**import** numpy **as** np

**from** PIL **import** Image

**import** time

**from** matplotlib **import** pyplot **as** plt

**import** cv2

*#from shapedetector import ShapeDetector*

**import** argparse

**import** imutils

In [ ]:

str1**=**'20170925\_20171112.geo.unw' *# lines betwen subs*

In [ ]:

img **=** Image**.**open(str1 **+** '.png' )

data **=** np**.**array(img, dtype**=**'uint8')

image **=** cv2**.**imread(str1 **+** '.png')

In [ ]:

print(data**.**shape, image**.**shape)

plt**.**figure(figsize**=**(8, 8))

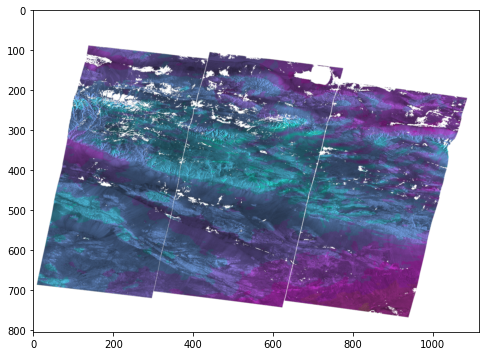
plt**.**imshow(data)

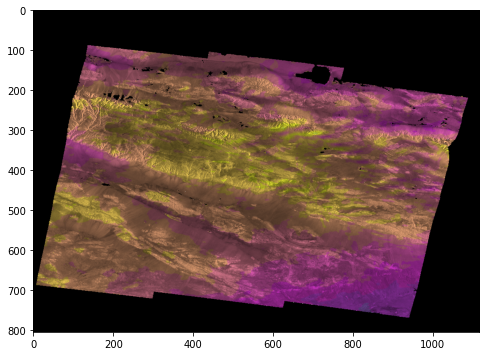
plt**.**figure(figsize**=**(8, 8))

plt**.**imshow(image)

plt**.**show()

(805, 1117, 4) (805, 1117, 3)





In [ ]:

a**=**data[:,:,3]

*#a= cv2.cvtColor(image,cv2.COLOR\_BGR2GRAY)*

a**=**255**-**a

thresh **=** cv2**.**threshold(a, 10, 255, cv2**.**THRESH\_BINARY)[1]

*#thresh = cv2.adaptiveThreshold(a,255,cv2.ADAPTIVE\_THRESH\_MEAN\_C,cv2.THRESH\_BINARY,3,2)*

*#kernel=cv2.getStructuringElement(cv2.MORPH\_RECT,(15,15))*

*#thresh = cv2.morphologyEx(thresh, cv2.MORPH\_CLOSE, kernel)*

In [ ]:

*###### line detection*

low\_threshold **=** 50

high\_threshold **=** 150

kernel**=**5

edges **=** cv2**.**Canny(thresh, low\_threshold, high\_threshold,kernel)

*#laplacian = cv2.Laplacian(thresh,cv2.CV\_64F)*

In [ ]:

*####### Hough line detection*

rho **=** 1 *# distance resolution in pixels of the Hough grid*

theta **=** np**.**pi **/** 180 *# angular resolution in radians of the Hough grid*

threshold **=** 10 *# minimum number of votes (intersections in Hough grid cell)*

min\_line\_length **=** 100 *# minimum number of pixels making up a line*

max\_line\_gap **=** 20 *# maximum gap in pixels between connectable line segments*

line\_image **=** np**.**copy(image) **\*** 0 *# creating a blank to draw lines on*

In [ ]:

*# Run Hough on edge detected image*

*# Output "lines" is an array containing endpoints of detected line segments*

lines **=** cv2**.**HoughLinesP(edges, rho, theta, threshold, np**.**array([]),

min\_line\_length, max\_line\_gap)

**for** line **in** lines:

**for** x1,y1,x2,y2 **in** line:

cv2**.**line(line\_image,(x1,y1),(x2,y2),(255,0,0),5)

lines\_edges **=** cv2**.**addWeighted(image, 0.8, line\_image, 1, 0)

In [ ]:

plt**.**figure()

plt**.**imshow(a,cmap**=**'gray')

plt**.**figure()

plt**.**imshow(thresh,cmap**=**'gray')

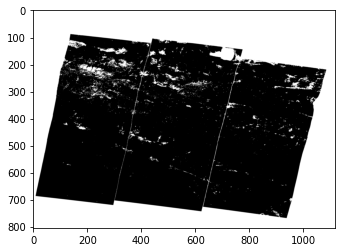
plt**.**figure()

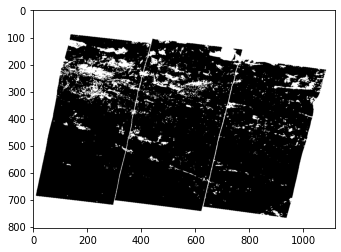
plt**.**imshow(edges,cmap**=**'gray')

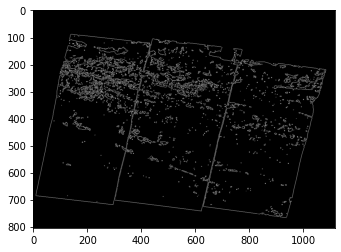
plt**.**figure()

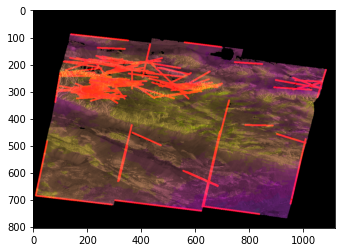
plt**.**imshow(lines\_edges,cmap**=**'gray')

Out[ ]:









In [ ]:

*# filling holes*

thresh **=** cv2**.**threshold(a, 10, 255, cv2**.**THRESH\_BINARY\_INV)[1]

im\_floodfill **=** thresh**.**copy()

h, w **=** thresh**.**shape[:2]

mask **=** np**.**zeros((h**+**2, w**+**2), np**.**uint8)

cv2**.**floodFill(im\_floodfill, mask, (0,0), 255);

final **=** np**.**multiply(255**-**thresh, im\_floodfill)

line\_image **=** np**.**copy(image) **\*** 0

mask **=** 1 **-** final

kernel**=**cv2**.**getStructuringElement(cv2**.**MORPH\_RECT,(5,5))

mask **=** cv2**.**morphologyEx(mask, cv2**.**MORPH\_DILATE, kernel)

final **=** cv2**.**morphologyEx(final, cv2**.**MORPH\_DILATE, kernel)

final **=** np**.**multiply(mask, final)

mask\_dil **=** cv2**.**morphologyEx(mask, cv2**.**MORPH\_DILATE, kernel)

mask\_erd **=** cv2**.**morphologyEx(mask, cv2**.**MORPH\_ERODE, kernel)

bound **=** mask\_dil **-** mask\_erd

bound **=** cv2**.**morphologyEx(bound, cv2**.**MORPH\_DILATE, kernel)

bound **=** 1 **-** bound

final **=** np**.**multiply(bound, final)

*# Run Hough on edge detected image*

*# Output "lines" is an array containing endpoints of detected line segments*

lines **=** cv2**.**HoughLinesP(final, rho, theta, threshold, np**.**array([]),

min\_line\_length, max\_line\_gap)

**for** line **in** lines:

**for** x1,y1,x2,y2 **in** line:

cv2**.**line(line\_image,(x1,y1),(x2,y2),(255,0,0),5)

lines\_edges **=** cv2**.**addWeighted(image, 0.8, line\_image, 1, 0)

plt**.**figure(figsize**=**(15, 15))

plt**.**subplot(321), plt**.**imshow(thresh, cmap**=**'gray')

plt**.**subplot(322), plt**.**imshow(im\_floodfill, cmap**=**'gray')

plt**.**subplot(323), plt**.**imshow(mask, cmap**=**'gray')

plt**.**subplot(324), plt**.**imshow(bound, cmap**=**'gray')

plt**.**subplot(325), plt**.**imshow(lines\_edges, cmap**=**'gray')

plt**.**subplot(326), plt**.**imshow(final, cmap**=**'gray')

Out[ ]:

(,

)

