3. PROPOSED WORK / METHODOLOGY

1. Image acquisition/capture:

First collect CT scan images of lung cancer which are stored in matlab .CT scan images has low noise so we select them. Computed Tomography having better clarity, low distortion and noise. CT scan images stored in database in JPEG/PNG format.

2. Image Enhancement:

Image enhancement is basically used to improve the quality of the image. This technique can be performed in both domains, Spatial domain as well as in Frequency domain. The objective of the Image Enhancement is to make Image better, improving the quality of the Image. An Enhancement algorithm is used for the purpose of some particular application, which can be done either increasing or the suppressing the noise or contrast of the image. In Image enhancement we can increase or decrease the brightness of the image as per our application. This technique improves the visual perception of the image. Basically, this technique can be classified into two categories. 1. Spatial domain method. 2. Transform domain method Spatial domain method are operates directly on the image, while Transform domain method operates on Fourier transform of the image and then again it back into the spatial domain. Enhancement techniques are basically, based on the histograms, because it is very simple and fast.

Image Enhancement is used to make the image sharp. It accentuates and sharp the image features such as edges, boundaries and contrast, hence we can easily visualize the graphical data inside of the image. For Enhancement we used Median filter.

3. Image Segmentation:

The main objective of the Image Segmentation is to extract various part of the image. It is a process of partitioning the image into pixel. Segmentation algorithms are based on area oriented, thus segmentation is used to divide the required region from the image. The main goal of Image Segmentation is to divide the image and change the style to represent it. Image Segmentation is a process of partitioning the image into multiple segments.

Segmentation is basically used to locate boundaries, curves, lines around the object. To make contours' this method is used, hence we can easily detect our object. For Segmentation we used the ROI algorithm.

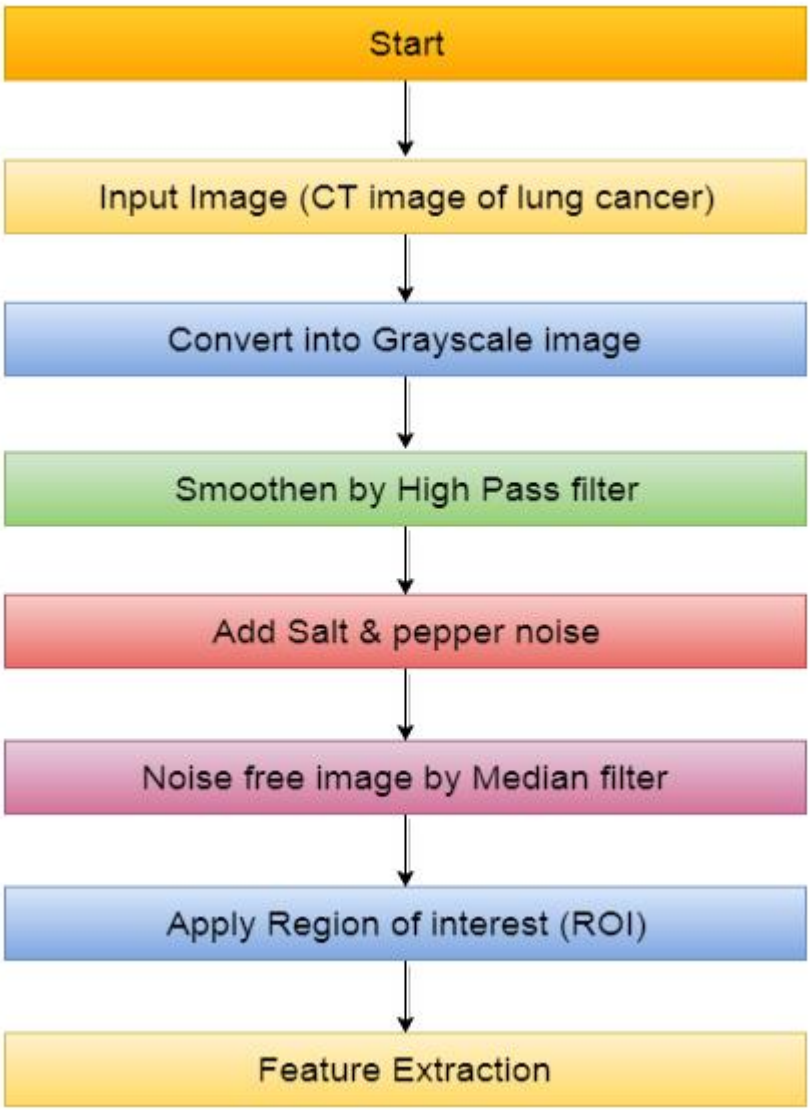
3.1 Thresholding Approach

Thresholding is one of the most powerful tools for image segmentation. The segmented image obtained from thresholding has the advantages of smaller storage space, fast processing speed and ease in manipulation, compared with gray level image which usually contains 256 levels. In this we have a gray scale image given for a thresholding procedure it converts the rgb image into a binary image i.e black and white image which has only two shades i.e black and white which represent the level 0 and 1 only.the threshold value for this will be lies between 0 and 1 because it has only two levels., after achieving the threshold value; image will be segmented based on it.

4. Feature Extraction:

It represents the interested part of the image. It describes the information in the extracted region. Features of the image can be extracted by it's content like colors, textures, shape, position, edges and the regions etc... The selected features are the expected information data from the Input data , hence we can perform the desired task

The algorithm of lung cancer detection is:



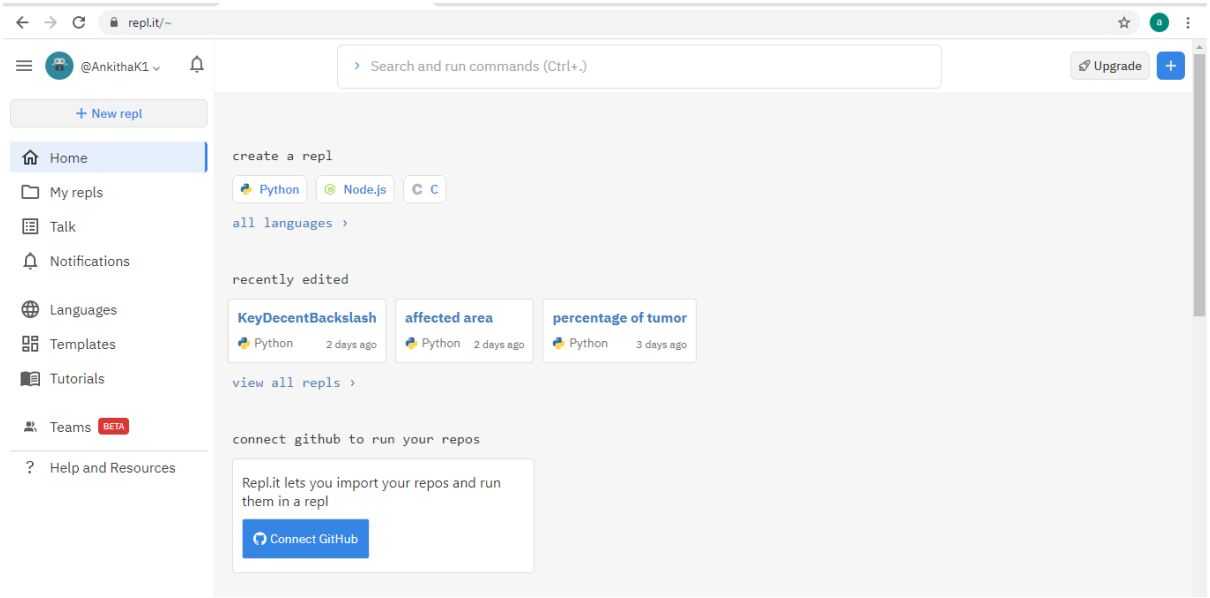
5. DESGIN AND IMPLEMENTATION

5.1 Environmental setup

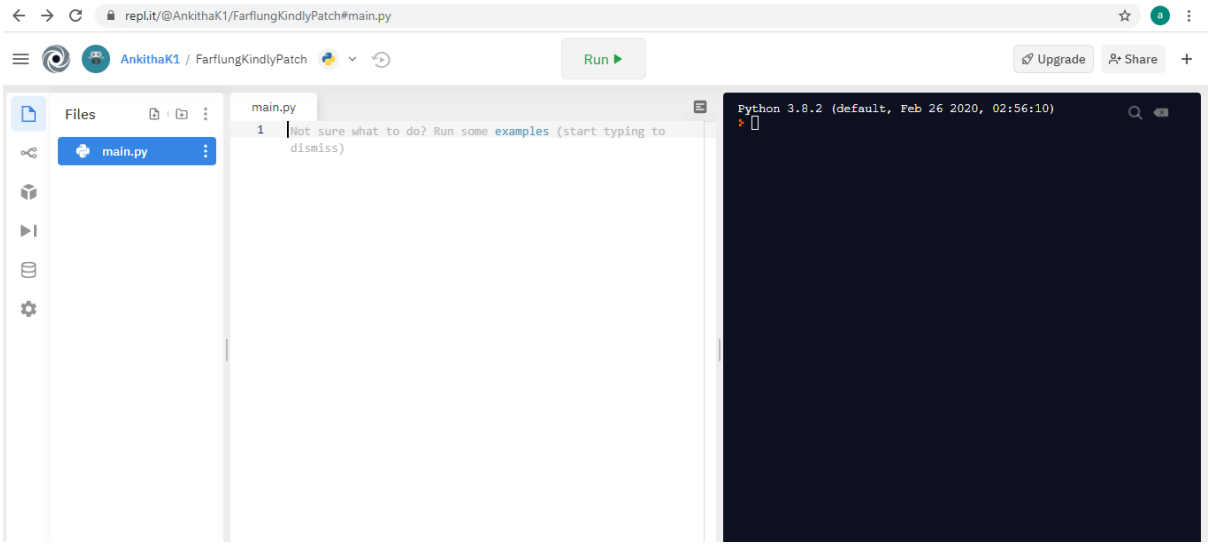
5.2 Implementation Steps

- **Use online tool Repl.it to run and implement the project**

  Use Python package and install other necessary libraries.



Python editor

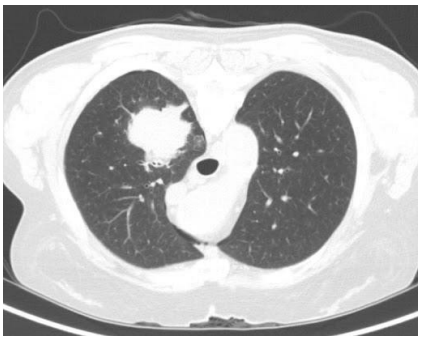


❖ Download the Lung images

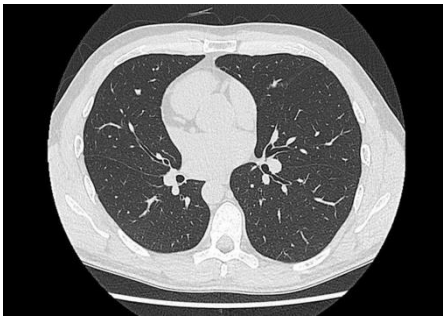
Download both normal and diseased Lung image.

CT Scan images: In JPEG or PNG format.

❖ Input CT images



Diseased lung (Tumor)



Normal lung

5.3 Work-flow / Implementation Steps

5.3.1 Input image

The taken input image is the computed tomography (CT) of the lung tumor

5.3.2 Gray scale Image:

The value of each pixel is single sample, which carries only intensity information of the pixel. Gray scale image is also known as black and white image, where it varying from black (weakest) intensity to white (hardest) intensity. Grayscale image consist exclusive shades of gray. This type of Images are the result of measuring the intensity of light at each pixel in a single band of the light spectrum such as infrared, ultraviolet, visible light etc… We can also covert the RGB and YGB (Colored) image into Grayscale image.

5.3.3 High Pass Filter:

To make image sharp and for smoothing purpose, we used High pass filter. High pass filters are passes the high frequency but it attenuates the frequency lower than the cut-off frequency. High pass filter sets the high threshold cut-off, hence to detect the information contain in the image while cutting low frequency.

5.3.4 Median Filter:

Median filter is a very useful and effective technique to reduce the noise in the image. This detects the edges, first noise should be removed in image and then, edge removal is performed. The main feature of this filter is to remove the noise without removing edges.

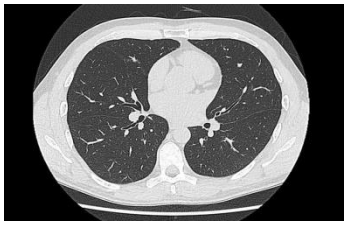
5.3.5 Region Of Interest (ROI):

Basically it is used to detect the area. Using ROI we defined the boundaries of the object and we can know the area of the object. REGION OF INTEREST (ROI) It is the samples within a Image. The CT-scanned image which consist cancer tumor in the lungs node.

6. RESULTS AND DISCUSSION

Result:

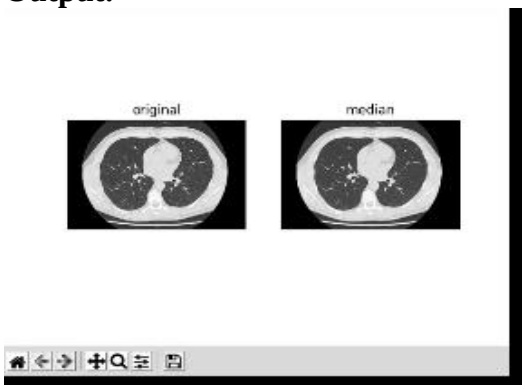
6.1 Input image :



6.2 Image Enhancement:(Median filter)

```
import cv2
import numpy as np
from matplotlib import pyplot as plt

img = cv2.imread('CT scan normal.jpg')

median = cv2.medianBlur(img,5)

plt.subplot(121),plt.imshow(img),plt.title('original')
plt.xticks([]), plt.yticks([])
plt.subplot(122),plt.imshow(median),plt.title('median')
plt.xticks([]), plt.yticks([])
plt.show()
```

Output:

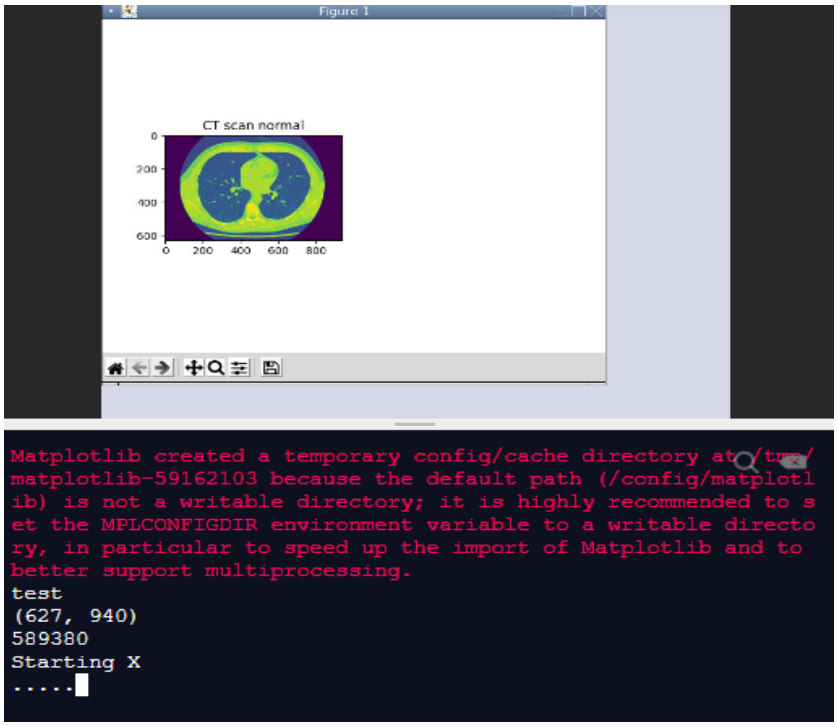


Size and shape of the image:

main.py

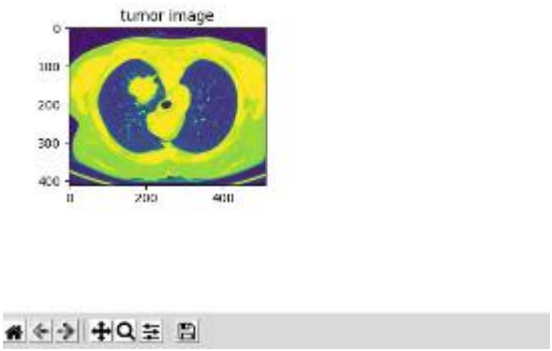
```
1   import cv2
2   from matplotlib import pyplot as plt
3   print('test')
4   ip = cv2.imread('CT scan normal.jpg',0)
5   print(ip.shape)
6   print(ip.size)
7   plt.subplot(121),plt.imshow(ip),plt.title('CT scan normal')
8   plt.show()
```

Output:



Normal lung

Diseased image:




```
test
(410, 512)
209920
```

6.3 Apply Grayscale image:

main.py

```python
import cv2

image = cv2.imread('tumor.jpg')
gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

cv2.imshow('Original image',image)
cv2.imshow('Gray image', gray)

cv2.waitKey(0)
cv2.destroyAllWindows()
```

Output:

6.4 high pass filter:

main.py

```python
import cv2
import numpy as np
from scipy import ndimage

kernel_5x5 = np.array([[-1,-1,-1,-1,-1],
                       [-1, 1, 2, 1,-1],
                       [-1, 2, 4, 2,-1],
                       [-1, 1, 2, 1,-1],
                       [-1,-1,-1,-1,-1]])


img = cv2.imread('gray scale img.png', 0)

k5 = ndimage.convolve(img, kernel_5x5)


Blurred = cv2.GaussianBlur(img, (11,11), 0)
g_hpf = img - Blurred


cv2.imshow("5x5", k5)

cv2.imshow("g_hpf", g_hpf)
cv2.waitKey()
cv2.destroyAllWindows()
```

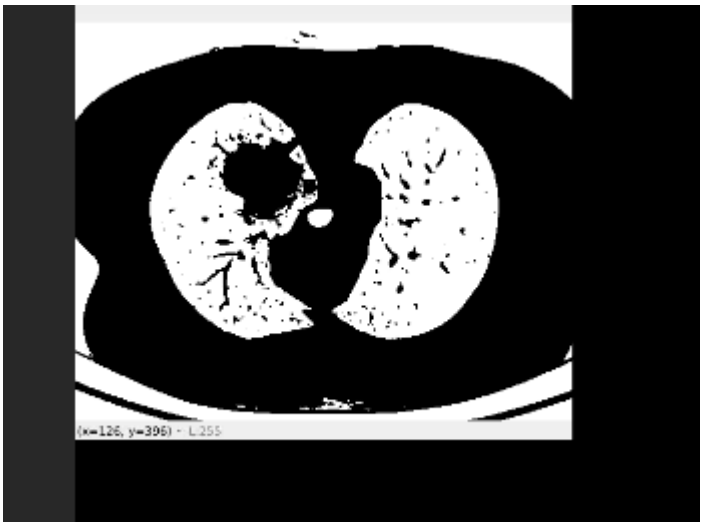
Output:

6.5 Segmentation using Thresholding:

main.py

```
import cv2
import numpy as np

image1 = cv2.imread('tumor.jpg')


img = cv2.cvtColor(image1, cv2.COLOR_BGR2GRAY)

ret, thresh1 = cv2.threshold(img, 120, 255, cv2.THRESH_BINARY)
ret, thresh2 = cv2.threshold(img, 120, 255, cv2.THRESH_BINARY_INV)

cv2.imshow('Binary Threshold', thresh1)
cv2.imshow('Binary Threshold Inverted', thresh2)

if cv2.waitKey(0) & 0xff == 27:
    cv2.destroyAllWindows()
```

Output:



Affected area

Code:

```
import cv2
import numpy as np
import argparse

img1 = cv2.imread('anki.jpg')
img = cv2.resize(img1, (0,0), fx=0.5, fy=0.5)
original = img.copy()
neworiginal = img.copy()
roi = img.copy()
originalroi = img.copy()

blur1 = cv2.GaussianBlur(img,(3,3),1)

newimg = np.zeros((img.shape[0], img.shape[1],3),np.uint8)
criteria = (cv2.TERM_CRITERIA_EPS + cv2.TERM_CRITERIA_MAX_ITER , 10 ,1.0)
img = cv2.pyrMeanShiftFiltering(blur1, 20, 30, newimg, 0, criteria)

blur = cv2.GaussianBlur(img,(11,11),1)

kernel = np.ones((5,5),np.uint8)
canny = cv2.Canny(blur, 200, 290)
res = cv2.morphologyEx(canny,cv2.MORPH_CLOSE, kernel)
canny = cv2.cvtColor(canny,cv2.COLOR_GRAY2BGR)
cv2.imshow('Canny',res)

hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)
lower = np.array([5,25,25])
```

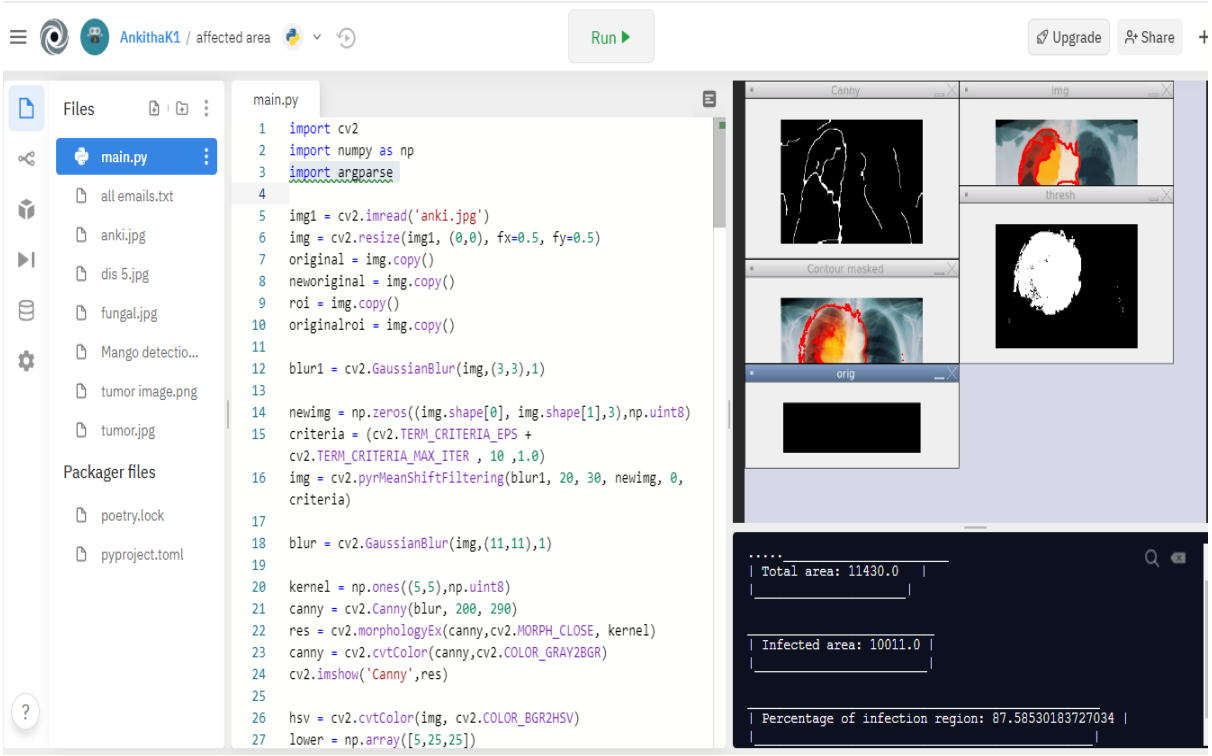
```
upper = np.array([70,255,255])

mask = cv2.inRange(hsv, lower, upper)

res = cv2.bitwise_and(hsv,hsv, mask= mask)
gray = cv2.cvtColor(res,cv2.COLOR_BGR2GRAY)
ret, thresh = cv2.threshold(gray,0,255,cv2.THRESH_BINARY+cv2.THRESH_OTSU)
contours, hierarchy = cv2.findContours(thresh,cv2.RETR_TREE,cv2.CHAIN_APPROX_NONE)


for i in contours:
    cnt = cv2.contourArea(i)
    #M = cv2.momens(i)
    #cx = int(M['m10']/M['m00'])
    if cnt > 1000:
        cv2.drawContours(img, [i], 0, (0,0,255), 2)

gray = cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
ret, thresh = cv2.threshold(gray,0,255,cv2.THRESH_BINARY+cv2.THRESH_OTSU)
contours, hierarchy = cv2.findContours(thresh,cv2.RETR_TREE,cv2.CHAIN_APPROX_NONE)
cnt = max(contours, key=cv2.contourArea)
Tarea = cv2.contourArea(cnt)
cv2.imshow('img', img)

height, width, _ = canny.shape
min_x, min_y = width, height
max_x = max_y = 0
frame = canny.copy()

for contour, hier in zip(contours, hierarchy):

 (x,y,w,h) = cv2.boundingRect(contour)
min_x, max_x = min(x, min_x), max(x+w, max_x)
min_y, max_y = min(y, min_y), max(y+h, max_y)
if w > 80 and h > 80:
            cv2.rectangle(frame, (x,y), (x+w,y+h), (255, 0, 0), 2)
            roi = img[y:y+h , x:x+w]
            originalroi = original[y:y+h , x:x+w]
if max_x - min_x > 0 and max_y - min_y > 0:
    cv2.rectangle(frame, (min_x, min_y), (max_x, max_y), (255, 0, 0), 2)

img = roi

imghls = cv2.cvtColor(roi, cv2.COLOR_BGR2HSV)
```

```
imghls[np.where((imghls==[30,200,2]).all(axis=2))] = [0,200,0]

huehls = imghls[:,:,0]

huehls[np.where(huehls==[0])] = [35]

#Thresholding on hue image
ret, thresh = cv2.threshold(huehls,28,255,cv2.THRESH_BINARY_INV)
cv2.imshow('thresh', thresh)


mask = cv2.bitwise_and(originalroi,originalroi,mask = thresh)

contours,heirarchy = cv2.findContours(thresh, cv2.RETR_TREE, cv2.CHAIN_APPROX_NONE)

Infarea = 0

for x in range(len(contours)):

    cv2.drawContours(originalroi,contours[x],-1,(0,0,255),2)

    cv2.imshow('Contour masked',originalroi)

    #Calculating area of infected region
    Infarea += cv2.contourArea(contours[x])



#if Infarea > Tarea:
    #Tarea = img.shape[0]*img.shape[1]

print ('_____\n| Total area: ' + str(Tarea) + '   |\n|_____|')

#Finding the percentage of infection in the banana
print ('\n_____\n| Infected area: ' + str(Infarea) + '
|\n|_____|')

try:
    per = 100 * Infarea/Tarea

except ZeroDivisionError:
    per = 0
```

print('\n_____\n| Percentage of infection region: ' + str(per) + ' |\n|_____|')

cv2.imshow('orig',original)

Output:



7. CONCLUSION

Lung cancer is a very dangerous disease in the world. The expert physicians diagnose the disease by experience. The treatment includes surgery, chemotherapy, radiation, and targeted therapy. These treatments are lengthy, costly, and painful. Hence an attempt is made to atomize this procedure to detect the lung cancer using image processing techniques. High pass filter and Median filter gives the best result for Image enhancement to make image sharp and noise free. The Region of Interest is used for Image Segmentation. For future work, we can also know the size of the tumor in the lungs such as we can find the staging of the cancer and also find the eccentricity, intensity, Average intensity, Perimeter of the tumor.

An image improvement technique is developing for earlier disease detection and treatment stages; the time factor was taken in account to discover the abnormality issues in target images. Image quality and accuracy is the core factors of this research, image quality assessment as well as enhancement stage where were adopted on low pre-processing techniques.