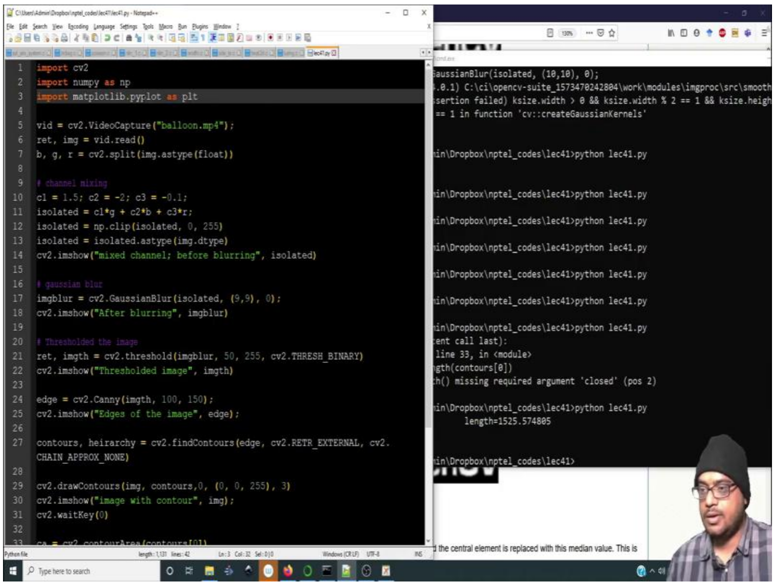
Tools in Scientific Computing
Prof. Aditya Bandopadhyay
Department of Mechanical Engineering
Indian Institute of Technology, Kharagpur

Lecture - 42
Balloon problem and viscous fingers

(Refer Slide Time: 00:29)



```python
import cv2
import numpy as np
import matplotlib.pyplot as plt

vid = cv2.VideoCapture("balloon.mp4");
ret, img = vid.read()
b, g, r = cv2.split(img.astype(float))

# channel mixing
c1 = 1.5; c2 = -2; c3 = -0.1;
isolated = c1*g + c2*b + c3*r;
isolated = np.clip(isolated, 0, 255)
isolated = isolated.astype(img.dtype)
cv2.imshow("mixed channel; before blurring", isolated)

# gaussian blur
imgblur = cv2.GaussianBlur(isolated, (9,9), 0);
cv2.imshow("After blurring", imgblur)

# Thresholded the image
ret, imgth = cv2.threshold(imgblur, 50, 255, cv2.THRESH_BINARY)
cv2.imshow("Thresholded image", imgth)

edge = cv2.Canny(imgth, 100, 150);
cv2.imshow("Edges of the image", edge);

contours, heirarchy = cv2.findContours(edge, cv2.RETR_EXTERNAL, cv2.
CHAIN_APPROX_NONE)

cv2.drawContours(img, contours,0, (0, 0, 255), 3)
cv2.imshow("image with contour", img);
cv2.waitKey(0)

ca = cv2.contourArea(contours[0])
```

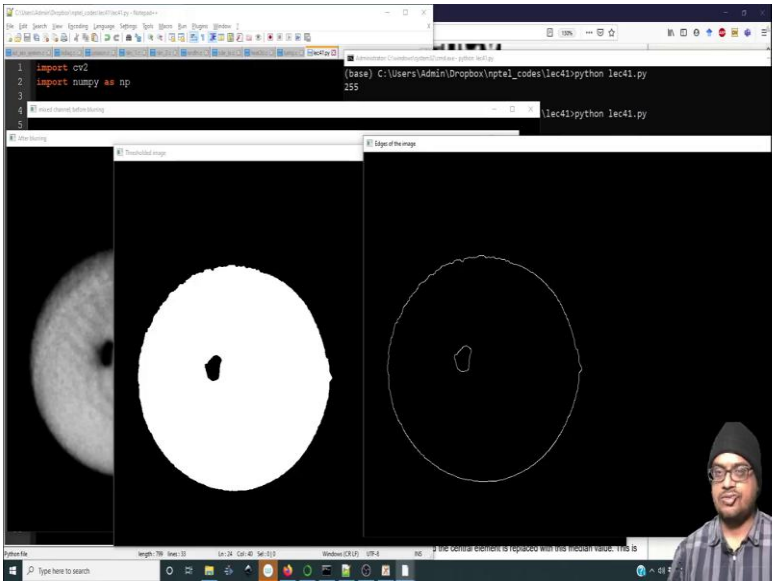
```
GaussianBlur(isolated, (10,10), 0);
.0.1) C:\ci\opencv-suite_1573470242804\work\modules\imgproc\src\smooth
ssertion failed) ksize.width > 0 && ksize.width % 2 == 1 && ksize.heigh
 == 1 in function 'cv::createGaussianKernels'


in\Dropbox\nptel_codes\lec41>python lec41.py


in\Dropbox\nptel_codes\lec41>python lec41.py

in\Dropbox\nptel_codes\lec41>python lec41.py

in\Dropbox\nptel_codes\lec41>python lec41.py

in\Dropbox\nptel_codes\lec41>python lec41.py

in\Dropbox\nptel_codes\lec41>python lec41.py

in\Dropbox\nptel_codes\lec41>python lec41.py
ent call last):
 line 33, in <module>
gth(contours[0])
h() missing required argument 'closed' (pos 2)

in\Dropbox\nptel_codes\lec41>python lec41.py
      length=1525.574805


in\Dropbox\nptel_codes\lec41>
```

d the central element is replaced with this median value. This is

Hello again, welcome to this next lecture where we going to proceed with the image processing part. So, so far we have figured out what the thresholded image looks like. Now, what we are going to do is actually find out the contour on that image; the contour that before, but before that we have to first detect the edge. So, let us do that. So, we are going to say edge = cv2.Canny image threshold.

Whatever we thresholded Canny lower limit Canny upper limit and yeah that is it. Then we will do cv2.imshow("Edges of the image", edge) we will say edges of the image we are going to plot edge. Let us see what we get.

(Refer Slide Time: 01:26)



```
import cv2
import numpy as np
```

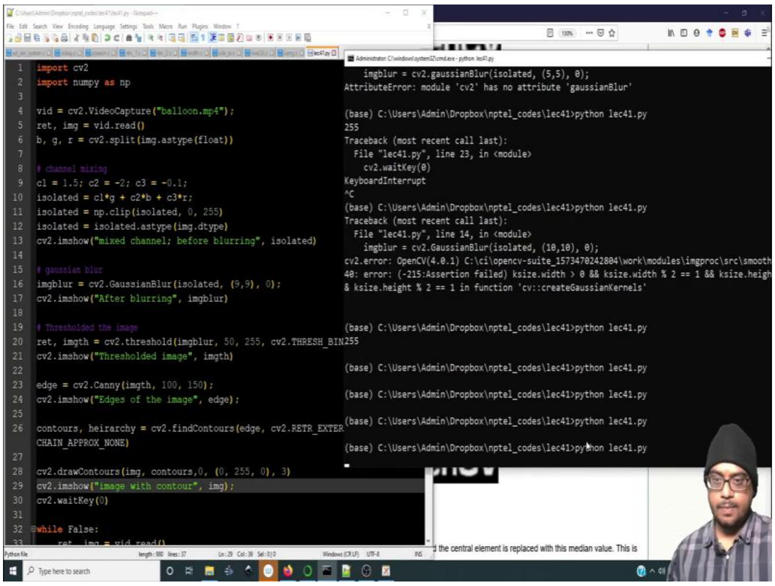
```
(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
255

\lec41>python lec41.py
```

So, this is what we obtain as the edge of the image it is quite nice to be honest. And so what did we start with? We started with an image which had or a frame which had my hand bunch of other things and thanks to the very first step in which we mixed the modes or mix the signals or mix the channels rather.

We were able to get rid of all those things we smoothened the image actually even if we do not smoothen the image nothing much changes, but it is always a good thing to smooth it ok. For finding out contours and all it is always a good idea to have less information. Then we have 2 contours, one is the two edges one is the outside thing another one is the inside thing.

So, the inside thing it comes up because of that reflection of the light nothing else, but we will see once we detect the contours that it will it will not cause a lot of image a lot of issues. So, now that we found out the edges of the image we are going to pass the edges to find out the contours. Well you could have done it directly with the thresholded image as well, but regardless.

(Refer Slide Time: 02:43)



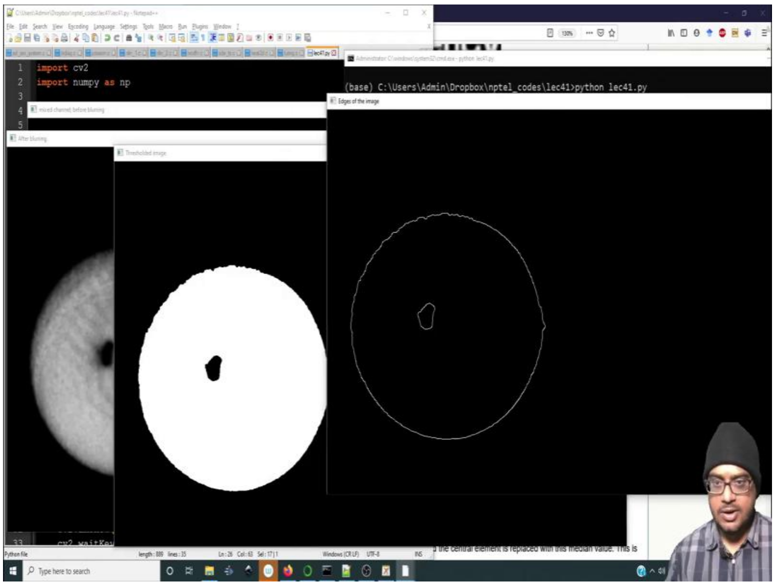
```
import cv2
import numpy as np

vid = cv2.VideoCapture("balloon.mp4");
ret, img = vid.read()
b, g, r = cv2.split(img.astype(float))

# channel mixing
c1 = 1.5; c2 = -2; c3 = -0.1;
isolated = c1*g + c2*b + c3*r;
isolated = np.clip(isolated, 0, 255)
isolated = isolated.astype(img.dtype)
cv2.imshow("mixed channel; before blurring", isolated)

# gaussian blur
imgblur = cv2.GaussianBlur(isolated, (9,9), 0);
cv2.imshow("After blurring", imgblur)

# Thresholded the image
ret, imgth = cv2.threshold(imgblur, 50, 255, cv2.THRESH_BIN
cv2.imshow("Thresholded image", imgth)

edge = cv2.Canny(imgth, 100, 150);
cv2.imshow("Edges of the image", edge);

contours, heirarchy = cv2.findContours(edge, cv2.RETR_EXTER
CHAIN_APPROX_NONE)

cv2.drawContours(img, contours,0, (0, 255, 0), 3)
cv2.imshow("image with contour", img);
cv2.waitKey(0)

while False:
    ret, img = vid.read()
```

```
    imgblur = cv2.gaussianBlur(isolated, (5,5), 0);
AttributeError: module 'cv2' has no attribute 'gaussianBlur'

(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
255
Traceback (most recent call last):
  File "lec41.py", line 23, in <module>
    cv2.waitKey(0)
KeyboardInterrupt
^C
(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
Traceback (most recent call last):
  File "lec41.py", line 14, in <module>
    imgblur = cv2.GaussianBlur(isolated, (10,10), 0);
cv2.error: OpenCV(4.0.1) C:\ci\opencv-suite_1573470242804\work\modules\imgproc\src\smooth
40: error: (-215:Assertion failed) ksize.width > 0 && ksize.width % 2 == 1 && ksize.heigh
& ksize.height % 2 == 1 in function 'cv::createGaussianKernels'


(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
255

(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py

(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py

(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py

(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
```

Administrator: C:\windows\system32\cmd.exe - python lec41.py
the central element is replaced with this median value. This is
Type here to search

So, we are going to do contours hierarchy well, you can call them anything you want is equal to cv2.findContours you are going to pass the image from which you want to find then you going to say RETURN EXTERNAL and you are going to CHAIN APPROX NONE. So, the findContours function it requires the image then it requires information about how you want to return the contour. So, RETR EXTERNAL means in that image.

(Refer Slide Time: 03:28)



```
import cv2
import numpy as np
```

```
(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
```

mixed channel; before blurring
After blurring
Thresholded image
Edges of the image
the central element is replaced with this median value. This is
Type here to search

So, let me run this. So, when it is going to detect these two contours it is going to give you only the parent contour because inside that parent contour there is a child contour as well. So, you are going to remove that contour from the list alright.

So, this is a very important thing you only want the biggest contour or the enclosing contour if you do not do this it is you are going to get this contour as well or you do not want that then you do cv2.CHAIN_APPROX_NONE so.

Is the purpose of CHAIN APPROX NONE, if you have for example, a straight line and you do not want to save all the points on the line we can simply save the two points. So, if you do CHAIN APPROX simple you are going to save the end points you are going to interpolate once you start doing something with it, but in in this case we have a complicated surface. So, we do CHAIN APPROX NONE right.

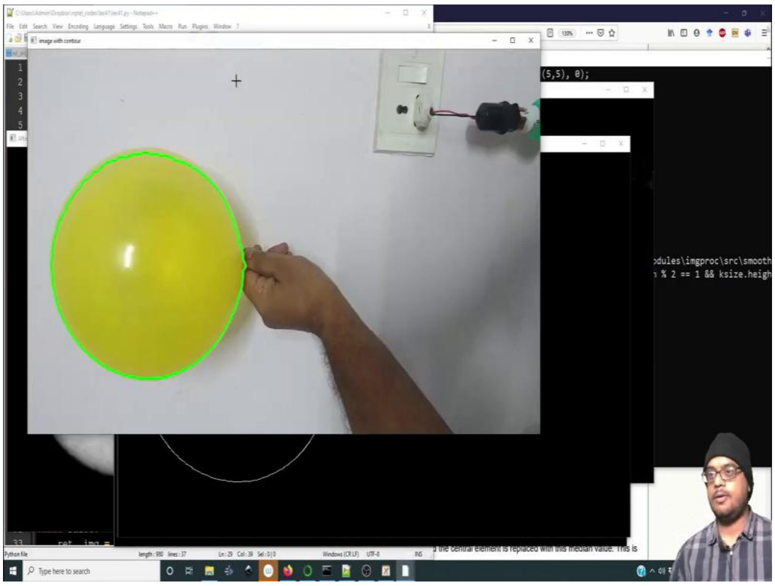
This helps you obtain the various contours not the various contour, but the largest contour. So, now, once we know what the contours are we will do cv2.DRAW on oops cv2.drawContours then we are going to plot the contour on the original image.

So, the original image was this alright. So, drawContours on image we are going to draw the contours we are going because there is only one contour. So, 0 means, so if you do minus 1 you going to plot all the contours 0 is the id of the contour.

In this case you have only one contour. So, contour 0 is the equivalent of this, then you need to specify the color of the contour. So, color will be specified as the tuple like this and we have to give the line thickness of the contour. So, it is going to overlay the contour on the image and show it alright.

So, let me save this and let me see what happens where is the original image? So, we need to show the image again cv2.imshow("image with contour", img) we have to give the name image with contour image alright.

(Refer Slide Time: 06:17)

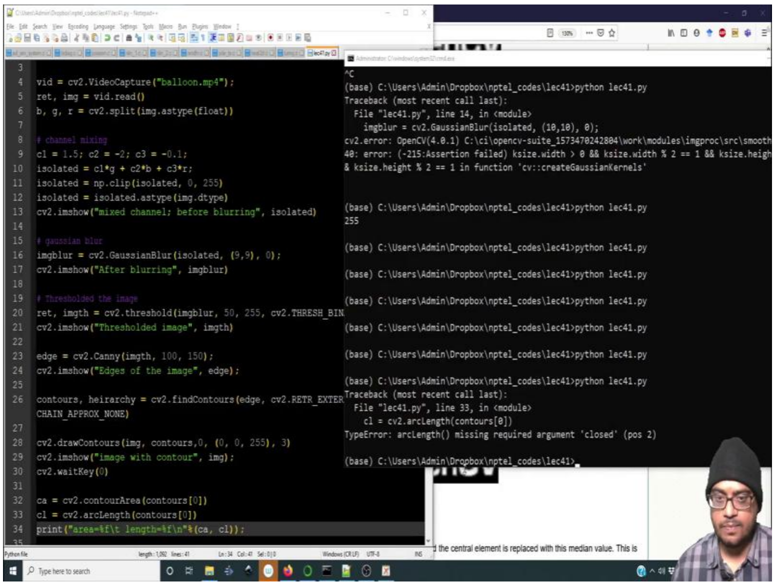


```
(5,5), 0);
```

```
dules\imgproc\src\smooth
 % 2 == 1 && ksize.heigh
```

the central element is replaced with this median value. This is

(Refer Slide Time: 06:38)



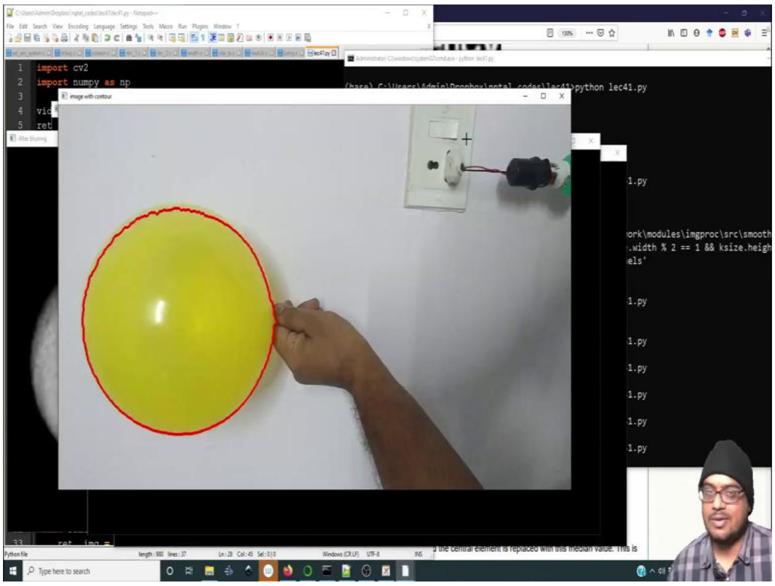
```

vid = cv2.VideoCapture("balloon.mp4");
ret, img = vid.read()
b, g, r = cv2.split(img.astype(float))

# channel mixing
c1 = 1.5; c2 = -2; c3 = -0.1;
isolated = c1*g + c2*b + c3*r;
isolated = np.clip(isolated, 0, 255)
isolated = isolated.astype(img.dtype)
cv2.imshow("mixed channel; before blurring", isolated)

# gaussian blur
imgblur = cv2.GaussianBlur(isolated, (9,9), 0);
cv2.imshow("After blurring", imgblur)

# Thresholded the image
ret, imgth = cv2.threshold(imgblur, 50, 255, cv2.THRESH_BIN
cv2.imshow("Thresholded image", imgth)

edge = cv2.Canny(imgth, 100, 150);
cv2.imshow("Edges of the image", edge);

contours, heirarchy = cv2.findContours(edge, cv2.RETR_EXTER
CHAIN_APPROX_NONE)

cv2.drawContours(img, contours,0, (0, 0, 255), 3)
cv2.imshow("image with contour", img);
cv2.waitKey(0)

ca = cv2.contourArea(contours[0])
cl = cv2.arcLength(contours[0])
print("area=%f\t length=%f\n"%(ca, cl));
```

```
^C
(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
Traceback (most recent call last):
  File "lec41.py", line 14, in <module>
    imgblur = cv2.GaussianBlur(isolated, (10,10), 0);
cv2.error: OpenCV(4.0.1) C:\ci\opencv-suite_1573470242804\work\modules\imgproc\src\smooth
40: error: (-215:Assertion failed) ksize.width > 0 && ksize.width % 2 == 1 && ksize.heigh
& ksize.height % 2 == 1 in function 'cv::createGaussianKernels'


(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
255

(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py

(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py

(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py

(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py

(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py

(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
Traceback (most recent call last):
  File "lec41.py", line 33, in <module>
    cl = cv2.arcLength(contours[0])
TypeError: arcLength() missing required argument 'closed' (pos 2)

(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>
```

the central element is replaced with this median value. This is

This is great we have detected the contour of the balloon. So, to 0, 255, 0 means b g r. So, g, so then the color of the contour becomes green. In case you want a red contour then, we instead of g channel to 255 we are going to set it to this to 255. So, this is going to make the contour to be red ok.

(Refer Slide Time: 06:46)



```
import cv2
import numpy as np

vid
ret
```

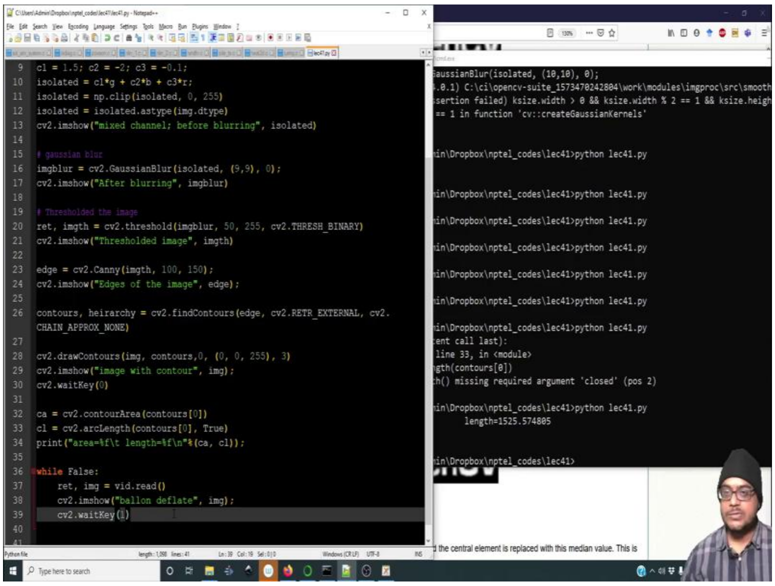
```
(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
1.py
ork\modules\imgproc\src\smooth
.width % 2 == 1 && ksize.heigh
els'
1.py
1.py
1.py
1.py
1.py
1.py
```

It is as simple as that. So, so far what we have done is great we have identified the contour we have plotted the contour, now we have also want the area of the contour right. So, what is the area of the contour? So, thankfully we have functions which help us in finding out the area the perimeter of the contour.

So, its cv2.contourArea(contours[0]). So, this is the main contour in case you have other contours you have to give different indices like this, but because we have only one contour it is going to be contour[0].

We also have cv2.arcLength(contours[0], True) and we are going to give contours[0] right. So, these two should give us the area equals %f\t and length equal to %f\n. % we are going to call this ca and we are going to call this cl. So, then (ca, cl). So, let us see what happens when we run this let me close all the images. So, the argument is required ok.

(Refer Slide Time: 08:32)



```
cl = 1.5; c2 = -2; c3 = -0.1;
isolated = cl*g + c2*b + c3*r;
isolated = np.clip(isolated, 0, 255)
isolated = isolated.astype(img.dtype)
cv2.imshow("mixed channel; before blurring", isolated)

# gaussian blur
imgblur = cv2.GaussianBlur(isolated, (9,9), 0);
cv2.imshow("After blurring", imgblur)

# Thresholded the image
ret, imgth = cv2.threshold(imgblur, 50, 255, cv2.THRESH_BINARY)
cv2.imshow("Thresholded image", imgth)

edge = cv2.Canny(imgth, 100, 150);
cv2.imshow("Edges of the image", edge);

contours, heirarchy = cv2.findContours(edge, cv2.RETR_EXTERNAL, cv2.
CHAIN_APPROX_NONE)

cv2.drawContours(img, contours,0, (0, 0, 255), 3)
cv2.imshow("image with contour", img);
cv2.waitKey(0)

ca = cv2.contourArea(contours[0])
cl = cv2.arcLength(contours[0], True)
print("area=%f\t length=%f\n"%(ca, cl));

while False:
    ret, img = vid.read()
    cv2.imshow("ballon deflate", img);
    cv2.waitKey(1)
```

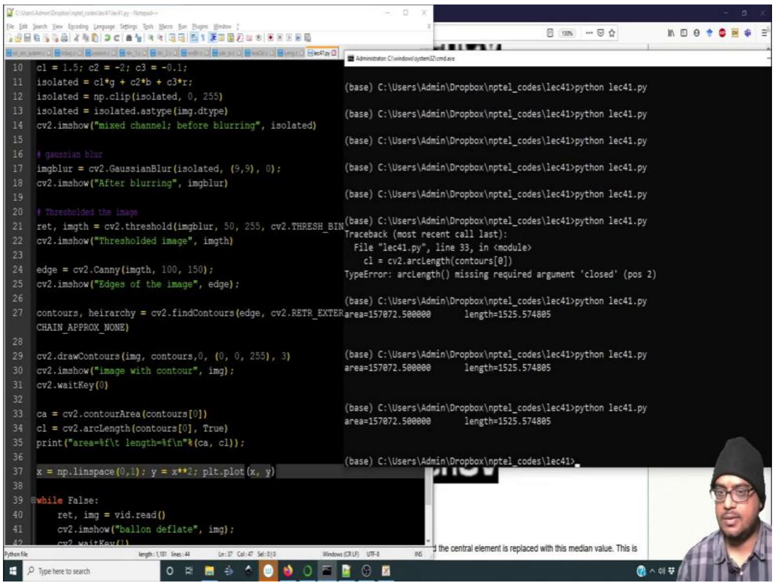
```
GaussianBlur(isolated, (10,10), 0);
.0.1) C:\ci\opencv-suite_1573470242804\work\modules\imgproc\src\smooth
sertion failed) ksize.width > 0 && ksize.width % 2 == 1 && ksize.heigh
== 1 in function 'cv::createGaussianKernels'


in\Dropbox\nptel_codes\lec41>python lec41.py


in\Dropbox\nptel_codes\lec41>python lec41.py

in\Dropbox\nptel_codes\lec41>python lec41.py

in\Dropbox\nptel_codes\lec41>python lec41.py

in\Dropbox\nptel_codes\lec41>python lec41.py

in\Dropbox\nptel_codes\lec41>python lec41.py

in\Dropbox\nptel_codes\lec41>python lec41.py
ent call last):
line 33, in <module>
gth(contours[0])
h() missing required argument 'closed' (pos 2)

in\Dropbox\nptel_codes\lec41>python lec41.py
      length=1525.574805


in\Dropbox\nptel_codes\lec41>
```

the central element is replaced with this median value. This is

So, the arcLength also requires another argument which helps in telling whether it is a closed contour or an open contour. So, if it is a closed contour we have to tell True if it is an open contour we have to say false. So, the area is this and the arc length is this. Well great if you divided by 2π you get the projected radius that is fine I am not going to do that because it is quite trivial to do what we have to do is to wrap all of this inside a for loop that is it ok.

We have to now wrap all of this inside the for loop and then, we are done sort of ok. Let me see whether I can invoke my plot lib.

(Refer Slide Time: 09:30)



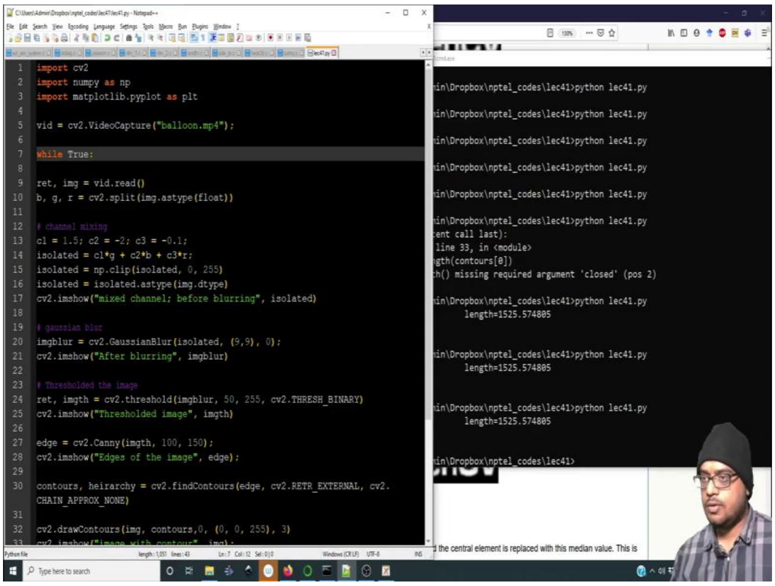
```
cl = 1.5; c2 = -2; c3 = -0.1;
isolated = cl*g + c2*b + c3*r;
isolated = np.clip(isolated, 0, 255)
isolated = isolated.astype(img.dtype)
cv2.imshow("mixed channel; before blurring", isolated)

# gaussian blur
imgblur = cv2.GaussianBlur(isolated, (9,9), 0);
cv2.imshow("After blurring", imgblur)

# Thresholded the image
ret, imgth = cv2.threshold(imgblur, 50, 255, cv2.THRESH_BIN
cv2.imshow("Thresholded image", imgth)

edge = cv2.Canny(imgth, 100, 150);
cv2.imshow("Edges of the image", edge);

contours, heirarchy = cv2.findContours(edge, cv2.RETR_EXTER
CHAIN_APPROX_NONE)

cv2.drawContours(img, contours,0, (0, 0, 255), 3)
cv2.imshow("image with contour", img);
cv2.waitKey(0)

ca = cv2.contourArea(contours[0])
cl = cv2.arcLength(contours[0], True)
print("area=%f\t length=%f\n"%(ca, cl));

x = np.linspace(0,1); y = x**2; plt.plot(x, y)

while False:
    ret, img = vid.read()
    cv2.imshow("ballon deflate", img);
```

```
(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py

(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py

(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py

(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py

(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py

(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
Traceback (most recent call last):
  File "lec41.py", line 33, in <module>
    cl = cv2.arcLength(contours[0])
TypeError: arcLength() missing required argument 'closed' (pos 2)

(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
area=157072.500000       length=1525.574805


(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
area=157072.500000       length=1525.574805


(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
area=157072.500000       length=1525.574805


(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>
```

So, import matplotlib.pyplot as plt and x = np.linspace (0, 1) , $y = x^2$ here cannot plot x, y. So, let me see if it executes this in this because in, but it does not show the plot and that is ok. We can dump it to a file, we can read the file, later on no harm done ok. Let us not bother with this if you are using spider or something this will be very easy to do.

So, so far so good we have obtained this. Now we need to wrap everything. So, in order to wrap everything we will have to wrap all of this inside the appropriate program right.

(Refer Slide Time: 10:39)

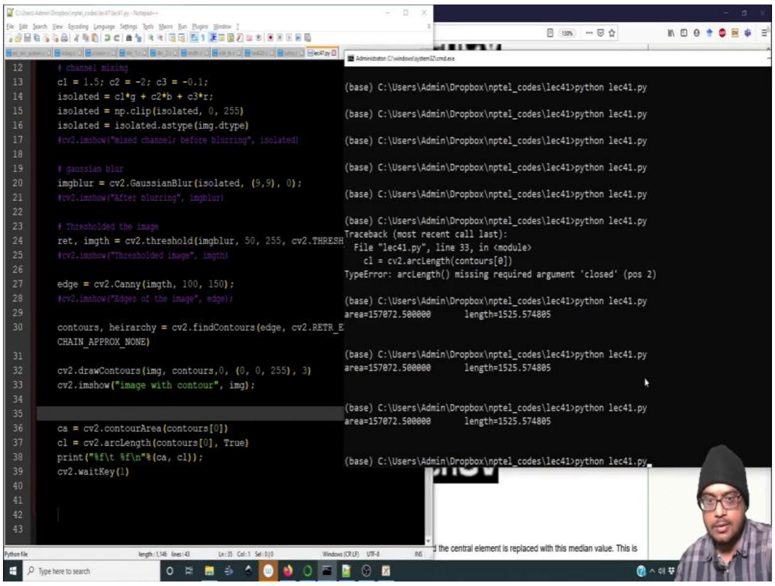


```
import cv2
import numpy as np
import matplotlib.pyplot as plt

vid = cv2.VideoCapture("balloon.mp4");

while True:

ret, img = vid.read()
b, g, r = cv2.split(img.astype(float))

# channel mixing
cl = 1.5; c2 = -2; c3 = -0.1;
isolated = cl*g + c2*b + c3*r;
isolated = np.clip(isolated, 0, 255)
isolated = isolated.astype(img.dtype)
cv2.imshow("mixed channel; before blurring", isolated)

# gaussian blur
imgblur = cv2.GaussianBlur(isolated, (9,9), 0);
cv2.imshow("After blurring", imgblur)

# Thresholded the image
ret, imgth = cv2.threshold(imgblur, 50, 255, cv2.THRESH_BINARY)
cv2.imshow("Thresholded image", imgth)

edge = cv2.Canny(imgth, 100, 150);
cv2.imshow("Edges of the image", edge);

contours, heirarchy = cv2.findContours(edge, cv2.RETR_EXTERNAL, cv2.
CHAIN_APPROX_NONE)

cv2.drawContours(img, contours,0, (0, 0, 255), 3)
```

```
in\Dropbox\nptel_codes\lec41>python lec41.py

in\Dropbox\nptel_codes\lec41>python lec41.py

in\Dropbox\nptel_codes\lec41>python lec41.py

in\Dropbox\nptel_codes\lec41>python lec41.py

in\Dropbox\nptel_codes\lec41>python lec41.py

in\Dropbox\nptel_codes\lec41>python lec41.py
ent call last):
 line 33, in <module>
gth(contours[0])
h() missing required argument 'closed' (pos 2)

in\Dropbox\nptel_codes\lec41>python lec41.py
        length=1525.574805


in\Dropbox\nptel_codes\lec41>python lec41.py
        length=1525.574805


in\Dropbox\nptel_codes\lec41>python lec41.py
        length=1525.574805


in\Dropbox\nptel_codes\lec41>
```

So, forget about this. So, we are going to say while True we are going to take all of this we are going to indent it alright.

(Refer Slide Time: 10:57)

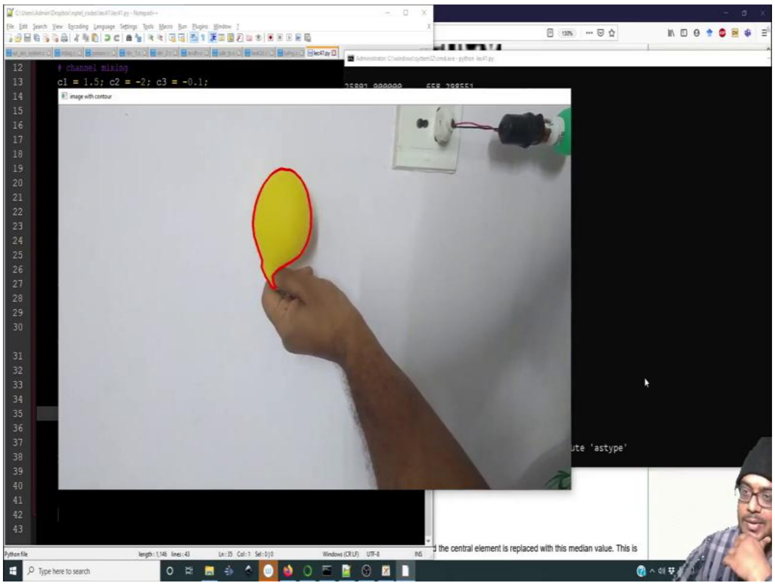


```
# channel mixing
c1 = 1.5; c2 = -2; c3 = -0.1;
isolated = c1*g + c2*b + c3*r;
isolated = np.clip(isolated, 0, 255)
isolated = isolated.astype(img.dtype)
#cv2.imshow("mixed channels before blurring", isolated)

# gaussian blur
imgblur = cv2.GaussianBlur(isolated, (9,9), 0);
#cv2.imshow("After blurring", imgblur)

# Thresholded the image
ret, imgth = cv2.threshold(imgblur, 50, 255, cv2.THRESH
#cv2.imshow("Thresholded image", imgth)

edge = cv2.Canny(imgth, 100, 150);
#cv2.imshow("Edges of the image", edge);

contours, heirarchy = cv2.findContours(edge, cv2.RETR_E
CHAIN_APPROX_NONE)

cv2.drawContours(img, contours,0, (0, 0, 255), 3)
cv2.imshow("image with contour", img);


ca = cv2.contourArea(contours[0])
cl = cv2.arcLength(contours[0], True)
print("%f\t %f\n"%(ca, cl));
cv2.waitKey(1)
```

```
(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py

(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py

(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py

(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py

(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py

(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
Traceback (most recent call last):
  File "lec41.py", line 33, in <module>
    cl = cv2.arcLength(contours[0])
TypeError: arcLength() missing required argument 'closed' (pos 2)

(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
area=157072.500000      length=1525.574805


(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
area=157072.500000      length=1525.574805


(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
area=157072.500000      length=1525.574805


(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
```

And instead of writing down area and length if we going to simply output the percentage f I mean the area and the lengths. This waitKey we will make it as 1 and we are going to only show the last image we do not want to show the edges and all because in the end the manifestation of all that is through the red contour on the original image. So, the final image should be showing the balloon with the red contour alright.

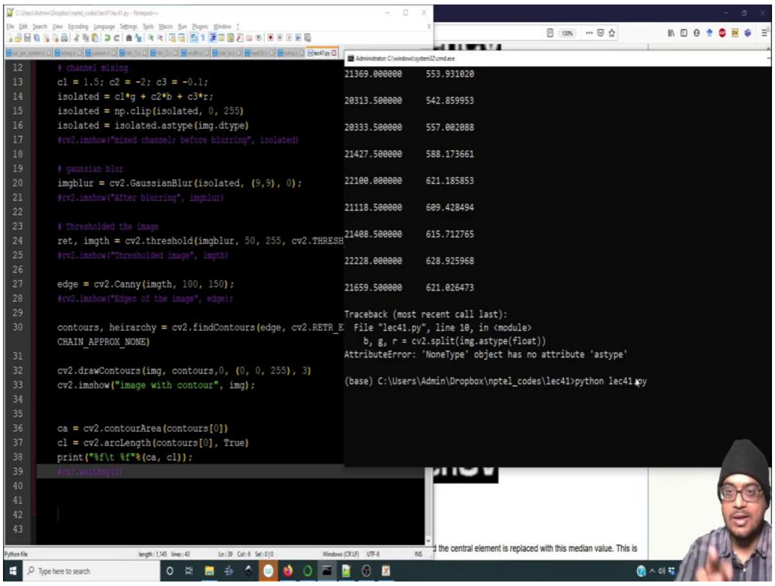
(Refer Slide Time: 11:48)



```
# channel mixing
c1 = 1.5; c2 = -2; c3 = -0.1;
```

So, let us see what happens amazing. So, you can see that sometimes the internal thing is also showing. So, we need to find out a way of avoiding that internal thing I mean for the most of the part it does not really cause an issue perfect. So, for most of the part it does not cause an issue, but when it does cause an issue how do you get rid of that? Well, while printing the contour 0s as such you do not face an error ok.

(Refer Slide Time: 12:18)



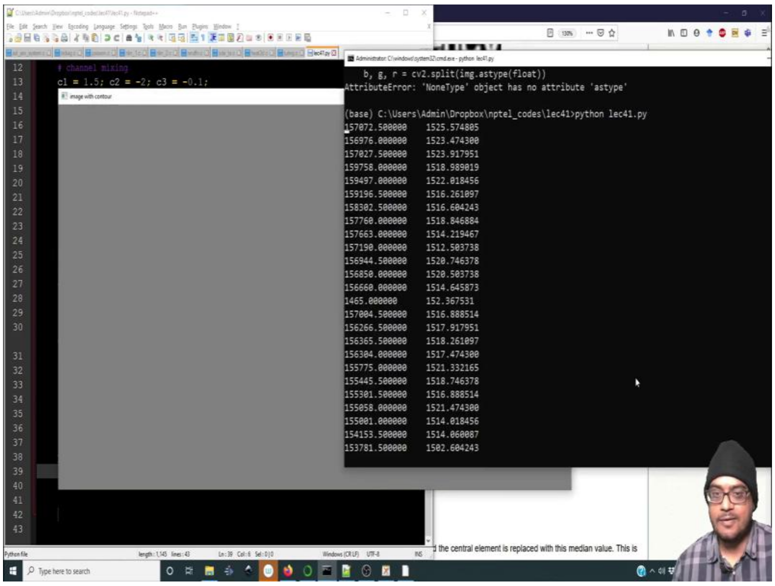
```
# channel mixing
c1 = 1.5; c2 = -2; c3 = -0.1;
isolated = c1*g + c2*b + c3*r;
isolated = np.clip(isolated, 0, 255)
isolated = isolated.astype(img.dtype)
#cv2.imshow("mixed channelr before blurring", isolated)

# gaussian blur
imgblur = cv2.GaussianBlur(isolated, (9,9), 0);
#cv2.imshow("After blurring", imgblur)

# Thresholded the image
ret, imgth = cv2.threshold(imgblur, 50, 255, cv2.THRESH
#cv2.imshow("Thresholded image", imgth)

edge = cv2.Canny(imgth, 100, 150);
#cv2.imshow("Edges of the image", edge);

contours, heirarchy = cv2.findContours(edge, cv2.RETR_E
CHAIN_APPROX_NONE)

cv2.drawContours(img, contours,0, (0, 0, 255), 3)
cv2.imshow("image with contour", img);


ca = cv2.contourArea(contours[0])
cl = cv2.arcLength(contours[0], True)
print("%f\t %f"%(ca, cl));
#cv2.waitKey(0)
```

```
21369.000000     553.931020

20313.500000     542.859953

20333.500000     557.002088

21427.500000     588.173661

22100.000000     621.185853

21118.500000     609.428494

21408.500000     615.712765

22228.000000     628.925968

21659.500000     621.026473

Traceback (most recent call last):
  File "lec41.py", line 10, in <module>
    b, g, r = cv2.split(img.astype(float))
AttributeError: 'NoneType' object has no attribute 'astype'

(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
```

the central element is replaced with this median value. This is

So, I tell you what, we are going to persist with this. Now how do you dump it to a file I am going to remove this percentage n.

(Refer Slide Time: 12:34)

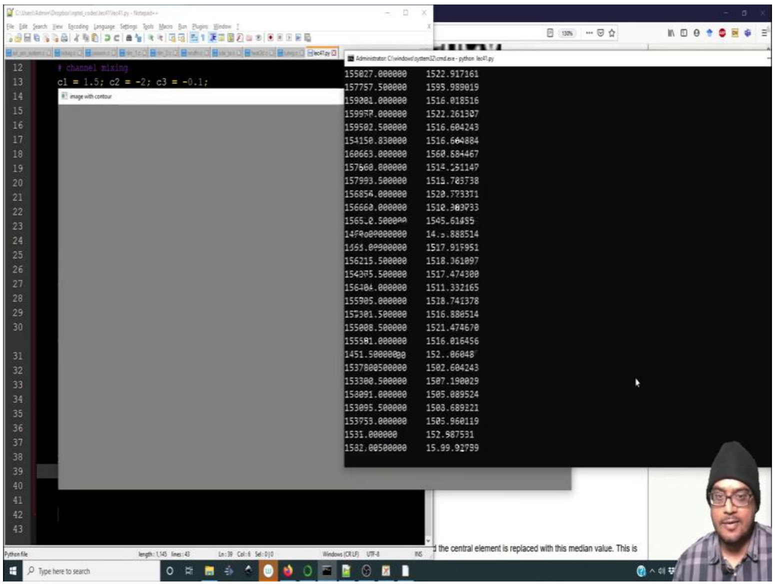


```
# channel mixing
c1 = 1.5; c2 = -2; c3 = -0.1;
```

```
    b, g, r = cv2.split(img.astype(float))
AttributeError: 'NoneType' object has no attribute 'astype'

(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
157072.500000    1525.574805
156976.000000    1523.474300
157027.500000    1523.917951
159758.000000    1518.989019
159497.000000    1522.018456
159196.500000    1516.261097
158302.500000    1516.604243
157760.000000    1518.846884
157663.000000    1514.219467
157190.000000    1512.503738
156944.500000    1520.746378
156850.000000    1520.503738
156660.000000    1514.645873
1465.000000      152.367531
157004.500000    1516.888514
156266.500000    1517.917951
156365.500000    1518.261097
156304.000000    1517.474300
155775.000000    1521.332165
155445.500000    1518.746378
155301.500000    1516.888514
155058.000000    1521.474300
155001.000000    1514.018456
154153.500000    1514.060087
153781.500000    1502.604243
```

the central element is replaced with this median value. This is

(Refer Slide Time: 12:36)



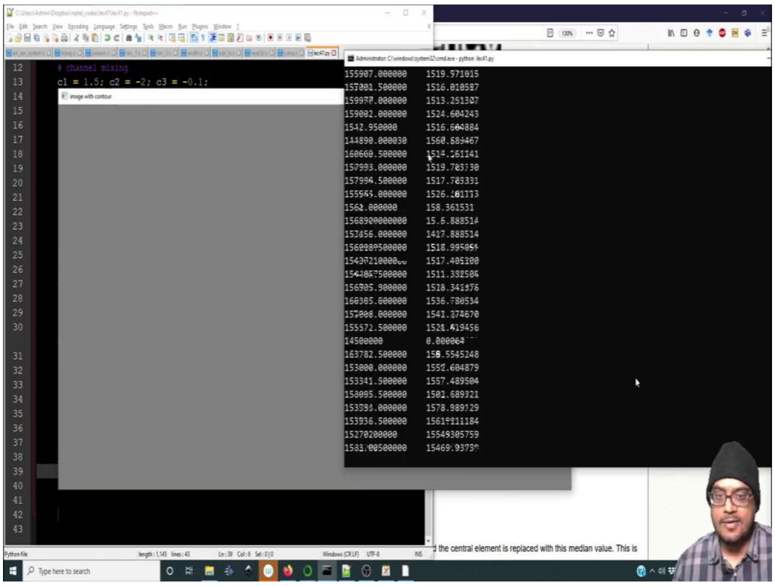
```
# channel mixing
c1 = 1.5; c2 = -2; c3 = -0.1;
```

Administrator: C:\windows\system32\cmd.exe - python lec41.py

```
155027.000000    1522.917161
157757.500000    1595.989019
159001.000000    1516.018516
159977.000000    1522.261307
159502.500000    1516.604243
154150.830000    1516.604884
160663.000000    1560.684467
157660.000000    1514.151147
157993.500000    1515.703738
156854.000000    1520.773371
156660.000000    1510.303733
1565.0.500000    1545.61455
1470000000000    14.5.888514
1555.09900000    1517.915951
156215.500000    1518.361097
154375.500000    1517.474300
156404.000000    1511.332165
155905.000000    1518.741378
157301.500000    1516.880514
155008.500000    1521.474670
155501.000000    1516.016456
1451.50000000    152..06048
1537800500000    1502.604243
153300.500000    1507.190029
150091.000000    1505.089524
153095.500000    1503.689221
153755.000000    1505.960119
1531.000000      152.987531
1582.00500000    15.99.92799
```

the central element is replaced with this median value. This is

(Refer Slide Time: 12:38)



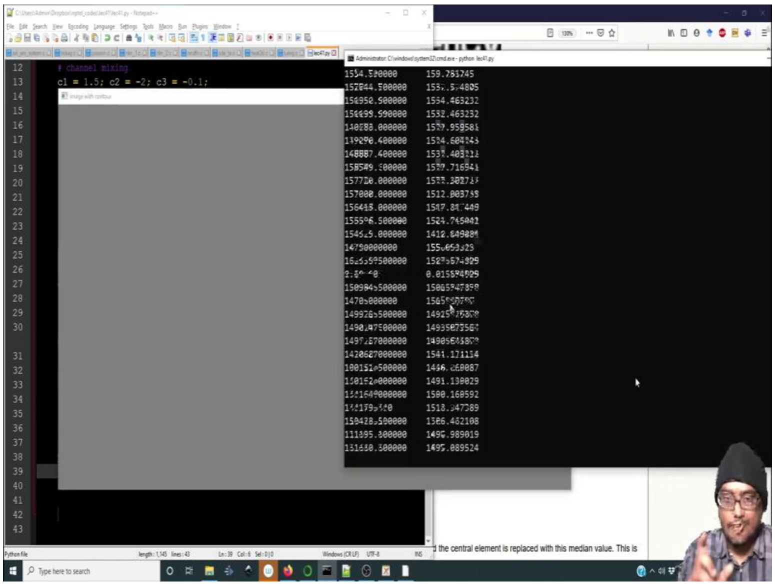
```
# channel mixing
c1 = 1.5; c2 = -2; c3 = -0.1;
```

Administrator: C:\windows\system32\cmd.exe - python lec41.py

```
155907.000000    1519.971015
157001.500000    1516.010587
159977.000000    1513.251307
159002.000000    1524.604243
1542.950000      1516.604884
144890.000030    1560.689467
160660.500000    1514.261141
157993.000000    1519.703730
157994.500000    1517.703331
155965.000000    1526.281773
1562.000000      158.361531
1568900000000    15.6.888514
157456.000000    1417.888514
1560080500000    1518.995059
15437210000000   1517.405100
1544087500000    1511.332504
156305.900000    1518.341376
160365.600000    1536.780534
157006.000000    1541.274670
155572.500000    1521.619456
14500000         0.000064
163782.500000    150.5545248
153000.000000    1552.604879
153341.500000    1557.489504
150095.500000    1502.689321
153394.000000    1578.989129
153936.500000    1561.911184
15270200000      15549305759
1583.00500000    15469.93739
```

the central element is replaced with this median value. This is

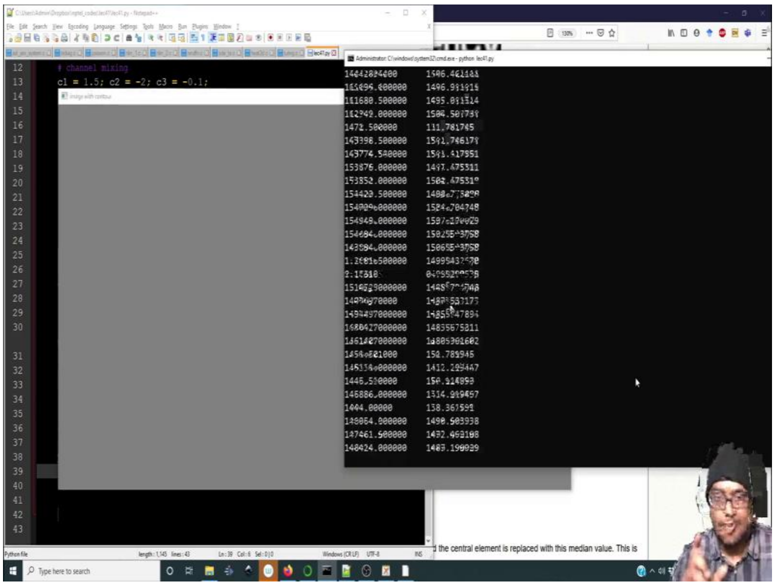
(Refer Slide Time: 12:39)



```
# channel mixing
c1 = 1.5; c2 = -2; c3 = -0.1;
```

```
1534.500000     159.781745
152844.500000   1537.574805
156950.500000   1554.463232
156499.590000   1532.463232
140288.000000   1537.959581
139290.400000   1534.604143
148887.400000   1537.403111
158549.500000   1537.716941
157720.000000   1533.302711
157000.000000   1512.003733
156418.000000   1517.817449
155596.500000   1524.745042
154525.000000   1412.849884
14750000000     1550653323
1625559500000   1527567489
2.59e+08        0.015574929
150984500000    150657473798
1470000000      156554975
149928500000    149259753078
149014750000    14935027356
149725700000    14905684387
142062700000    1541.171114
100151500000    1436.268087
150152000000    1491.130029
151164900000    1500.168592
14217935120     1518.347389
150428500000    1506.462108
111395.300000   1496.989019
131630.300000   1495.089524
```

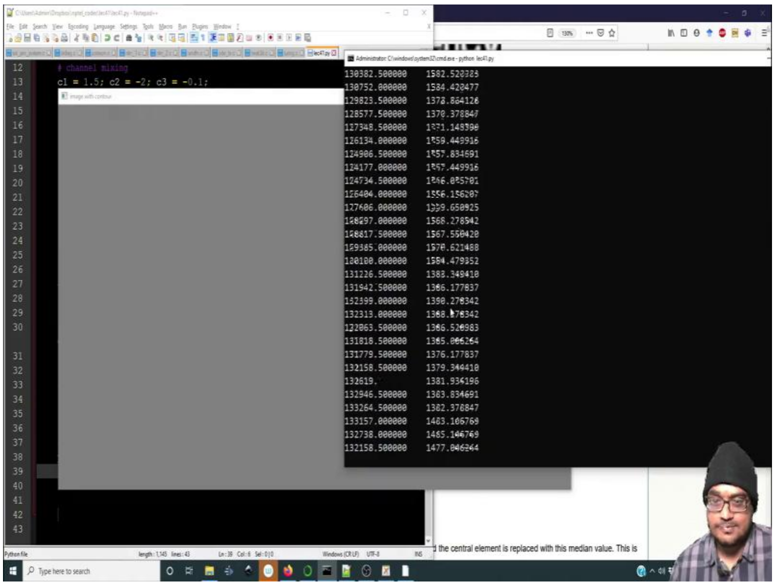
(Refer Slide Time: 12:39)



```
# channel mixing
c1 = 1.5; c2 = -2; c3 = -0.1;
```

```
14442824400     1506.461181
161695.600000   1496.911915
111680.500000   1495.011514
112749.000000   1504.501739
1472.500000     111.781745
143398.500000   1591.746179
143774.540000   1591.417951
153876.000000   1497.475311
153852.000000   1502.475319
154420.500000   1480.773820
154929000000    1524.704748
154949.000000   1597.170029
154684.000000   150255437358
143384.000000   15065543758
112081500000    149994370
2.15510         0.95520953
151462900000    14857797048
14874970000     14873533175
149449700000    14855447894
168842700000    14835675311
146142700000    14805301682
14580621000     150.789945
145134000000    1412.299447
1445.500000     150.914893
145886.000000   1314.919457
1444.00000      138.367591
140864.000000   1498.503938
147461.500000   1432.463108
148424.000000   1483.199939
```

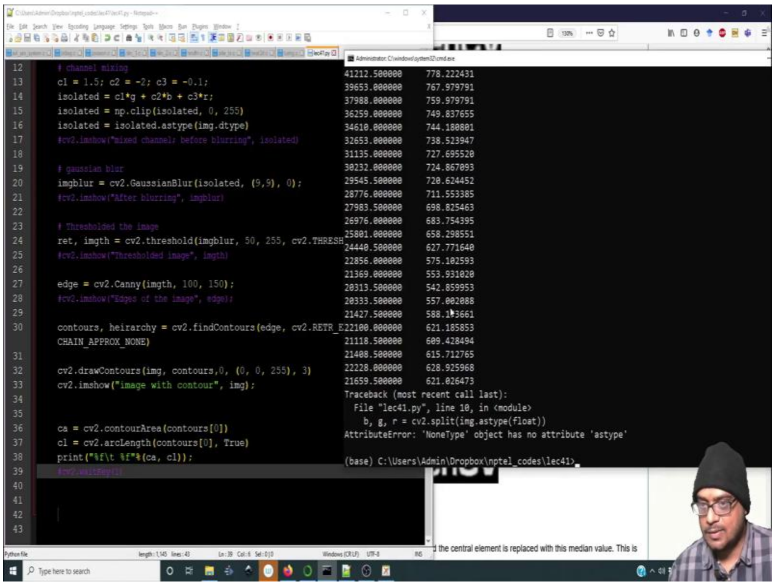
(Refer Slide Time: 12:44)



```
# channel mixing
c1 = 1.5; c2 = -2; c3 = -0.1;
```

```
130382.500000    1582.522323
130752.000000    1584.422477
129823.500000    1378.864128
128577.500000    1370.378847
127348.500000    1371.148390
126134.000000    1359.449916
124906.500000    1357.834691
124177.000000    1357.449916
124734.500000    1346.085701
126404.000000    1556.156207
127606.000000    1359.650925
128297.000000    1568.278542
128817.500000    1567.560420
129365.000000    1570.621488
130100.000000    1584.479352
131226.500000    1383.349410
131942.500000    1386.177837
132399.000000    1390.278342
132313.000000    1368.878342
132063.500000    1366.520983
131818.500000    1365.066264
131779.500000    1376.177837
132158.500000    1379.349410
132619.          1381.936196
132946.500000    1383.834691
133264.500000    1382.378847
133157.000000    1483.106769
132738.000000    1465.146769
132158.500000    1477.046264
```

(Refer Slide Time: 12:52)

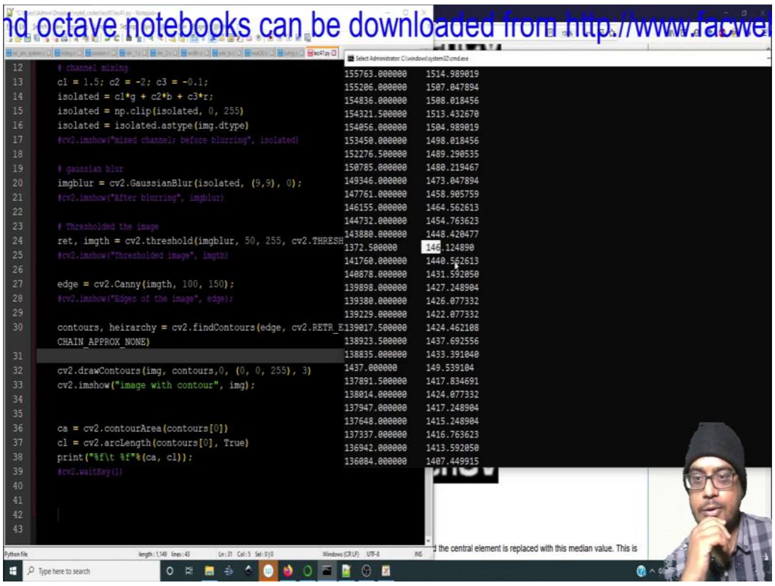


```
# channel mixing
c1 = 1.5; c2 = -2; c3 = -0.1;
isolated = c1*g + c2*b + c3*r;
isolated = np.clip(isolated, 0, 255)
isolated = isolated.astype(img.dtype)
#cv2.imshow("mixed channel; before blurring", isolated)

# gaussian blur
imgblur = cv2.GaussianBlur(isolated, (9,9), 0);
#cv2.imshow("After blurring", imgblur)

# Thresholded the image
ret, imgth = cv2.threshold(imgblur, 50, 255, cv2.THRESH
#cv2.imshow("Thresholded image", imgth)

edge = cv2.Canny(imgth, 100, 150);
#cv2.imshow("Edges of the image", edge);

contours, heirarchy = cv2.findContours(edge, cv2.RETR_E
CHAIN_APPROX_NONE)

cv2.drawContours(img, contours,0, (0, 0, 255), 3)
cv2.imshow("image with contour", img);


ca = cv2.contourArea(contours[0])
cl = cv2.arcLength(contours[0], True)
print("%f\t %f"%(ca, cl));
#cv2.waitKey(0)
```

```
41212.500000    778.222431
39653.000000    767.979791
37988.000000    759.979791
36259.000000    749.837655
34610.000000    744.180801
32653.000000    738.523947
31135.000000    727.695520
30232.000000    724.867093
29545.500000    720.624452
28776.000000    711.553385
27983.500000    698.825463
26976.000000    683.754395
25801.000000    658.298551
24440.500000    627.771640
22856.000000    575.102593
21369.000000    553.931020
20313.500000    542.859953
20333.500000    557.002088
21427.500000    588.143661
22100.000000    621.185853
21118.500000    609.428494
21408.500000    615.712765
22228.000000    628.925968
21659.500000    621.026473
Traceback (most recent call last):
  File "lec41.py", line 10, in <module>
    b, g, r = cv2.split(img.astype(float))
AttributeError: 'NoneType' object has no attribute 'astype'

(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>
```

I am going to remove this because I do not want to see the image. I just want to look at the values on the command line ok. Sometimes you do have that small contour appearing right. So, let us try to fix that let us try to fix that alright this code execute alright. So, over here we have the contours and the hierarchy now we need to extract the largest contour ok.

(Refer Slide Time: 13:17)

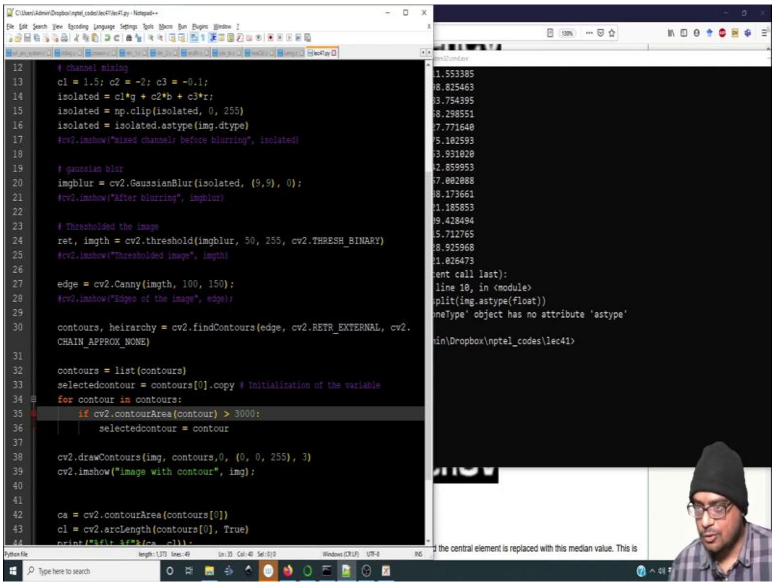

id octave notebooks can be downloaded from http://www.facwel
Select Administrator: C:\windows\system32\cmd.exe
the central element is replaced with this median value. This is

```
# channel mixing
c1 = 1.5; c2 = -2; c3 = -0.1;
isolated = c1*g + c2*b + c3*r;
isolated = np.clip(isolated, 0, 255)
isolated = isolated.astype(img.dtype)
#cv2.imshow("mixed channel: before blurring", isolated)

# gaussian blur
imgblur = cv2.GaussianBlur(isolated, (9,9), 0);
#cv2.imshow("After blurring", imgblur)

# Thresholded the image
ret, imgth = cv2.threshold(imgblur, 50, 255, cv2.THRESH
#cv2.imshow("Thresholded image", imgth)

edge = cv2.Canny(imgth, 100, 150);
#cv2.imshow("Edges of the image", edge);

contours, heirarchy = cv2.findContours(edge, cv2.RETR_E
CHAIN_APPROX_NONE)


cv2.drawContours(img, contours,0, (0, 0, 255), 3)
cv2.imshow("image with contour", img);


ca = cv2.contourArea(contours[0])
cl = cv2.arcLength(contours[0], True)
print("%f\t %f"%(ca, cl));
#cv2.waitKey(1)
```

```
155763.000000    1514.989019
155206.000000    1507.047894
154836.000000    1508.018456
154321.500000    1513.432670
154056.000000    1504.989019
153450.000000    1498.018456
152276.500000    1489.290535
150785.000000    1480.219467
149346.000000    1473.047894
147761.000000    1458.905759
146155.000000    1464.562613
144732.000000    1454.763623
143880.000000    1448.420477
1372.500000      146.124890
141760.000000    1440.562613
140878.000000    1431.592050
139898.000000    1427.248904
139380.000000    1426.077332
139229.000000    1422.077332
139017.500000    1424.462108
138923.500000    1437.692556
138835.000000    1433.391040
1437.000000      149.539104
137891.500000    1417.834691
138014.000000    1424.077332
137947.000000    1417.248904
137648.000000    1415.248904
137337.000000    1416.763623
136942.000000    1413.592050
136084.000000    1407.449915
```

So, before that let us see some whether we can find some common trend, when we have the smaller contour the area of that smaller contour is quite small. So, it is like 1372, 1445, 1451 and so on. So, the larger contour is significantly larger than the smaller contour. So, that gives us a clue on how we can isolate the larger contour ok.

(Refer Slide Time: 13:49)


C:\Users\Admin\Dropbox\nptel_codes\lec41\lec41.py - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
the central element is replaced with this median value. This is

```
# channel mixing
c1 = 1.5; c2 = -2; c3 = -0.1;
isolated = c1*g + c2*b + c3*r;
isolated = np.clip(isolated, 0, 255)
isolated = isolated.astype(img.dtype)
#cv2.imshow("mixed channel: before blurring", isolated)

# gaussian blur
imgblur = cv2.GaussianBlur(isolated, (9,9), 0);
#cv2.imshow("After blurring", imgblur)

# Thresholded the image
ret, imgth = cv2.threshold(imgblur, 50, 255, cv2.THRESH_BINARY)
#cv2.imshow("Thresholded image", imgth)

edge = cv2.Canny(imgth, 100, 150);
#cv2.imshow("Edges of the image", edge);

contours, heirarchy = cv2.findContours(edge, cv2.RETR_EXTERNAL, cv2.
CHAIN_APPROX_NONE)

contours = list(contours)
selectedcontour = contours[0].copy # Initialization of the variable
for contour in contours:
    if cv2.contourArea(contour) > 3000:
        selectedcontour = contour

cv2.drawContours(img, contours,0, (0, 0, 255), 3)
cv2.imshow("image with contour", img);


ca = cv2.contourArea(contours[0])
cl = cv2.arcLength(contours[0], True)
print("%f\t %f"%(ca, cl));
```

```
11.553385
98.825463
83.754395
58.298551
27.771640
75.102593
53.931020
42.859953
57.002088
88.173661
21.185853
09.428494
15.712765
28.925968
21.026473
ent call last):
 line 10, in <module>
split(img.astype(float))
oneType' object has no attribute 'astype'

in\Dropbox\nptel_codes\lec41>
```

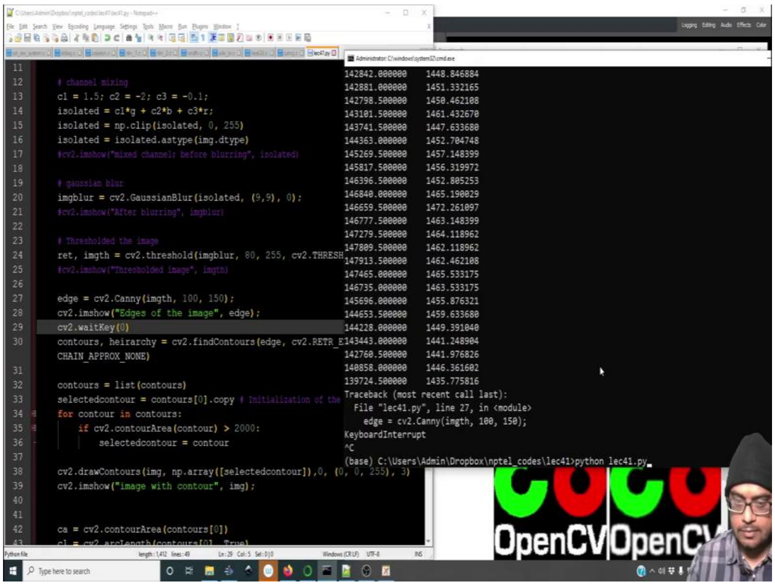
So, once we have obtained the contours we are going to take the contours and convert it to a list contours equal to list contours after converting it to a list. Let us take the selected contour or let us take the 0th contour and assign it to a selectedcontour. So, in the end the selectedcontour will be the largest contour, but we can assign selectedcontour default as the 0th contours 0 ok dot copy we going to make a copy of it.

So, now for contour in contours that is it will loop over the different i ds of the contours. We will say if if cv2.contourArea(contour) > 2000 then selectedcontour = contour alright. So, it means that if that particular contour is having, so this is basically initialization of the variable initialization of the variable. It has I mean you could do without this as well.

Then once you have initialized if you find the contour in the contours list which has an area larger than 2000 you make it as a related contour. Well 2000 also I mean we could make it very well as 3000, well towards the end what is the area let us see. We do not want to discard those towards the end it is 21000. So, we can make it 3000 as well.

So, then it is the selected contour and once we found this, we want to plot only the selectedcontour. So, contours comma ok as so it turns out when I was using a smaller kernel for the Gaussian blur that is the smaller area for the Gaussian blur it was giving me lot of isolated contours as well.

(Refer Slide Time: 16:15)

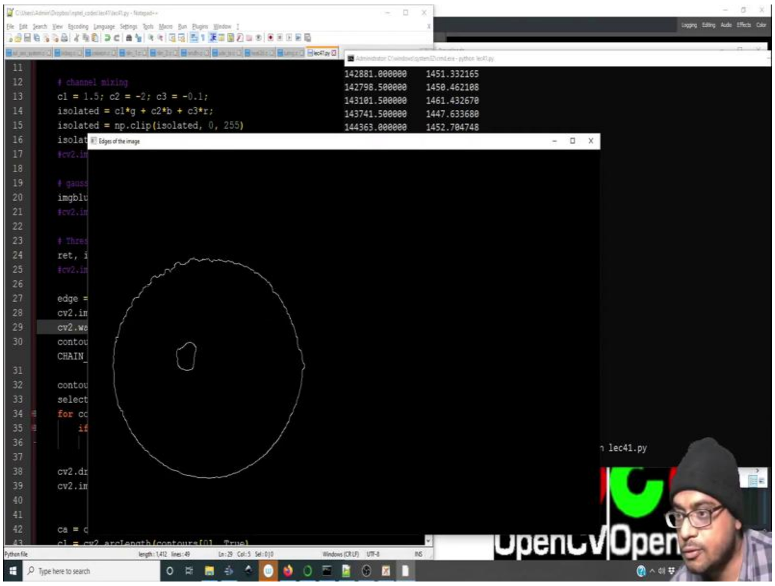


```

    # channel mixing
    c1 = 1.5; c2 = -2; c3 = -0.1;
    isolated = c1*g + c2*b + c3*r;
    isolated = np.clip(isolated, 0, 255)
    isolated = isolated.astype(img.dtype)
    #cv2.imshow("mixed channel; before blurring", isolated)

    # gaussian blur
    imgblur = cv2.GaussianBlur(isolated, (9,9), 0);
    #cv2.imshow("After blurring", imgblur)

    # Thresholded the image
    ret, imgth = cv2.threshold(imgblur, 80, 255, cv2.THRESH
    #cv2.imshow("Thresholded image", imgth)

    edge = cv2.Canny(imgth, 100, 150);
    cv2.imshow("Edges of the image", edge);
    cv2.waitKey(0)
    contours, heirarchy = cv2.findContours(edge, cv2.RETR_E
    CHAIN_APPROX_NONE)

    contours = list(contours)
    selectedcontour = contours[0].copy # Initialization of the
    for contour in contours:
        if cv2.contourArea(contour) > 2000:
            selectedcontour = contour

    cv2.drawContours(img, np.array([selectedcontour]),0, (0, 0, 255), 3)
    cv2.imshow("image with contour", img);


    ca = cv2.contourArea(contours[0])
    cl = cv2.arcLength(contours[0], True)
```

```
142842.000000    1448.846884
142881.000000    1451.332165
142798.500000    1450.462108
143101.500000    1461.432670
143741.500000    1447.633680
144363.000000    1452.704748
145269.500000    1457.148399
145817.500000    1456.319972
146396.500000    1452.805253
146840.000000    1465.190029
146659.500000    1472.261097
146777.500000    1463.148399
147279.500000    1464.118962
147809.500000    1462.118962
147913.500000    1462.462108
147465.000000    1465.533175
146735.000000    1463.533175
145696.000000    1455.876321
144653.500000    1459.633680
144228.000000    1449.391040
143443.000000    1441.248904
142760.500000    1441.976826
140858.000000    1446.361602
139724.500000    1435.775816
Traceback (most recent call last):
  File "lec41.py", line 27, in <module>
    edge = cv2.Canny(imgth, 100, 150);
KeyboardInterrupt
^C
(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
```

So, let me show you what I mean let me make it make the kernel size as 9 and let me show you the edge detected after making the kernel size 9.

(Refer Slide Time: 16:29)

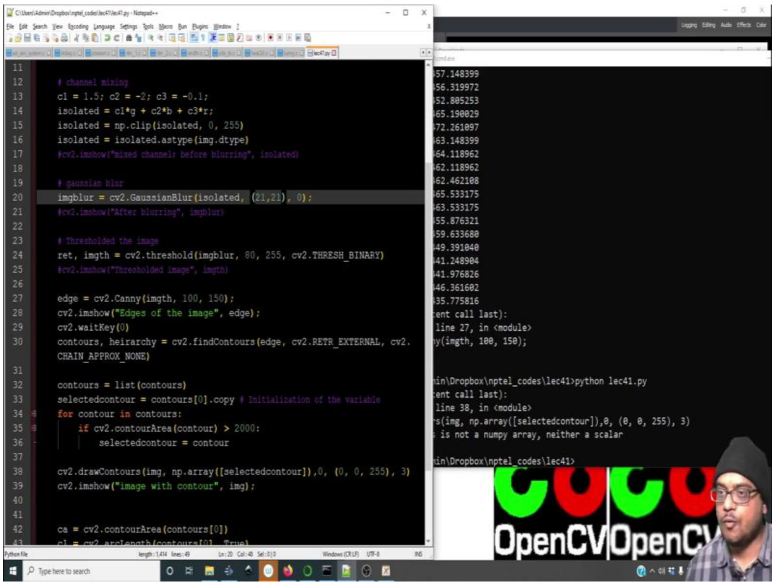


```
# channel mixing
c1 = 1.5; c2 = -2; c3 = -0.1;
isolated = c1*g + c2*b + c3*r;
isolated = np.clip(isolated, 0, 255)
isolat
#cv2.im

# gauss
imgblu
#cv2.im

# Thres
ret, i
#cv2.im

edge =
cv2.im
cv2.wa
contou
CHAIN_

contou
select
for cc
    if

cv2.dr
cv2.im


ca = c
c1 = cv2.arcLength(contours[0], True)
```

```
142881.000000    1451.332165
142798.500000    1450.462108
143101.500000    1461.432670
143741.500000    1447.633680
144363.000000    1452.704748
```

So, let me run this. So, if you look closely I am not sure I will be able to zoom it go ahead there is a small contour pocket near this which is not actually joined ok and that gives us a bunch of broken contours which are not helping our cause.

(Refer Slide Time: 16:52)

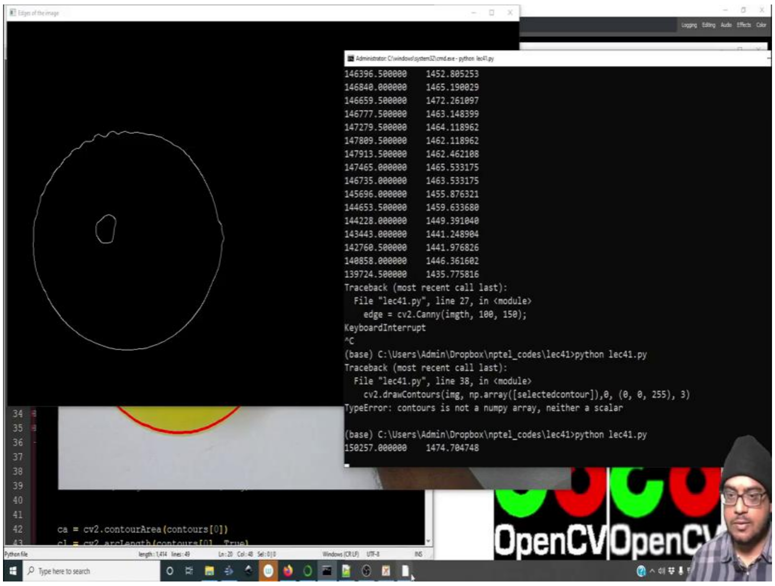


```
# channel mixing
c1 = 1.5; c2 = -2; c3 = -0.1;
isolated = c1*g + c2*b + c3*r;
isolated = np.clip(isolated, 0, 255)
isolated = isolated.astype(img.dtype)
#cv2.imshow("mixed channel: before blurring", isolated)

# gaussian blur
imgblur = cv2.GaussianBlur(isolated, (21,21), 0);
#cv2.imshow("After blurring", imgblur)

# Thresholded the image
ret, imgth = cv2.threshold(imgblur, 80, 255, cv2.THRESH_BINARY)
#cv2.imshow("Thresholded image", imgth)

edge = cv2.Canny(imgth, 100, 150);
cv2.imshow("Edges of the image", edge);
cv2.waitKey(0)
contours, heirarchy = cv2.findContours(edge, cv2.RETR_EXTERNAL, cv2.
CHAIN_APPROX_NONE)

contours = list(contours)
selectedcontour = contours[0].copy # Initialization of the variable
for contour in contours:
    if cv2.contourArea(contour) > 2000:
        selectedcontour = contour

cv2.drawContours(img, np.array([selectedcontour]),0, (0, 0, 255), 3)
cv2.imshow("image with contour", img);


ca = cv2.contourArea(contours[0])
c1 = cv2.arcLength(contours[0], True)
```

```
57.148399
56.319972
52.805253
65.190029
72.261097
63.148399
64.118962
62.118962
62.462108
65.533175
63.533175
55.876321
59.633680
49.391040
41.248904
41.976826
46.361602
35.775816
ent call last):
 line 27, in <module>
y(imgth, 100, 150);

in\Dropbox\nptel_codes\lec41>python lec41.py
ent call last):
 line 38, in <module>
s(img, np.array([selectedcontour]),0, (0, 0, 255), 3)
 is not a numpy array, neither a scalar

in\Dropbox\nptel_codes\lec41>
```

So, we need to blur it with a larger kernel here in this case its 21 after blurring we will remove those isolated pockets its more smooth now ok.

(Refer Slide Time: 16:58)

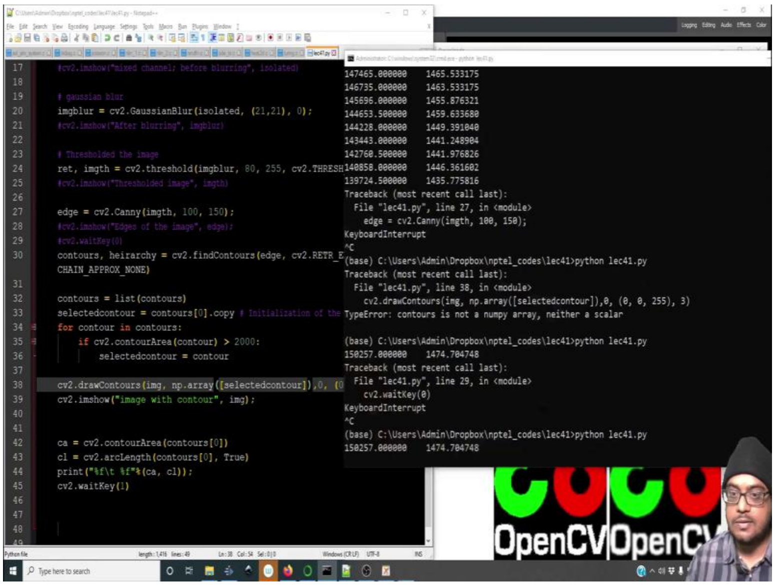


```
146396.500000    1452.805253
146840.000000    1465.190029
146659.500000    1472.261097
146777.500000    1463.148399
147279.500000    1464.118962
147809.500000    1462.118962
147913.500000    1462.462108
147465.000000    1465.533175
146735.000000    1463.533175
145696.000000    1455.876321
144653.500000    1459.633680
144228.000000    1449.391040
143443.000000    1441.248904
142760.500000    1441.976826
140858.000000    1446.361602
139724.500000    1435.775816
Traceback (most recent call last):
  File "lec41.py", line 27, in <module>
    edge = cv2.Canny(imgth, 100, 150);
KeyboardInterrupt
^C
(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
Traceback (most recent call last):
  File "lec41.py", line 38, in <module>
    cv2.drawContours(img, np.array([selectedcontour]),0, (0, 0, 255), 3)
TypeError: contours is not a numpy array, neither a scalar

(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
150257.000000    1474.704748
```

```
ca = cv2.contourArea(contours[0])
cl = cv2.arcLength(contours[0], True)
```

So, let me stop this. So, let me comment out the ok. So, what happens is you have the selectedcontour initialized in the contourArea is larger than 2000 you make the contour to be in the selected contour then you cast it to the form of an array and you plot the contour on top of the image.

(Refer Slide Time: 17:30)

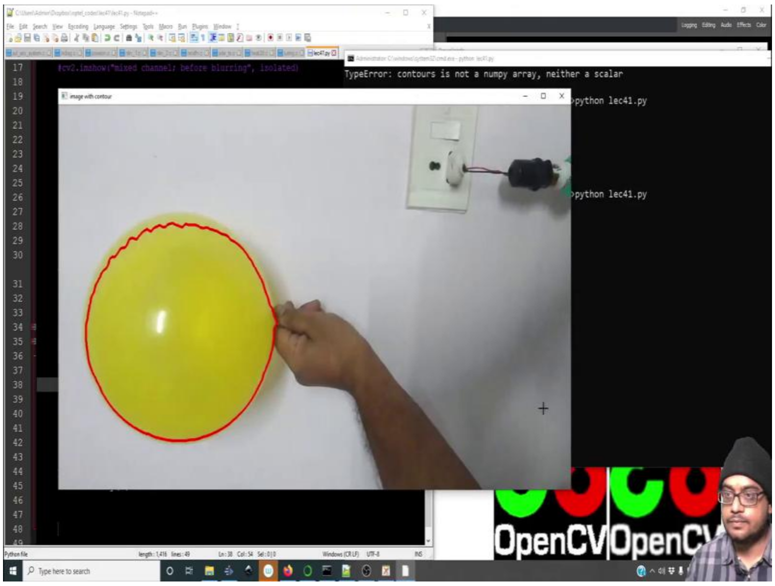


```
#cv2.imshow("mixed channel; before blurring", isolated)

# gaussian blur
imgblur = cv2.GaussianBlur(isolated, (21,21), 0);
#cv2.imshow("After blurring", imgblur)

# Thresholded the image
ret, imgth = cv2.threshold(imgblur, 80, 255, cv2.THRESH
#cv2.imshow("Thresholded image", imgth)

edge = cv2.Canny(imgth, 100, 150);
#cv2.imshow("Edges of the image", edge);
#cv2.waitKey(0)
contours, heirarchy = cv2.findContours(edge, cv2.RETR_E
CHAIN_APPROX_NONE)

contours = list(contours)
selectedcontour = contours[0].copy # Initialization of the
for contour in contours:
    if cv2.contourArea(contour) > 2000:
        selectedcontour = contour

cv2.drawContours(img, np.array([selectedcontour]),0, (0
cv2.imshow("image with contour", img);


ca = cv2.contourArea(contours[0])
cl = cv2.arcLength(contours[0], True)
print("%f\t %f"%(ca, cl));
cv2.waitKey(1)
```

```
147465.000000    1465.533175
146735.000000    1463.533175
145696.000000    1455.876321
144653.500000    1459.633680
144228.000000    1449.391040
143443.000000    1441.248904
142760.500000    1441.976826
140858.000000    1446.361602
139724.500000    1435.775816
Traceback (most recent call last):
  File "lec41.py", line 27, in <module>
    edge = cv2.Canny(imgth, 100, 150);
KeyboardInterrupt
^C
(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
Traceback (most recent call last):
  File "lec41.py", line 38, in <module>
    cv2.drawContours(img, np.array([selectedcontour]),0, (0, 0, 255), 3)
TypeError: contours is not a numpy array, neither a scalar

(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
150257.000000    1474.704748
Traceback (most recent call last):
  File "lec41.py", line 29, in <module>
    cv2.waitKey(0)
KeyboardInterrupt
^C
(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
150257.000000    1474.704748
```

So, this is needed in order to plot the contour which was originally converted to a list. So, this helps it to convert back to the data structure that the contour is the draw contour function requires ok. So, once this is done we should be able to plot the largest contour.

(Refer Slide Time: 17:59)

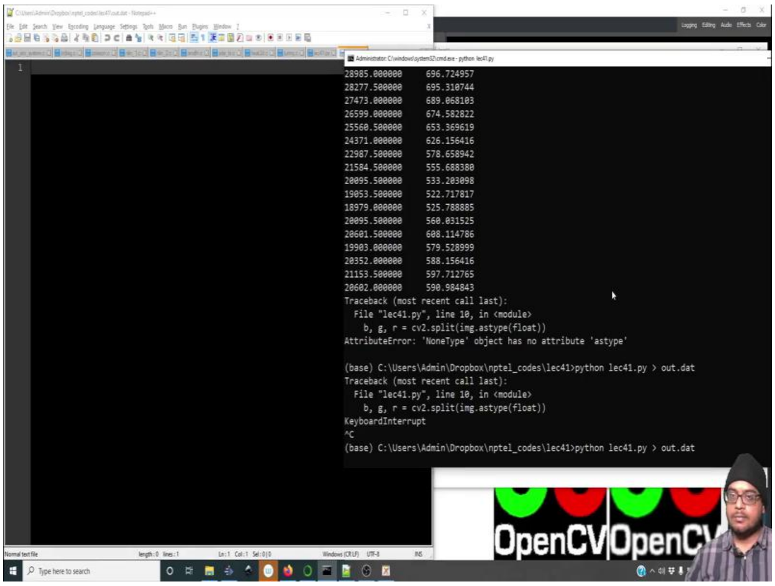


```
#cv2.imshow("mixed channel; before blurring", isolated)
```

```
TypeError: contours is not a numpy array, neither a scalar

>python lec41.py

>python lec41.py
```

So, let me run this everything seems to be going as per plan. Wow it is nice silky great alright. So, now, we are getting a bunch of outputs let us pipe that output to a data file. So, that later on you can plot it as well. So, in order to do that we will put it to out dot that. Let me stop plotting all this we do not need to plot once we a certain what needs to be done.

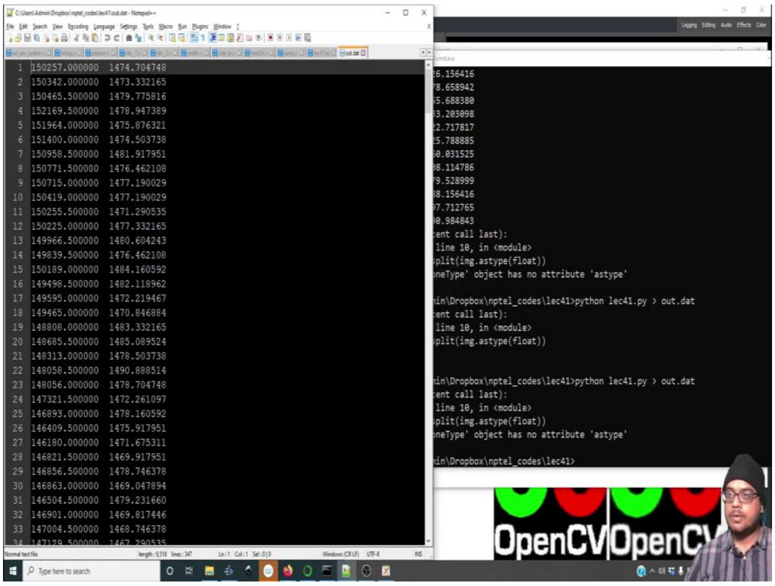
(Refer Slide Time: 18:56)



```
28985.000000    696.724957
28277.500000    695.310744
27473.000000    689.068103
26599.000000    674.582822
25560.500000    653.369619
24371.000000    626.156416
22987.500000    578.658942
21584.500000    555.688380
20095.500000    533.203098
19053.500000    522.717817
18979.000000    525.788885
20095.500000    560.031525
20601.500000    608.114786
19903.000000    579.528999
20352.000000    588.156416
21153.500000    597.712765
20602.000000    590.984843
Traceback (most recent call last):
  File "lec41.py", line 10, in <module>
    b, g, r = cv2.split(img.astype(float))
AttributeError: 'NoneType' object has no attribute 'astype'

(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py > out.dat
Traceback (most recent call last):
  File "lec41.py", line 10, in <module>
    b, g, r = cv2.split(img.astype(float))
KeyboardInterrupt
^C
(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py > out.dat
```

So, let me clear this file and let me re run it.

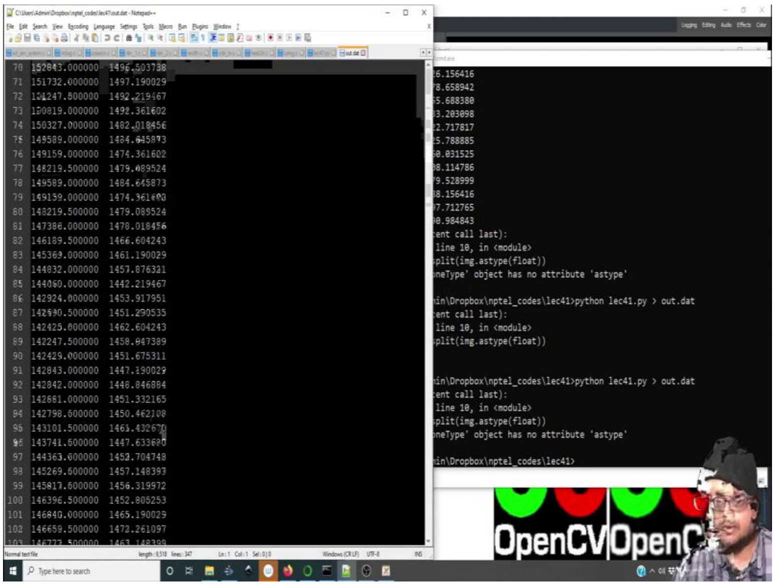
(Refer Slide Time: 19:29)



```
150257.000000  1474.704748
150342.000000  1473.332165
150465.500000  1479.775816
152169.500000  1478.947389
151964.000000  1475.876321
151400.000000  1474.503738
150958.500000  1481.917951
150771.500000  1476.462108
150715.000000  1477.190029
150419.000000  1477.190029
150255.500000  1471.290535
150225.000000  1477.332165
149966.500000  1480.604243
149839.500000  1476.462108
150189.000000  1484.160592
149498.500000  1482.118962
149595.000000  1472.219467
149465.000000  1470.846884
148808.000000  1483.332165
148685.500000  1485.089524
148313.000000  1478.503738
148058.500000  1490.888514
148056.000000  1478.704748
147321.500000  1472.261097
146893.000000  1478.160592
146409.500000  1475.917951
146180.000000  1471.675311
146821.500000  1469.917951
146856.500000  1478.746378
146863.000000  1469.047894
146504.500000  1479.231660
146901.000000  1469.817446
147004.500000  1468.746378
147129.500000  1467.290535
```

```
26.156416
78.658942
55.688380
33.203098
22.717817
25.788885
50.031525
98.114786
79.528999
38.156416
97.712765
00.984843
ent call last):
 line 10, in <module>
split(img.astype(float))
neType' object has no attribute 'astype'

in\Dropbox\nptel_codes\lec41>python lec41.py > out.dat
ent call last):
 line 10, in <module>
split(img.astype(float))


in\Dropbox\nptel_codes\lec41>python lec41.py > out.dat
ent call last):
 line 10, in <module>
split(img.astype(float))
neType' object has no attribute 'astype'

in\Dropbox\nptel_codes\lec41>
```

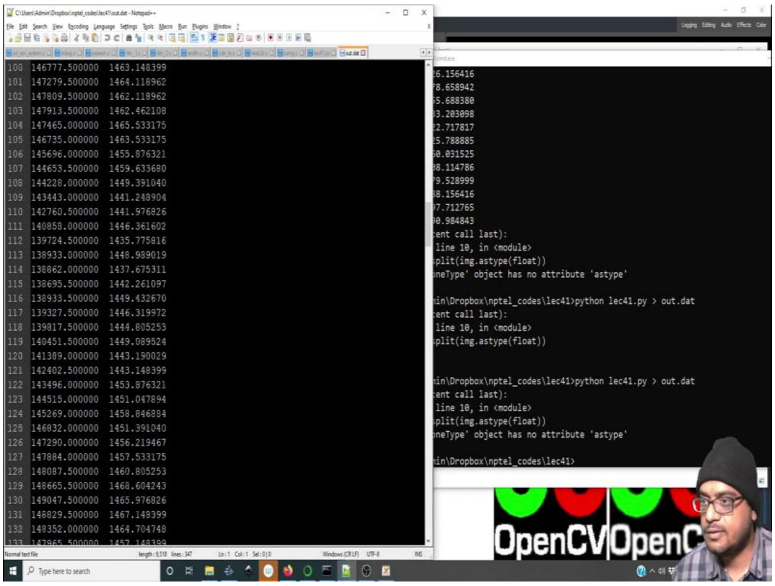
(Refer Slide Time: 19:30)



```
152843.000000  1496.503738
151732.000000  1497.190029
151247.500000  1492.219467
150819.000000  1492.361602
150327.000000  1482.018456
149589.000000  1484.645873
149159.000000  1474.361602
148219.500000  1479.089524
149589.000000  1484.645873
149159.000000  1474.361602
148219.500000  1479.089524
147386.000000  1478.018456
146189.500000  1466.604243
145369.000000  1461.190029
144832.000000  1457.876321
144060.000000  1442.219467
142924.000000  1453.917951
142590.500000  1451.290535
142425.000000  1462.604243
142247.500000  1458.047389
142429.000000  1451.675311
142843.000000  1447.190029
142842.000000  1448.846884
142681.000000  1451.332165
142798.500000  1450.462108
143101.500000  1461.432670
143741.500000  1447.633680
144363.500000  1452.704748
145269.500000  1457.148393
145817.500000  1456.319972
146396.500000  1452.805253
146840.000000  1465.190029
146659.500000  1472.261097
146777.500000  1463.148399
```

```
26.156416
78.658942
55.688380
33.203098
22.717817
25.788885
50.031525
98.114786
79.528999
38.156416
97.712765
00.984843
ent call last):
 line 10, in <module>
split(img.astype(float))
neType' object has no attribute 'astype'

in\Dropbox\nptel_codes\lec41>python lec41.py > out.dat
ent call last):
 line 10, in <module>
split(img.astype(float))


in\Dropbox\nptel_codes\lec41>python lec41.py > out.dat
ent call last):
 line 10, in <module>
split(img.astype(float))
neType' object has no attribute 'astype'

in\Dropbox\nptel_codes\lec41>
```

(Refer Slide Time: 19:30)

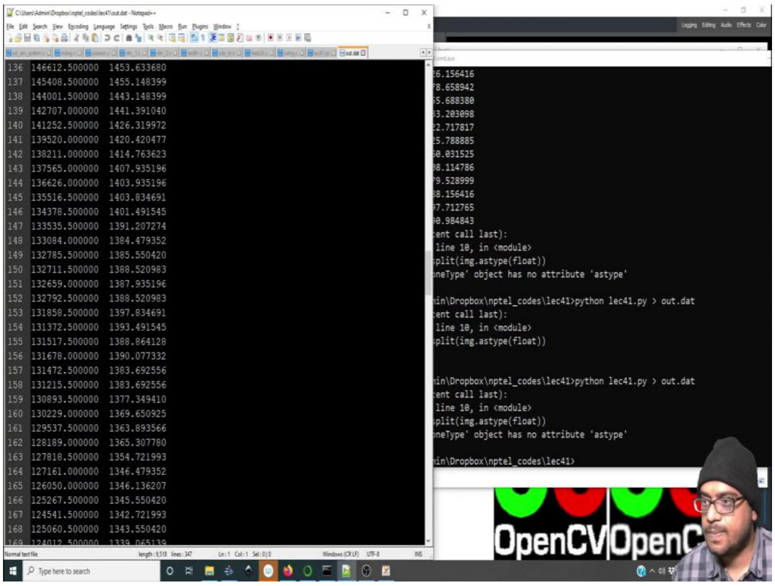


```
146777.500000  1463.148399
147279.500000  1464.118962
147809.500000  1462.118962
147913.500000  1462.462108
147465.000000  1465.533175
146735.000000  1463.533175
145696.000000  1455.876321
144653.500000  1459.633680
144228.000000  1449.391040
143443.000000  1441.248904
142760.500000  1441.976826
140858.000000  1446.361602
139724.500000  1435.775816
138933.000000  1448.989019
138862.000000  1437.675311
138695.500000  1442.261097
138933.500000  1449.432670
139327.500000  1446.319972
139817.500000  1444.805253
140451.500000  1449.089524
141389.000000  1443.190029
142402.500000  1443.148399
143496.000000  1453.876321
144515.000000  1451.047894
145269.000000  1458.846884
146832.000000  1451.391040
147290.000000  1456.219467
147884.000000  1457.533175
148087.500000  1460.805253
148665.500000  1468.604243
149047.500000  1465.976826
148829.500000  1467.148399
148352.000000  1464.704748
147965.500000  1457.148399
```

```
26.156416
78.658942
55.688380
33.203098
22.717817
25.788885
50.031525
98.114786
79.528999
88.156416
97.712765
90.984843
cent call last):
 line 10, in <module>
split(img.astype(float))
oneType' object has no attribute 'astype'

min\Dropbox\nptel_codes\lec41>python lec41.py > out.dat
cent call last):
 line 10, in <module>
split(img.astype(float))


min\Dropbox\nptel_codes\lec41>python lec41.py > out.dat
cent call last):
 line 10, in <module>
split(img.astype(float))
oneType' object has no attribute 'astype'

min\Dropbox\nptel_codes\lec41>
```

C:\Users\Admin\Dropbox\nptel_codes\lec41\out.dat - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
Normal text file
length : 9,518 lines : 347
Ln : 1 Col : 1 Sel : 0 | 0
Windows (CR LF) UTF-8
INS
Type here to search
Logging Editing Audio Effects Color
OpenCV OpenC

(Refer Slide Time: 19:31)

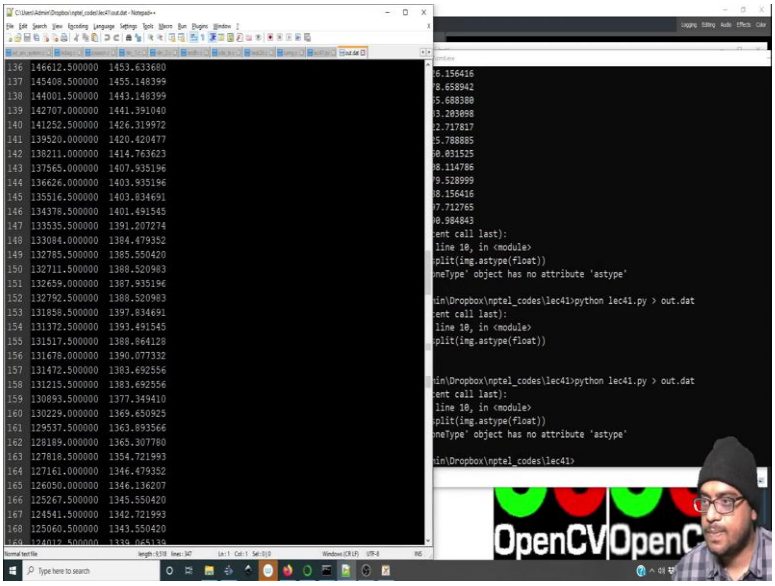


```
146612.500000  1453.633680
145408.500000  1455.148399
144001.500000  1443.148399
142707.000000  1441.391040
141252.500000  1426.319972
139520.000000  1420.420477
138211.000000  1414.763623
137565.000000  1407.935196
136626.000000  1403.935196
135516.500000  1403.834691
134378.500000  1401.491545
133535.500000  1391.207274
133084.000000  1384.479352
132785.500000  1385.550420
132711.500000  1388.520983
132659.000000  1387.935196
132792.500000  1388.520983
131858.500000  1397.834691
131372.500000  1393.491545
131517.500000  1388.864128
131678.000000  1390.077332
131472.500000  1383.692556
131215.500000  1383.692556
130893.500000  1377.349410
130229.000000  1369.650925
129537.500000  1363.893566
128189.000000  1365.307780
127818.500000  1354.721993
127161.000000  1346.479352
126050.000000  1346.136207
125267.500000  1345.550420
124541.500000  1342.721993
125060.500000  1343.550420
124012.500000  1339.065139
```

```
26.156416
78.658942
55.688380
33.203098
22.717817
25.788885
50.031525
98.114786
79.528999
88.156416
97.712765
90.984843
cent call last):
 line 10, in <module>
split(img.astype(float))
oneType' object has no attribute 'astype'

min\Dropbox\nptel_codes\lec41>python lec41.py > out.dat
cent call last):
 line 10, in <module>
split(img.astype(float))


min\Dropbox\nptel_codes\lec41>python lec41.py > out.dat
cent call last):
 line 10, in <module>
split(img.astype(float))
oneType' object has no attribute 'astype'

min\Dropbox\nptel_codes\lec41>
```

C:\Users\Admin\Dropbox\nptel_codes\lec41\out.dat - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
Normal text file
length : 9,518 lines : 347
Ln : 1 Col : 1 Sel : 0 | 0
Windows (CR LF) UTF-8
INS
Type here to search
Logging Editing Audio Effects Color
OpenCV OpenC

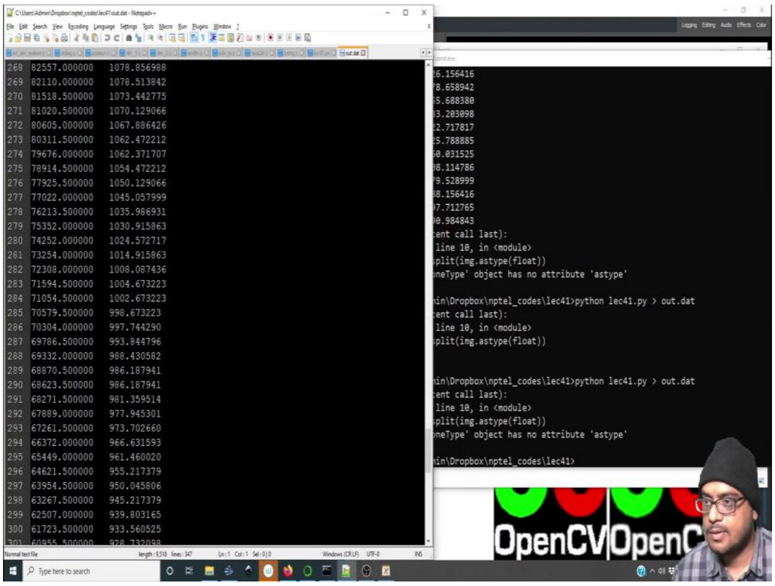
(Refer Slide Time: 19:32)



```
146612.500000  1453.633680
145408.500000  1455.148399
144001.500000  1443.148399
142707.000000  1441.391040
141252.500000  1426.319972
139520.000000  1420.420477
138211.000000  1414.763623
137565.000000  1407.935196
136626.000000  1403.935196
135516.500000  1403.834691
134378.500000  1401.491545
133535.500000  1391.207274
133084.000000  1384.479352
132785.500000  1385.550420
132711.500000  1388.520983
132659.000000  1387.935196
132792.500000  1388.520983
131858.500000  1397.834691
131372.500000  1393.491545
131517.500000  1388.864128
131678.000000  1390.077332
131472.500000  1383.692556
131215.500000  1383.692556
130893.500000  1377.349410
130229.000000  1369.650925
129537.500000  1363.893566
128189.000000  1365.307780
127818.500000  1354.721993
127161.000000  1346.479352
126050.000000  1346.136207
125267.500000  1345.550420
124541.500000  1342.721993
125060.500000  1343.550420
124012.500000  1339.065139
```

```
26.156416
78.658942
55.688380
33.203098
22.717817
25.788885
50.031525
08.114786
79.528999
38.156416
97.712765
00.984843
cent call last):
 line 10, in <module>
split(img.astype(float))
oneType' object has no attribute 'astype'

min\Dropbox\nptel_codes\lec41>python lec41.py > out.dat
cent call last):
 line 10, in <module>
split(img.astype(float))


min\Dropbox\nptel_codes\lec41>python lec41.py > out.dat
cent call last):
 line 10, in <module>
split(img.astype(float))
oneType' object has no attribute 'astype'

min\Dropbox\nptel_codes\lec41>
```

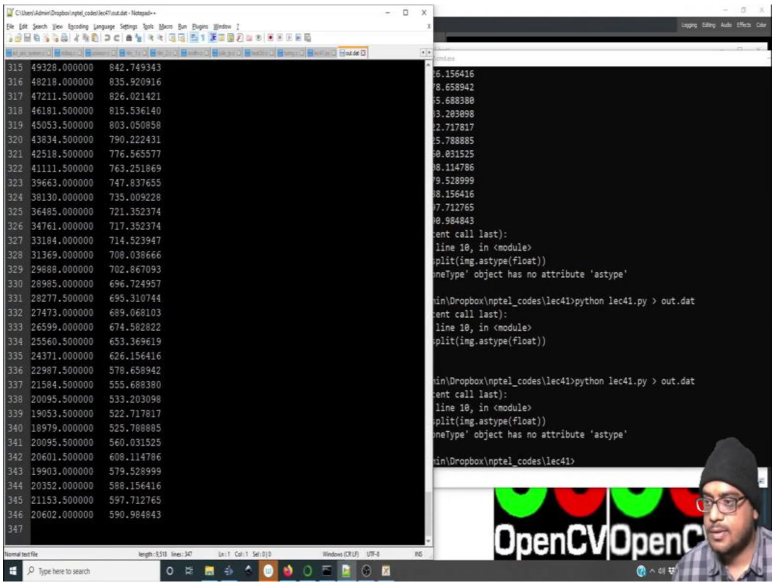
(Refer Slide Time: 19:34)



```
82557.000000   1078.856988
82110.000000   1078.513842
81518.500000   1073.442775
81020.500000   1070.129066
80605.000000   1067.886426
80311.500000   1062.472212
79676.000000   1062.371707
78914.500000   1054.472212
77925.500000   1050.129066
77022.000000   1045.057999
76213.500000   1035.986931
75352.000000   1030.915863
74252.000000   1024.572717
73254.000000   1014.915863
72308.000000   1008.087436
71594.500000   1004.673223
71054.500000   1002.673223
70579.500000   998.673223
70304.000000   997.744290
69786.500000   993.844796
69332.000000   988.430582
68870.500000   986.187941
68623.500000   986.187941
68271.500000   981.359514
67889.000000   977.945301
67261.500000   973.702660
66372.000000   966.631593
65449.000000   961.460020
64621.500000   955.217379
63954.500000   950.045806
63267.500000   945.217379
62507.000000   939.803165
61723.500000   933.560525
60955.500000   928.732098
```

```
26.156416
78.658942
55.688380
33.203098
22.717817
25.788885
50.031525
08.114786
79.528999
38.156416
97.712765
00.984843
cent call last):
 line 10, in <module>
split(img.astype(float))
oneType' object has no attribute 'astype'

min\Dropbox\nptel_codes\lec41>python lec41.py > out.dat
cent call last):
 line 10, in <module>
split(img.astype(float))


min\Dropbox\nptel_codes\lec41>python lec41.py > out.dat
cent call last):
 line 10, in <module>
split(img.astype(float))
oneType' object has no attribute 'astype'

min\Dropbox\nptel_codes\lec41>
```

(Refer Slide Time: 19:35)



C:\Users\Admin\Dropbox\nptel_codes\lec41\out.dat - Notepad++
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
Normal text file
length : 6,518 lines : 347
Ln : 1
Type here to search
Logging Editing Audio Effects Color
OpenCV OpenC

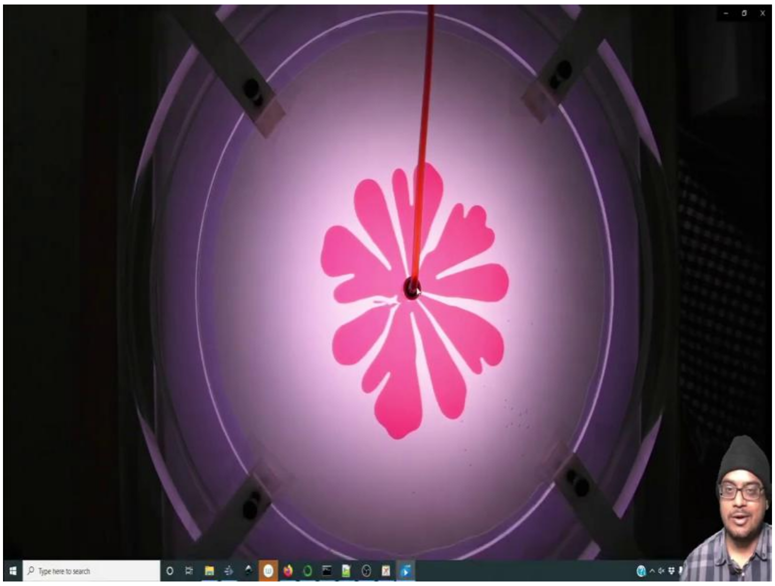
```
49328.000000   842.749343
48218.000000   835.920916
47211.500000   826.021421
46181.500000   815.536140
45053.500000   803.050858
43834.500000   790.222431
42518.500000   776.565577
41111.500000   763.251869
39663.000000   747.837655
38130.000000   735.009228
36485.000000   721.352374
34761.000000   717.352374
33184.000000   714.523947
31369.000000   708.038666
29888.000000   702.867093
28985.000000   696.724957
28277.500000   695.310744
27473.000000   689.068103
26599.000000   674.582822
25560.500000   653.369619
24371.000000   626.156416
22987.500000   578.658942
21584.500000   555.688380
20095.500000   533.203098
19053.500000   522.717817
18979.000000   525.788885
20095.500000   560.031525
20601.500000   608.114786
19903.000000   579.528999
20352.000000   588.156416
21153.500000   597.712765
20602.000000   590.984843
```

```
26.156416
78.658942
55.688380
33.203098
22.717817
25.788885
50.031525
08.114786
79.528999
88.156416
97.712765
90.984843
ent call last):
 line 10, in <module>
split(img.astype(float))
neType' object has no attribute 'astype'

in\Dropbox\nptel_codes\lec41>python lec41.py > out.dat
ent call last):
 line 10, in <module>
split(img.astype(float))


in\Dropbox\nptel_codes\lec41>python lec41.py > out.dat
ent call last):
 line 10, in <module>
split(img.astype(float))
neType' object has no attribute 'astype'

in\Dropbox\nptel_codes\lec41>
```

So, here we have all the areas and the perimeters ok. So, you can now load this file into python and plot it as well. So, with this in mind we have written a very easy problem. We have seen various concepts of blurring of isolation of mixing the channels finding out the edges finding out the contours ok.

The important thing is to smoothen out those edges otherwise then there will be small pockets of isolated contours which you do not want you want a nice single smooth contour. So, before concluding this lecture let me show you one more application of this for viscous fingering.

(Refer Slide Time: 20:29)

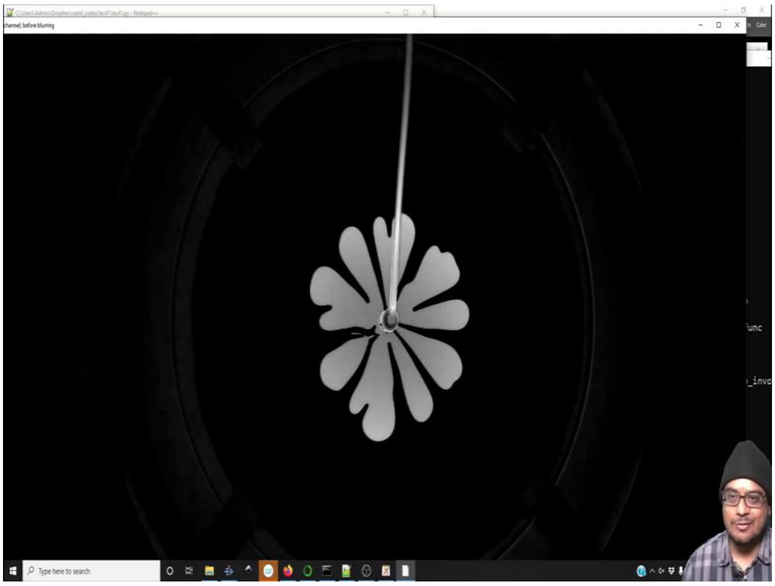


So, this is a video discourtesy of Dr. Saurab Mandal of the chemical engineering department and student Pooja Singh she is a PhD student and so, in this experiment viscous fluid is being displaced by a less viscous fluid and the less viscous fluid in this case is water and it is colored pink.

So, as you can imagine the initial channel mixing has to be such that the pink has to be more prominent than the background. So, let us see I have already encoded it with a bit of trial and error, the green channel has been biased to -2.12 the blue channel has been biased to 0.245 and the red channel has been biased to 1.359.

So, with the help of this I am just showing the channel before isolation the mixed channel. So, I am just showing the mix channel for now ok.

(Refer Slide Time: 21:36)



So, let me run this code ok. So, it takes a bit of time the video takes place after 6 seconds. So, you have to bear with me for with for this. So, let us let us just wait let us see what happens. So, as you can see the tube is quite prominent because the tube is carrying the liquid. Now once the liquid is being pushed you see that the front is nice and clear and once you threshold this it should be absolutely easy to find out the perimeter of the liquid ok.

The difficulty is this is and this is this has detached from the particular sample actually it will give you a bunch of contours. So, you have to loop over all contours to find out the perimeter I am not going to do that in this video, but maybe someone amongst you can do that anyway.

So, the point is once you find the appropriate bias of the channel, you can easily binarize the image. Now you can imagine how easy it will be to binarize this image because this is almost white everything is everything else is almost black ok. In addition to this because the tube is not moving you can sort of take the initial frame subtract the initial frame from all the frames to remove the tube, then you can sort of get a clear image of this fingering setup ok.

I am going to stop this over here. So, with this we end this particular lecture on image processing I know it is a bit power packed, but I have shown you all the important sort of workflow there is to most of the image processing problems. And I really hope you will find all this very useful.

Special thanks to Shreyas Darshan professor Saurab Mandal and Pooja Singh for contributing to this particular video and the previous one as well. So, with this I take leave I will see you again next time bye.