lab10_HW

1. Improve the GUI program you designed in Lab 9 by adding `"filter"` and `"opening"` functions. In `"09rice.png"`, there exist `"sparkles"` (small dots that are actually not grains) in the image that are falsely recognized as grains. See the number of very small grains in the histogram. Include a button in your GUI program with which the users are able to remove the sparkles. You may also let the users determine the sizes of sparkles to be removed. In addition, there exist connected grains in the image `"09rice.png"`. The connected grains are recognized as one grain by the connected labeling algorithm. See the number of very large grains in the histogram. Implement the morphological opening to separate connected grains.

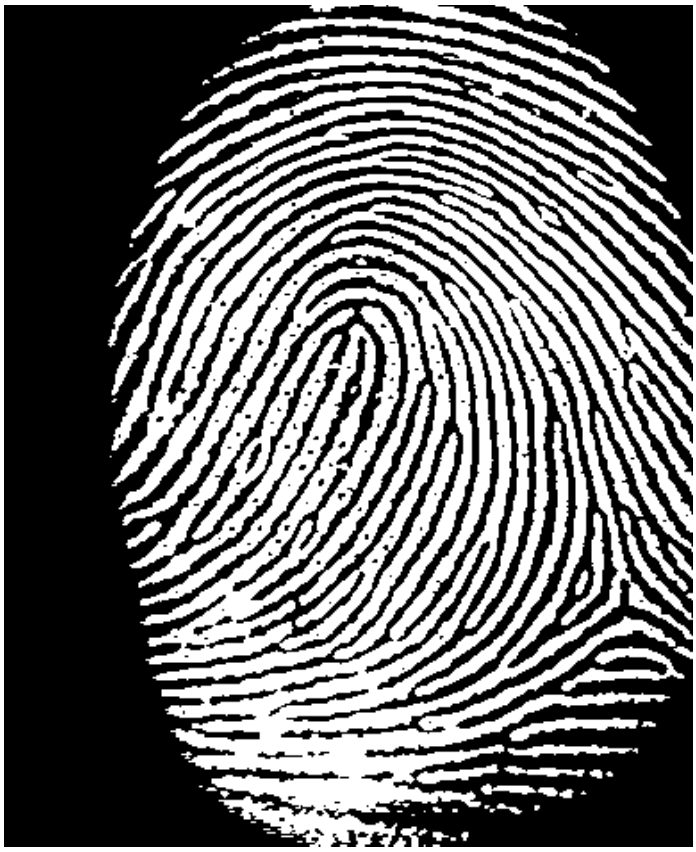
> I did it at Lab09
> go to see Lab09-Question2 (https://hackmd.io/s/r1HKmk0Bb)

2. Implement a serious of image processing algorithms to improve the quality of `"10Fingerprint.tif"`. If interested in, you may further move to perform eature extraction on fingerprints. Typical features for fingerprints include ridge ending, ifurcation, and short ridge. To extract these features, you will need to use thinning, a morphological algorithm that identifies the skeletons of foreground objects. Check MATLAB function `bwmorph`.

If you don’t know what is ridge ending ifuractoin and shor ridge
see https://en.wikipedia.org/wiki/Fingerprint_recognition#Minutiae_features
(https://en.wikipedia.org/wiki/Fingerprint_recognition#Minutiae_features)

First remove ugly background

```
>> img = imread('10Fingerprint.tif');
>> bwimg = ~imbinarize(img);
```



Good. almost no background and inverse to make our object white, now

remove small dot inside print

beacuse there are some important thing smaller than those small dot, so you cannot use average, or dilation.
Remove small connect component on finger print

```
nobimg = ~bwimg;
for i = 1:2
    nobimg = bwareaopen(nobimg, 15, 4);
end
nobimg = ~nobimg;
```


closing

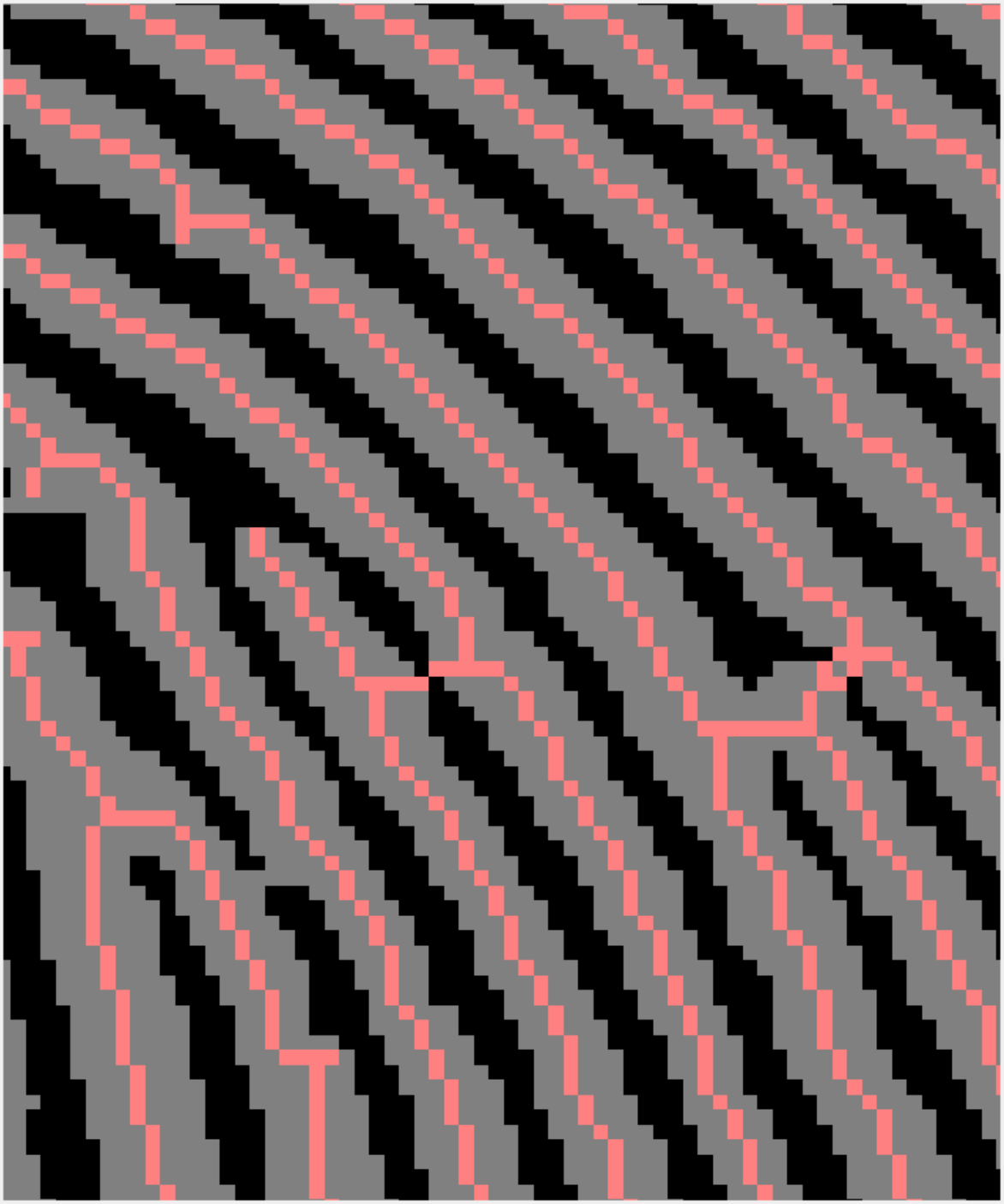
now go erosion and remove small component again
and dilate back small bridge and small dot disappear

```
eroimg = imerode(nobimg, strel('disk', 1));
eroimg = bwareaopen(eroimg, 10, 4);
eroimg = imdilate(eroimg, strel('disk', 1));
```

get the skeleton

now the **most important** thing of all

use `skelimg = bwmorph(eroimg,'skel', Inf);`



see looks very good
but this function has some disadvantage

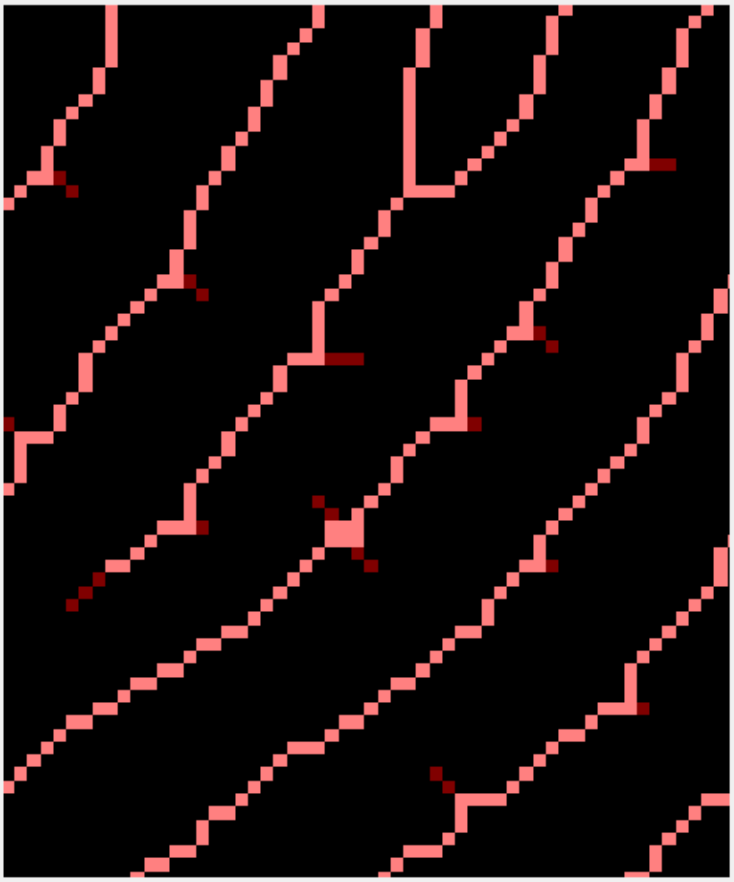
remove error branch

we need to remove some bad branch on it

however you should not use `shrink`

it will remove this feature

```
cutskelimg = bwmorph(skelimg, 'diag');
cutskelimg = bwmorph(cutskelimg, 'spur', 10);
cutskelimg = bwmorph(cutskelimg, 'skel', Inf);
cutskelimg = bwmorph(cutskelimg, 'spur', 3);
```



(remove red point)

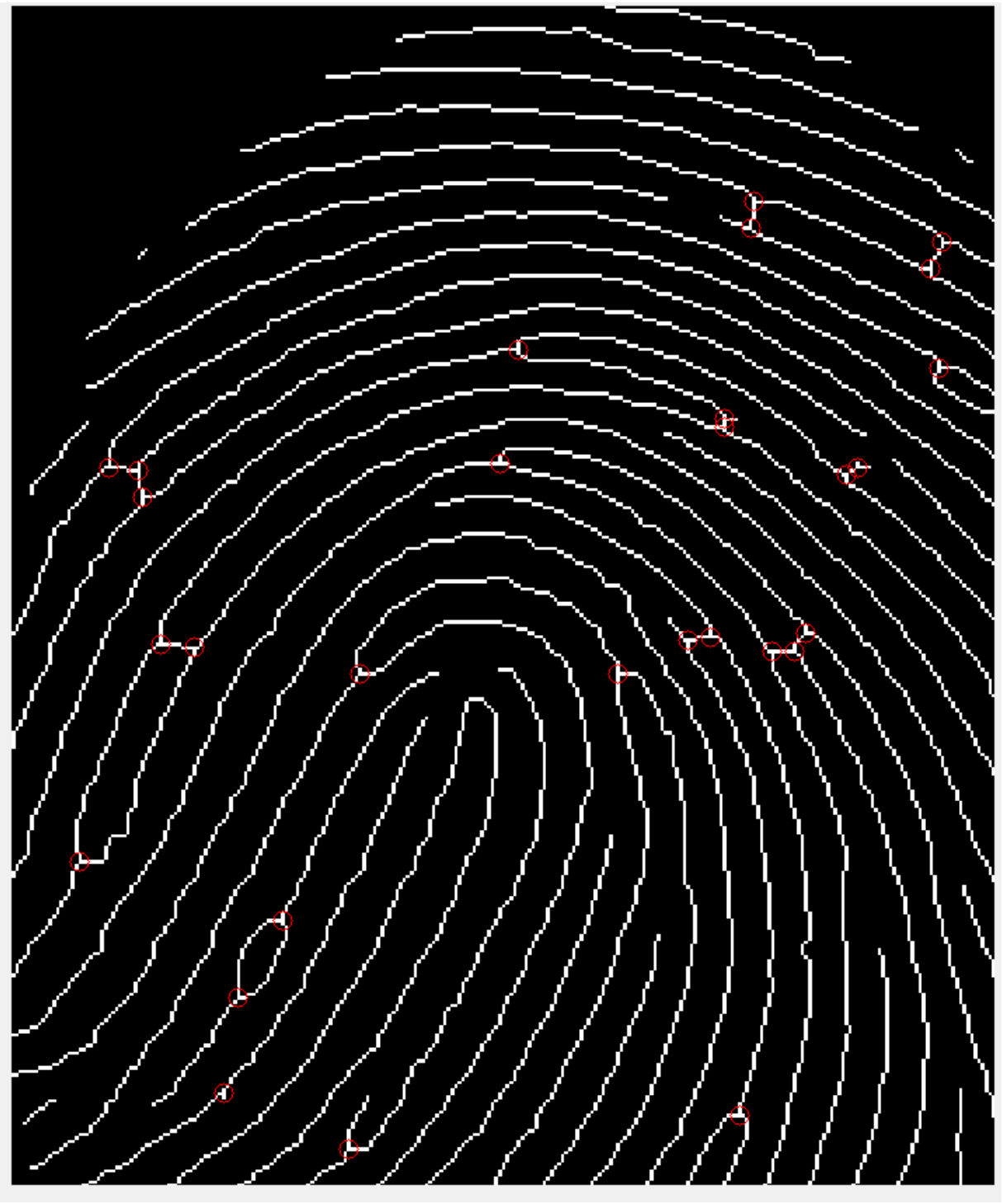
remove L shape

when we use 8-connect to determinate where is branch
so L shape is annoying

```
interval = [-1  1  0
             1  1 -1
             0 -1 -1];
noLimg = cutskelimg;
for ang = 0:90:270
    noLimg = noLimg - bwhitmiss(noLimg, imrotate(interval,  ang));
end
```

get the branch

```
imshow(noLimg);
branch = bwmorph(noLimg, 'branchpoints');
hold on
[y, x] = find(branch);
plot(x, y, 'ro');
hold off
```



Great!

get the endpoint

just change the code
`branchpoints` to `endpoints` above

clear H bridge on it

we can find small H bridge between two branchs which is very close to each other
find near bridge

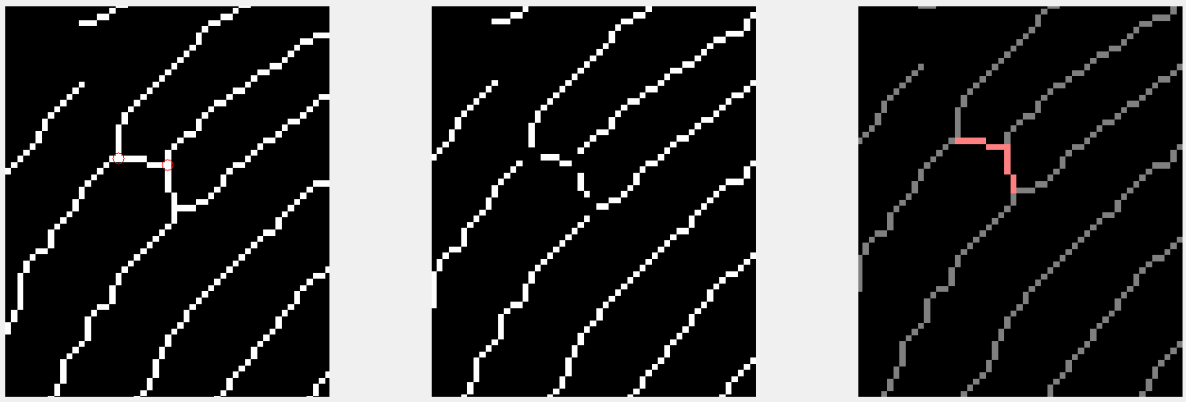
```
dis = (y - y(i)) .^ 2 + (x - x(i)) .^ 2;
near = find(dis > 16 & dis < 100);
```

erase branch point and label on it

```
tmpbgimg = nobgimg;
for p = 1:size(pair, 1)
    tmpbgimg(y(pair(p,1))-1:y(pair(p,1))+1, x(pair(p,1))-1:x(pair(p,1))+
    tmpbgimg(y(pair(p,2))-1:y(pair(p,2))+1, x(pair(p,2))-1:x(pair(p,2))+
end
tmplbimg = bwlabel(tmpbgimg);
bgimg = zeros(size(nobgimg), 'logical');
```

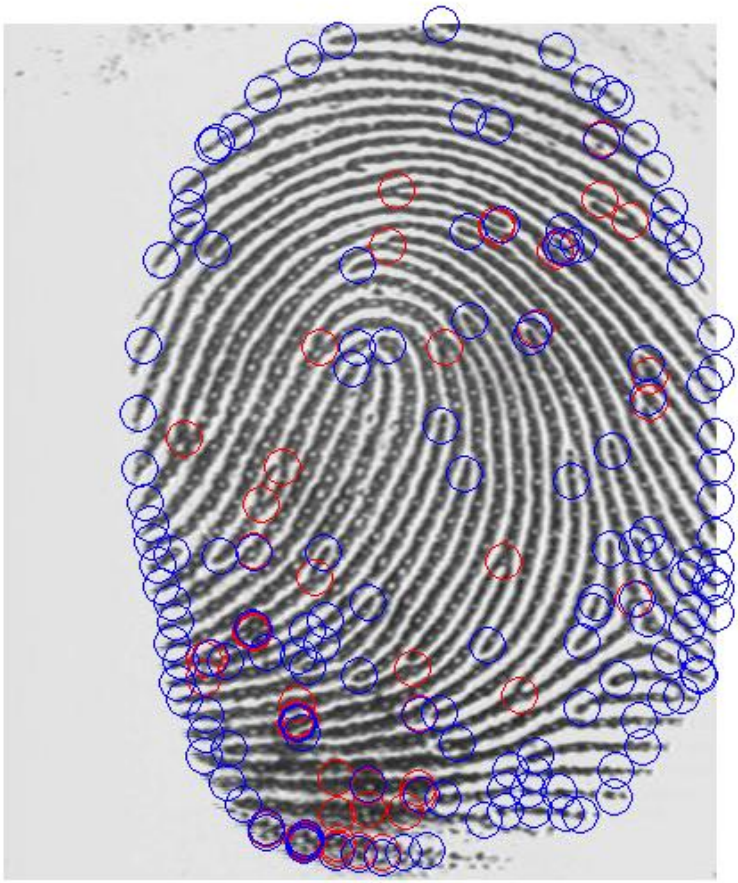
extract the small bridge by

```
if nnz(tmplbimg == mid) < 20
    bgimg = bgimg + (tmplbimg == mid);
end
bgimg = nobgimg & imdilate(bgimg, strel('disk', 2));
nobgimg = nobgimg - bgimg;
```



Finally

```
imshow(img);
ends = bwmorph(noLbgimg, 'endpoints');
branchs = bwmorph(noLbgimg, 'branchpoints');
hold on
[y, x] = find(branchs);
plot(x, y, 'ro', 'MarkerSize', 15);
[y, x] = find(ends);
plot(x, y, 'bo', 'MarkerSize', 15);
hold off
```



update 7/29

remove branch very carefully

I found that skeleton algorithm is very powerful
use it with a little pre-process is enough
methods same to remove bridge writing above
if endpoint is too close to brnach point, remove it

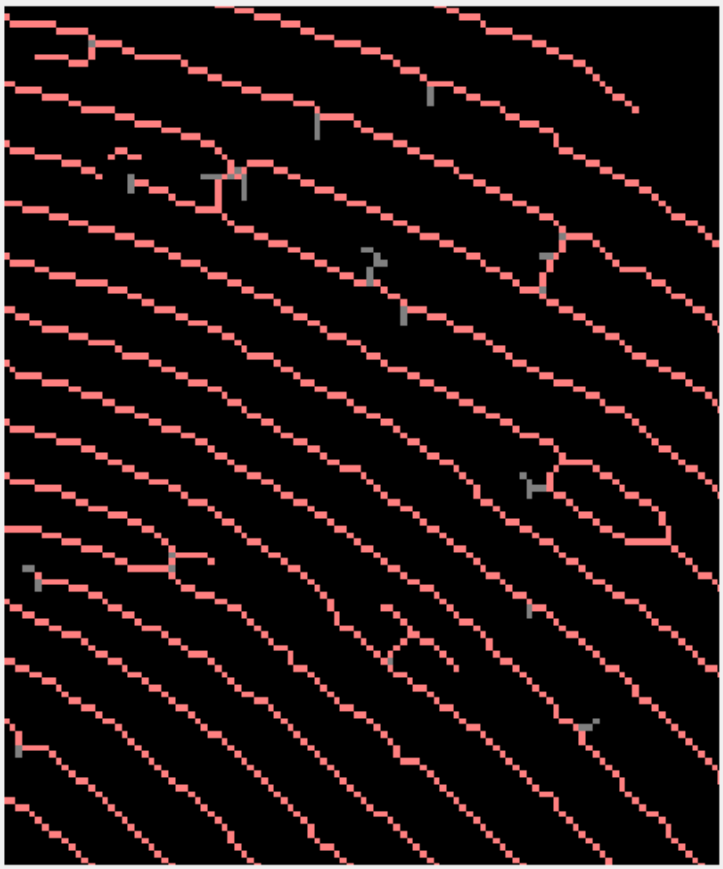
- pre-process

```
img = imread('10Fingerprint.tif');
bwimg = ~imbinarize(img);
nobimg = ~bwimg;
for i = 1:2
    nobimg = bwareaopen(nobimg, 15, 4);
end
nobimg = ~nobimg;
skelimg = bwmorph(nobimg, 'skel', Inf);
```

- remove branch
  same method described before

```
nobrimg = cutskelimg;
nobrimg = cutBranch(nobrimg);
nobrimg = cutBranch(nobrimg);
nobrskelimg = bwmorph(nobrimg, 'skel', Inf);
nobrskelimg = bwmorph(bwmorph(nobrskelimg, 'diag'), 'skel', Inf);
```

see `fingerprint_cleaner#cutBranch`



- remove bridge
  after remove bridge, remove branch

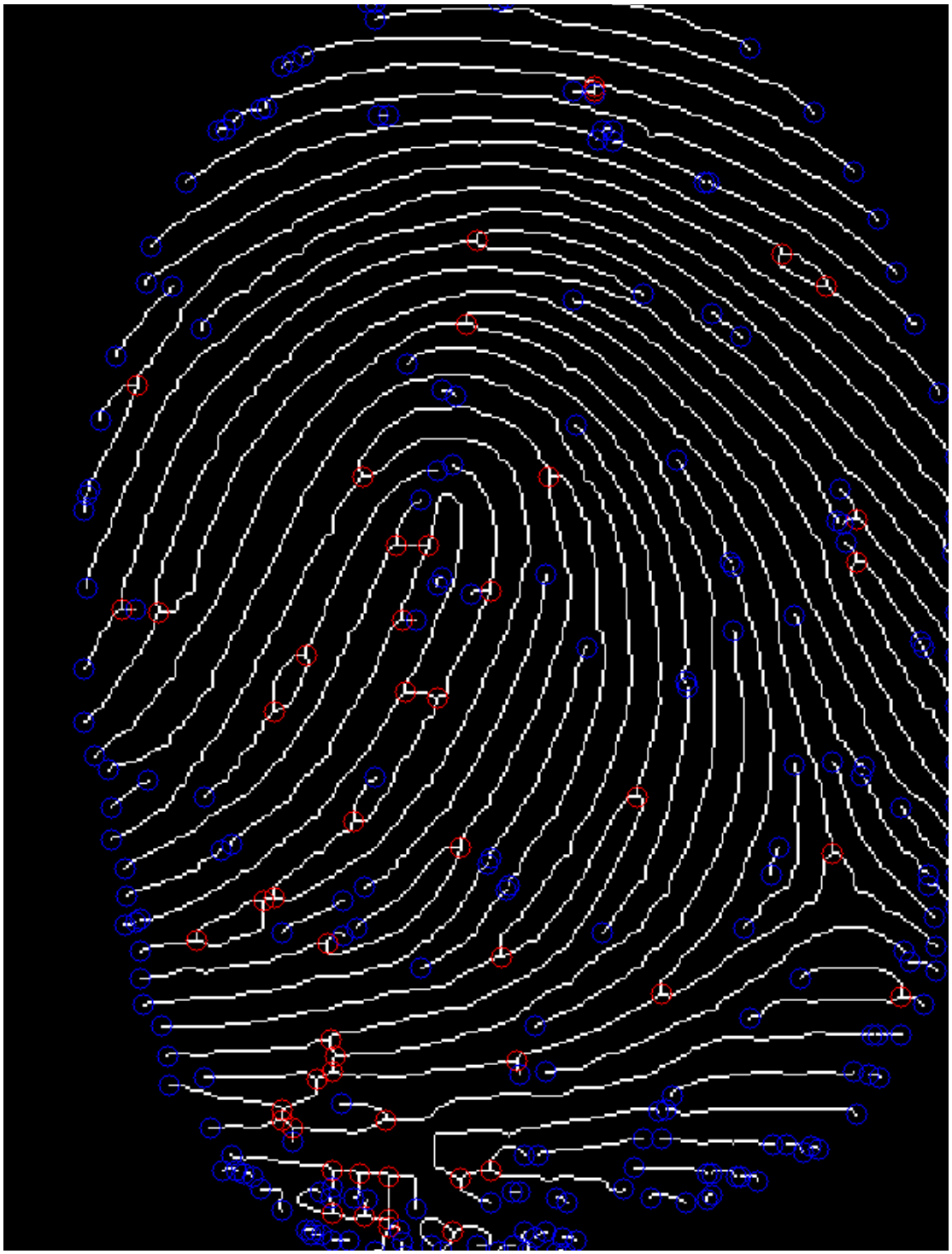
```
nobgimg = nobrskelimg;
nobgimg = cutBridge(nobgimg);
nobgbrimg = cutBranch(nobgimg);
nobgbrimg = cutBranch(nobgbrimg);
nobgskelimg = bwmorph(nobgbrimg, 'skel', Inf);
nobgskelimg = bwmorph(bwmorph(nobgskelimg, 'diag'), 'skel', Inf);
```

- connect if I acciendenitly break line

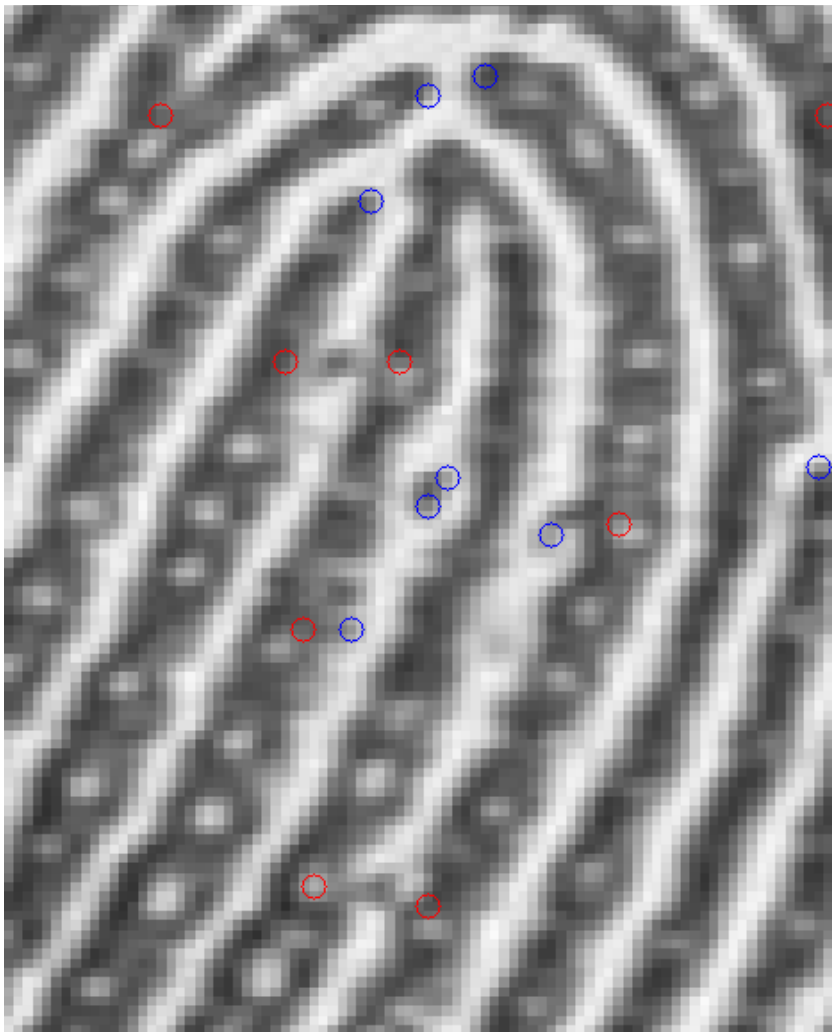
```
conimg = bwmorph(nobgskelimg, 'bridge');
```

perfect

now it is really perfect



see this no bridge
this bridge is caused by bad pre-processing or
the data is not very clean enough



code

see `fingerprint_cleaner.m`

```
>> fingerprint_cleaner('10Fingerprint.tif');
```

star me on Github
linnil1 (https://github.com/linnil1/Lab304_2017summer)

Test on ubuntu16.04 + Matlab R0217a academic use