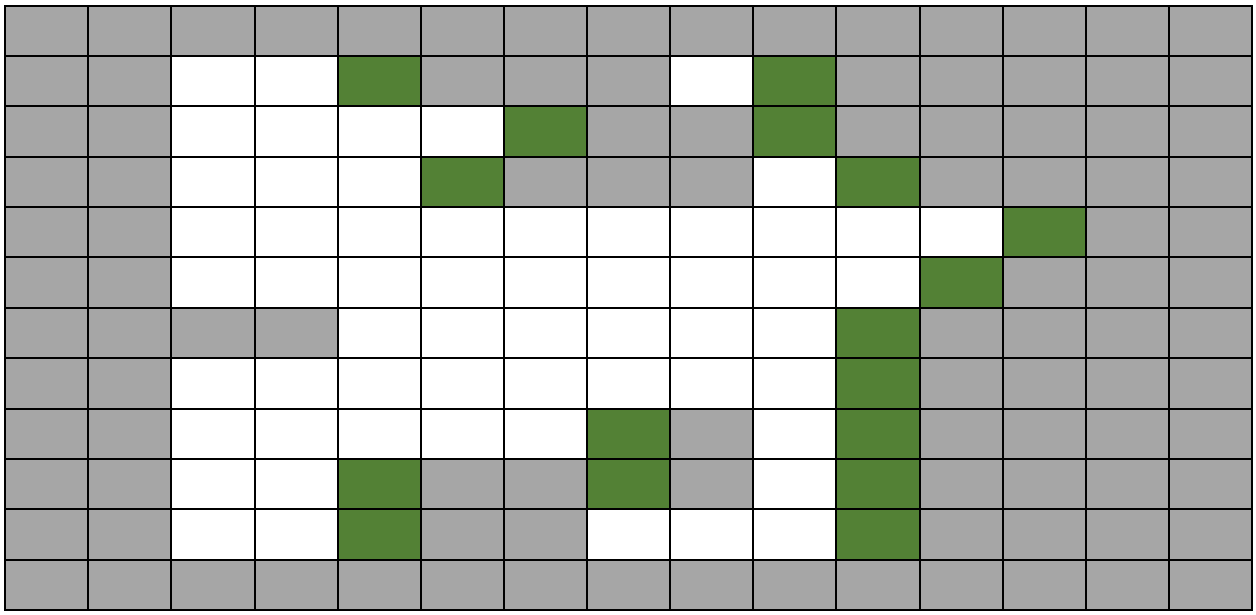
لبه بالا:



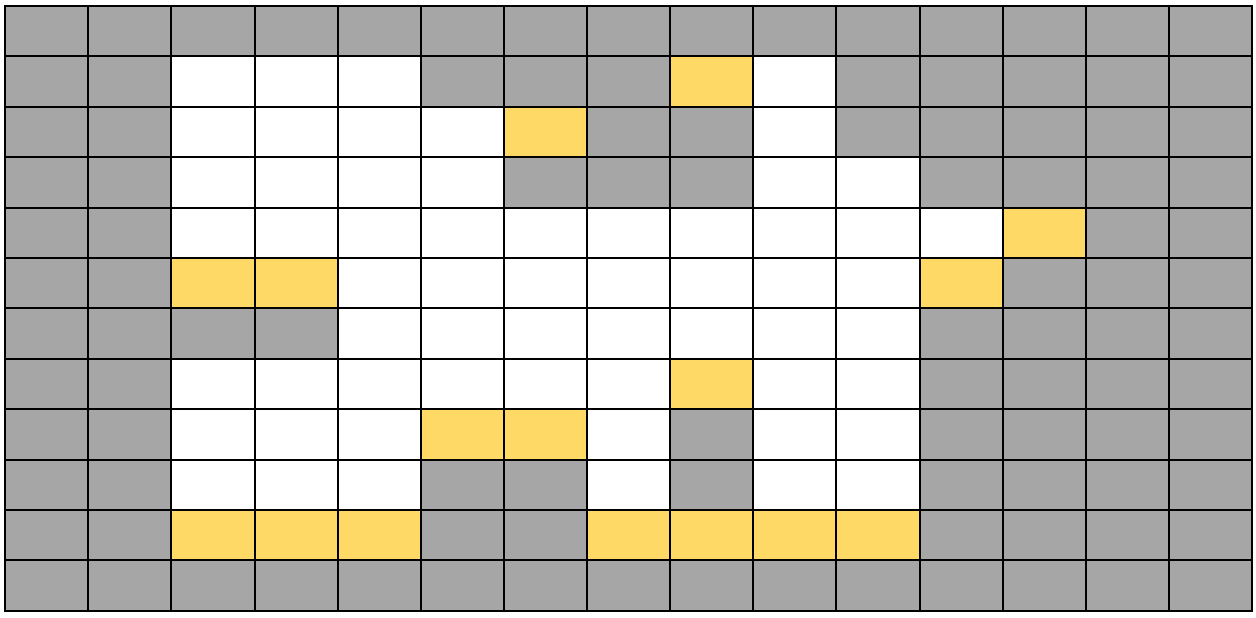
لبه چپ:



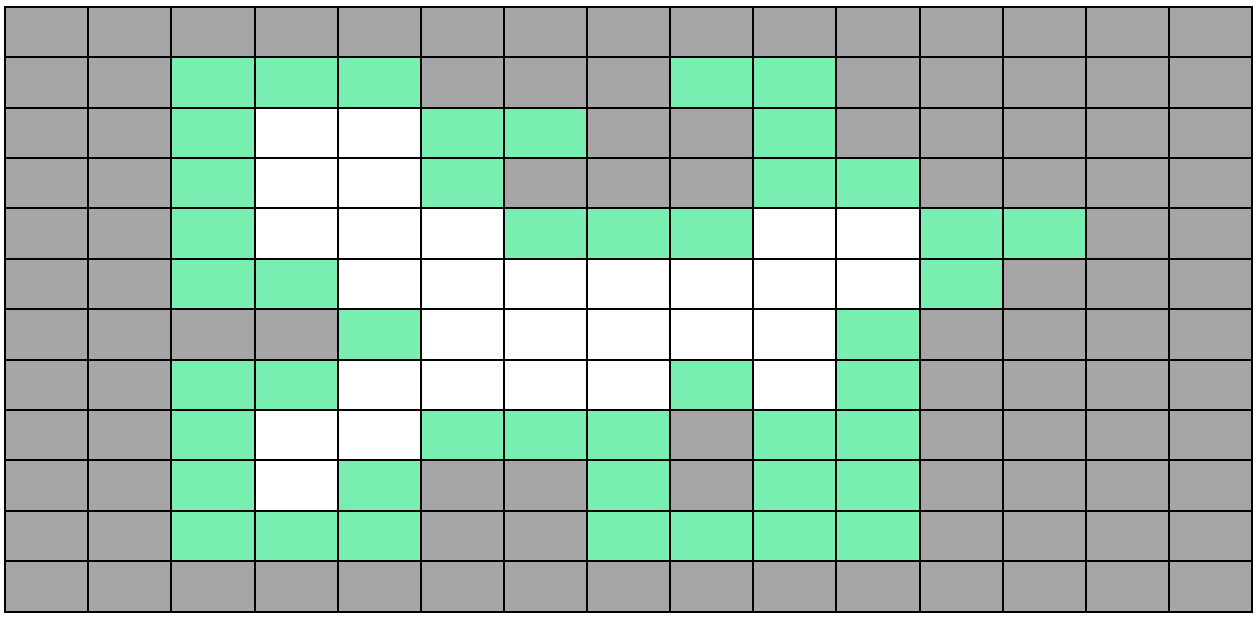
لبه راست:



لبه پایین:

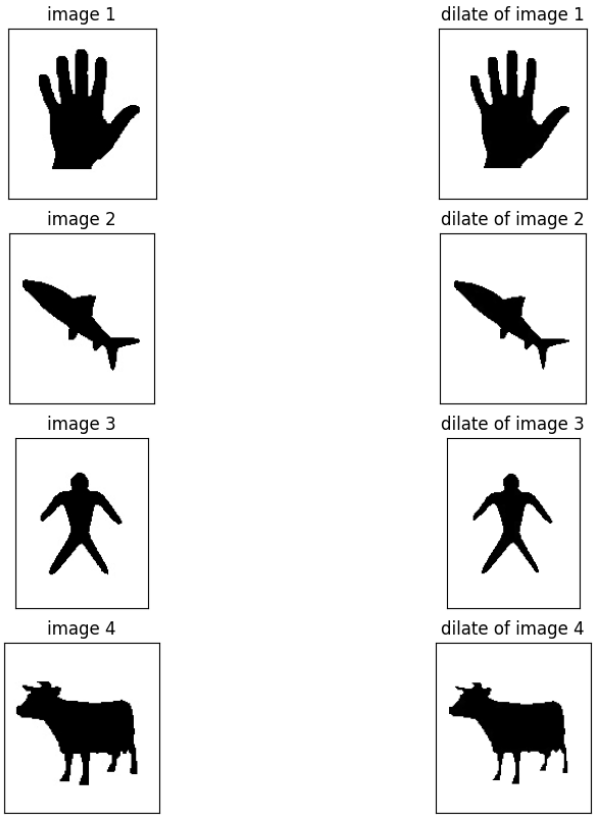


از همه ی موارد بالا اجتماع میگیریم :

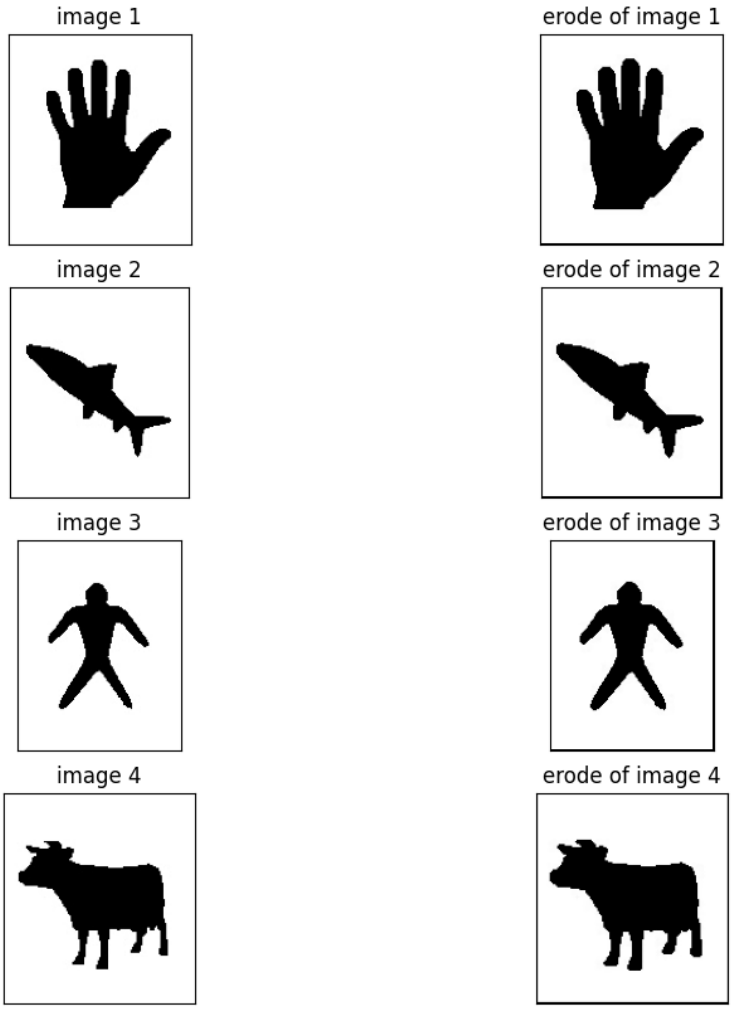


.6

الف) در اینجا برای عملگر گسترس بعد از اضافه کردن پدینگ (1)، کرنل را در قسمتی از عکس که میخواهیم گسترس بزنیم ضرب میکنیم این کار نقاطی که 0 اند 0 میماند و اگر 1 داشته باشیم و عکس هم مقدار 1 داشته باشد در اخر مقدار پیکسل با ماکسیمم گرفتن 1 میشود.



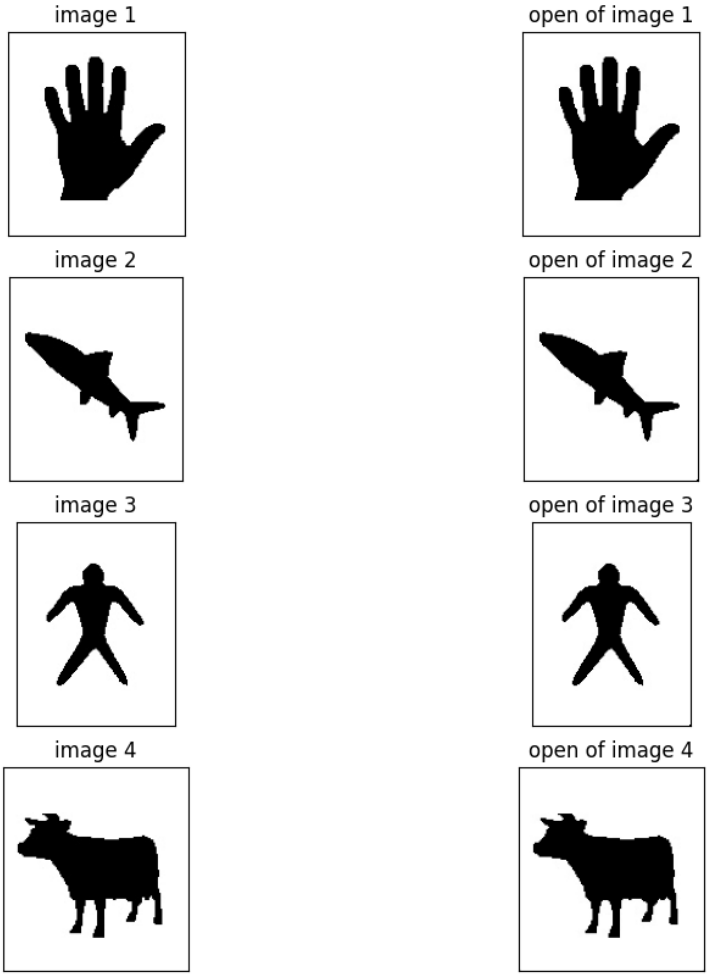
برای عملگر سایش بعد از اضافه کردن پدینگ هر بار که کرنل را روی عکس قرار میدهیم چک میکنیم اگر مقدار کرنل 1 بود مقدار عکس ذخیره شود و در آخر از همه ی مقادیر مینیمم میگیریم.



در عملگر باز مشابه عبارت زیر

$$A \circ B = (A \ominus B) \oplus B$$

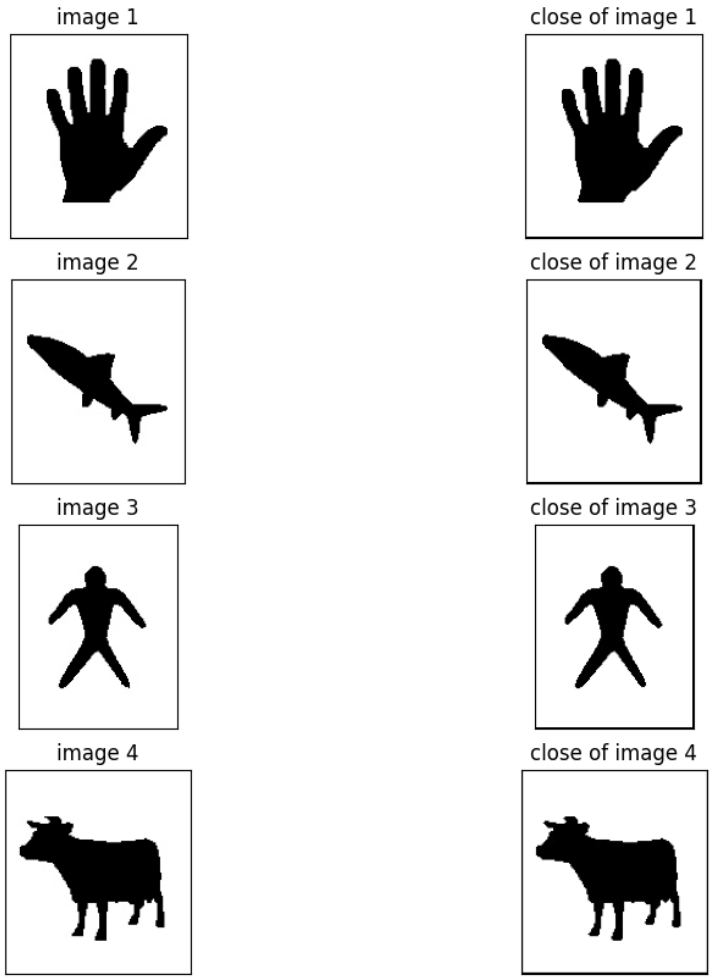
ابتدا عملگر سایش و سپس گسترس میزنیم.



در عملگر بسته مشابه عبارت زیر

$$A \cdot B = (A \oplus B) \ominus B$$

ابتدا عملگر گسترس و سپس سایش میزنی.



ب) تابع dilate و erode:

ورودی اول عکس را میگیرد بعد کرنل(عنصر ساختاری) و در iterations هم تعداد مراحلی که سایش یا گسترش انجام می شود.

تابع morphologyEx:

ورودی اول عکس بعد اینکه عملگر باز یا بسته است (MORPH_CLOSE , MORPH_OPEN) و بعد از آن کرنل (عنصر ساختاری)را میگیرد.

خروجی را هم در فایل ارسالی میتوانید ببینند.