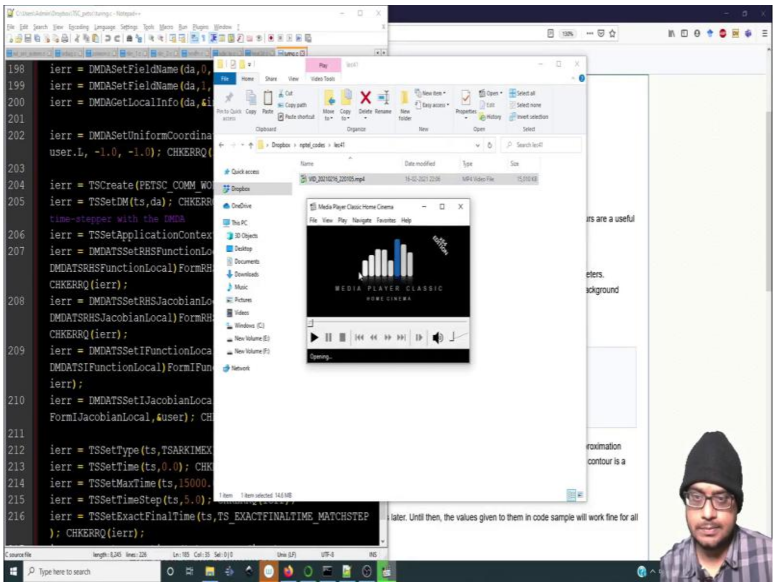
Tools in Scientific Computing

Prof. Aditya Bandopadhyay
Department of Mechanical Engineering
Indian Institute of Technology, Kharagpur

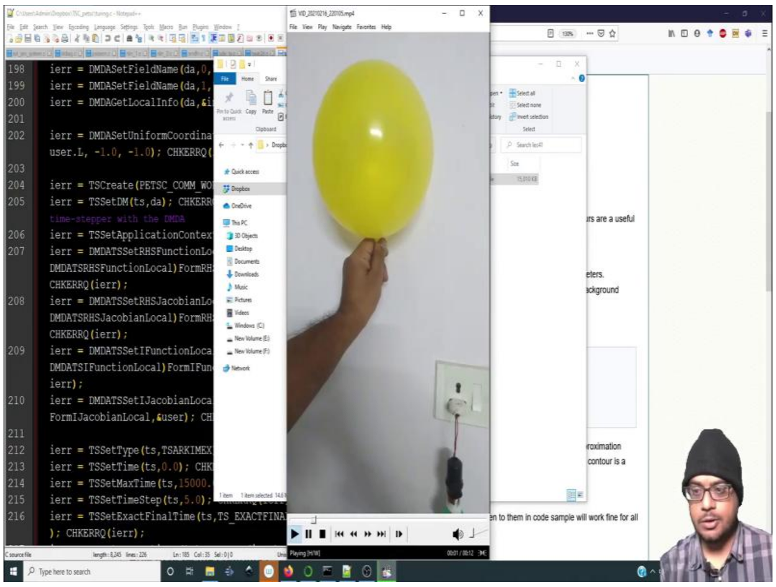
Lecture - 41
Image processing – preliminaries



```
ierr = DMDASetFieldName(da,0,
ierr = DMDASetFieldName(da,1,
ierr = DMDAGetLocalInfo(da,&i

ierr = DMDASetUniformCoordina
user.L, -1.0, -1.0); CHKERRQ(

ierr = TSCreate(PETSC_COMM_WO
ierr = TSSetDM(ts,da); CHKERR
time-stepper with the DMDA
ierr = TSSetApplicationContex
ierr = DMDATSSetRHSFunctionLo
DMDATSRHSFunctionLocal)FormRH
CHKERRQ(ierr);
ierr = DMDATSSetRHSJacobianLo
DMDATSRHSJacobianLocal)FormRH
CHKERRQ(ierr);
ierr = DMDATSSetIFunctionLoca
DMDATSIFunctionLocal)FormIFun
ierr);
ierr = DMDATSSetIJacobianLoca
FormIJacobianLocal,&user); CH

ierr = TSSetType(ts,TSARKIMEX
ierr = TSSetTime(ts,0.0); CHK
ierr = TSSetMaxTime(ts,15000.
ierr = TSSetTimeStep(ts,5.0);
ierr = TSSetExactFinalTime(ts,TS_EXACTFINALTIME_MATCHSTEP
); CHKERRQ(ierr);
```

Hi, everyone. In this lecture we are going to start with some Image Processing. So, let me show you a quick video.

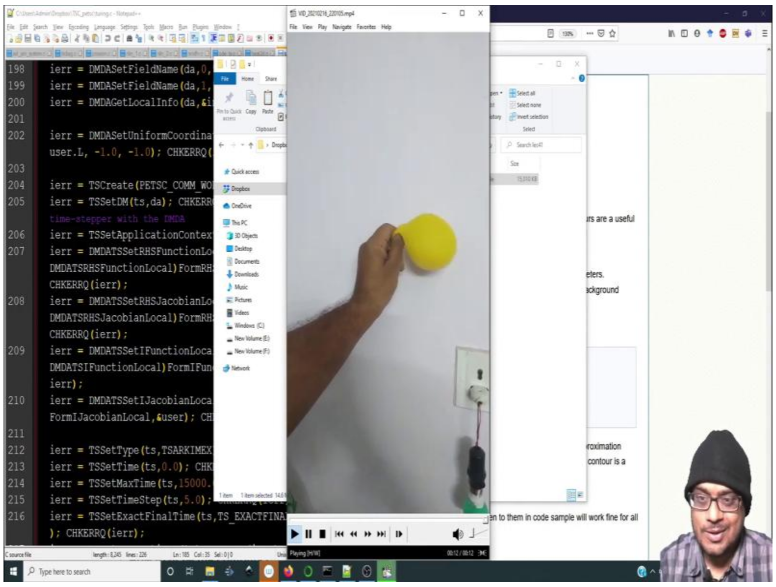
(Refer Slide Time: 00:40)



```
ierr = DMDASetFieldName(da,0,
ierr = DMDASetFieldName(da,1,
ierr = DMDAGetLocalInfo(da,&i

ierr = DMDASetUniformCoordina
user.L, -1.0, -1.0); CHKERRQ(

ierr = TSCreate(PETSC_COMM_WO
ierr = TSSetDM(ts,da); CHKERR
time-stepper with the DMDA
ierr = TSSetApplicationContex
ierr = DMDATSSetRHSFunctionLo
DMDATSRHSFunctionLocal)FormRH
CHKERRQ(ierr);
ierr = DMDATSSetRHSJacobianLo
DMDATSRHSJacobianLocal)FormRH
CHKERRQ(ierr);
ierr = DMDATSSetIFunctionLoca
DMDATSIFunctionLocal)FormIFun
ierr);
ierr = DMDATSSetIJacobianLoca
FormIJacobianLocal,&user); CH

ierr = TSSetType(ts,TSARKIMEX
ierr = TSSetTime(ts,0.0); CHK
ierr = TSSetMaxTime(ts,15000.
ierr = TSSetTimeStep(ts,5.0);
ierr = TSSetExactFinalTime(ts,TS_EXACTFINA
); CHKERRQ(ierr);
```

So, this is a video I took with my mobile phone of a deflating balloon.

(Refer Slide Time: 00:51)



```
ierr = DMDASetFieldName(da,0,
ierr = DMDASetFieldName(da,1,
ierr = DMDAGetLocalInfo(da,&i

ierr = DMDASetUniformCoordina
user.L, -1.0, -1.0); CHKERRQ(

ierr = TSCreate(PETSC_COMM_WO
ierr = TSSetDM(ts,da); CHKERR
time-stepper with the DMDA
ierr = TSSetApplicationContex
ierr = DMDATSSetRHSFunctionLo
DMDATSRHSFunctionLocal)FormRH
CHKERRQ(ierr);
ierr = DMDATSSetRHSJacobianLo
DMDATSRHSJacobianLocal)FormRH
CHKERRQ(ierr);
ierr = DMDATSSetIFunctionLoca
DMDATSIFunctionLocal)FormIFun
ierr);
ierr = DMDATSSetIJacobianLoca
FormIJacobianLocal,&user); CH

ierr = TSSetType(ts,TSARKIMEX
ierr = TSSetTime(ts,0.0); CHK
ierr = TSSetMaxTime(ts,15000.
ierr = TSSetTimeStep(ts,5.0);
ierr = TSSetExactFinalTime(ts,TS_EXACTFINA
); CHKERRQ(ierr);
```

So, it is a yellow coloured balloon ignore the light on the right, but essentially I fill the balloon with air and let it deflate and as trivial as it seems the question is how does the radius of the balloon change over time. And, I know what you are thinking I mean it is a very easy problem, but making the computer do this problem for you that is a bit of a challenge.

I have tried to make the video have a good contrast meaning visually speaking there has to be a good difference in the background and the foreground. So, obviously, the foreground consists of a bunch of things. It consists of my hand, it consists the balloon, it consists the switchboard, it consists the green light, it is a night lamp, but yeah it does consist of a bunch of things.

And visually we can clearly make out that the balloon is changing shape over time. And, additionally you can see the reflection of the tube light on the balloon as well over here. So, with all this the question is how do you make the computer do this? So, you may think ok I am going to find out the boundary of the balloon, find out the perimeter and do the thing like that.

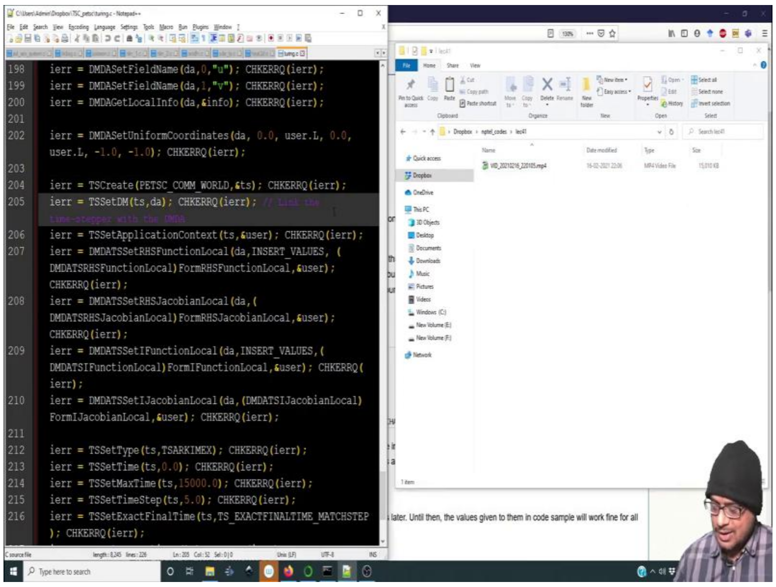
But, let us see how we can achieve this using Python. Most likely for this lecture we will not have something in octave we will just keep things in Python because OpenCV is a very robust library which helps us in image processing well. OpenCV originally is made for C++ once you require lot of speed you can start coding in C++, but in the end whatever you are doing in Python is actually calling the C++ routines or C++ programs in the backend.

So, it does not make a difference. And, this is just the starting of it I mean here it is a case of a balloon which is deflating. You could extend the same logic to for example, a vesicle. So, what is a vesicle? Vesicle is a membrane filled with fluid and if you have a solution which is hypertonic or hypotonic meaning the concentration of salt inside the in the fluid inside the membrane is different than the fluid outside.

Then either fluid will rush into the membrane causing it to swell or fluid will leave the membrane causing it to shrink such kinds of things. Once you observe under the microscope you can apply this kind of image processing techniques to analyze what their change of shape is and through this very simple example I will be demonstrating a bunch of image processing tools and I will leave it on to you to explore further.

And, such kinds of things are quite useful from the point of view of scientific computing because you know numeric's and experiments always go hand in hand. So, let me close this video. In this particular video we are not going to make use of the JupyterLab notebook environment, but rather we are going to directly write a code in Python.

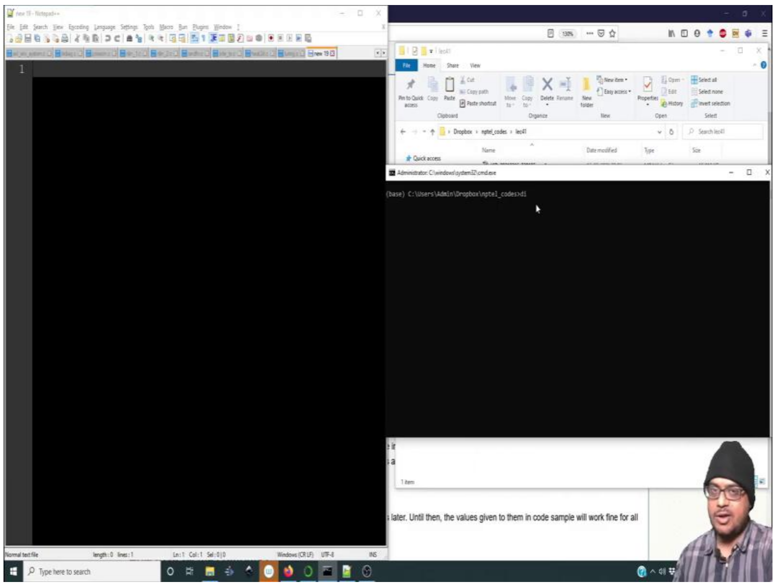
(Refer Slide Time: 04:28)



```
ierr = DMDASetFieldName(da,0,"u"); CHKERRQ(ierr);
ierr = DMDASetFieldName(da,1,"v"); CHKERRQ(ierr);
ierr = DMDAGetLocalInfo(da,&info); CHKERRQ(ierr);

ierr = DMDASetUniformCoordinates(da, 0.0, user.L, 0.0,
user.L, -1.0, -1.0); CHKERRQ(ierr);

ierr = TSCreate(PETSC_COMM_WORLD,&ts); CHKERRQ(ierr);
ierr = TSSetDM(ts,da); CHKERRQ(ierr); // Link the
time-stepper with the DMDA
ierr = TSSetApplicationContext(ts,&user); CHKERRQ(ierr);
ierr = DMDATSSetRHSFunctionLocal(da,INSERT_VALUES, (
DMDATSRHSFunctionLocal)FormRHSFunctionLocal,&user);
CHKERRQ(ierr);
ierr = DMDATSSetRHSJacobianLocal(da,(
DMDATSRHSJacobianLocal)FormRHSJacobianLocal,&user);
CHKERRQ(ierr);
ierr = DMDATSSetIFunctionLocal(da,INSERT_VALUES,(
DMDATSIFunctionLocal)FormIFunctionLocal,&user); CHKERRQ(
ierr);
ierr = DMDATSSetIJacobianLocal(da,(DMDATSIJacobianLocal)
FormIJacobianLocal,&user); CHKERRQ(ierr);

ierr = TSSetType(ts,TSARKIMEX); CHKERRQ(ierr);
ierr = TSSetTime(ts,0.0); CHKERRQ(ierr);
ierr = TSSetMaxTime(ts,15000.0); CHKERRQ(ierr);
ierr = TSSetTimeStep(ts,5.0); CHKERRQ(ierr);
ierr = TSSetExactFinalTime(ts,TS_EXACTFINALTIME_MATCHSTEP
); CHKERRQ(ierr);
```

Many of you have been wondering whether JupyterLab is required for doing all this. The answer is no. We can directly do stuff with simple plain text file as well a simple ASCII encoded file as well.

(Refer Slide Time: 04:44)

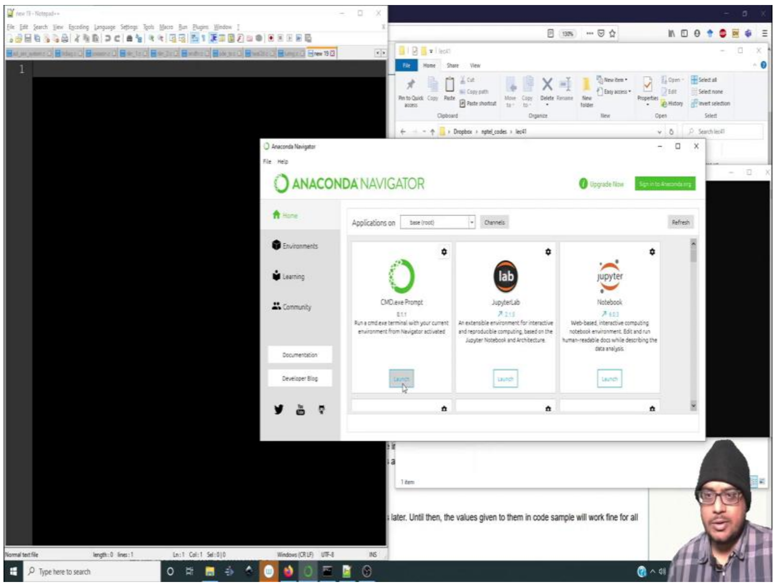


```
(base) C:\Users\Admin\Dropbox\nptel_codes>d1
```

later. Until then, the values given to them in code sample will work fine for all

So, to create a new file I have navigated to my folder.

(Refer Slide Time: 04:50)



Run a cmd.exe terminal with your current environment from Navigator activated

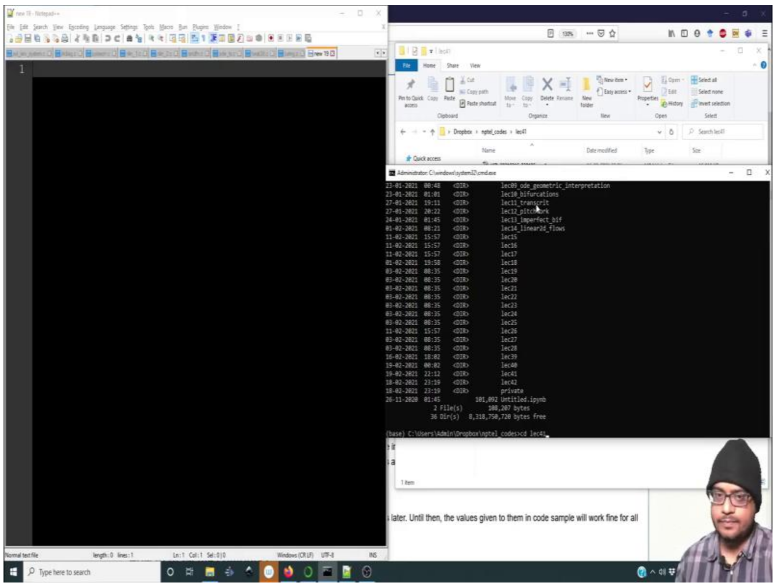
An extensible environment for interactive and reproducible computing, based on the Jupyter Notebook and Architecture.

Web-based, interactive computing notebook environment. Edit and run human-readable docs while describing the data analysis.

later. Until then, the values given to them in code sample will work fine for all

So, I have opened up the command prompt over here and I have navigated to the folder in which I have all my files.

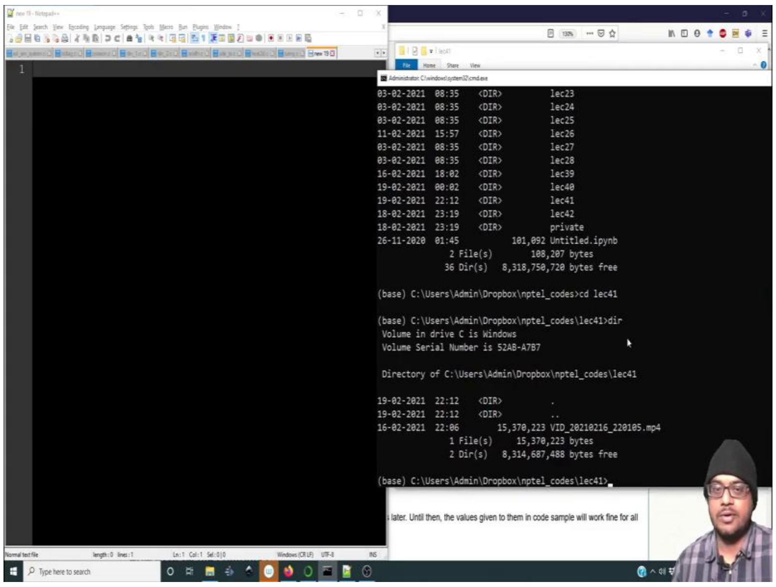
(Refer Slide Time: 04:58)



```
23-01-2021  00:48    <DIR>          lec09_ode_geometric_interpretation
23-01-2021  01:01    <DIR>          lec10_bifurcations
27-01-2021  19:11    <DIR>          lec11_transcrit
27-01-2021  20:22    <DIR>          lec12_pitchfork
24-01-2021  01:45    <DIR>          lec13_imperfect_bif
01-02-2021  08:21    <DIR>          lec14_linear2d_flows
11-02-2021  15:57    <DIR>          lec15
11-02-2021  15:57    <DIR>          lec16
11-02-2021  15:57    <DIR>          lec17
01-02-2021  19:58    <DIR>          lec18
03-02-2021  08:35    <DIR>          lec19
03-02-2021  08:35    <DIR>          lec20
03-02-2021  08:35    <DIR>          lec21
03-02-2021  08:35    <DIR>          lec22
03-02-2021  08:35    <DIR>          lec23
03-02-2021  08:35    <DIR>          lec24
03-02-2021  08:35    <DIR>          lec25
11-02-2021  15:57    <DIR>          lec26
03-02-2021  08:35    <DIR>          lec27
03-02-2021  08:35    <DIR>          lec28
16-02-2021  18:02    <DIR>          lec39
19-02-2021  00:02    <DIR>          lec40
19-02-2021  22:12    <DIR>          lec41
18-02-2021  23:19    <DIR>          lec42
18-02-2021  23:19    <DIR>          private
26-11-2020  01:45           101,092 Untitled.ipynb
               2 File(s)        108,207 bytes
              36 Dir(s)   8,318,750,720 bytes free

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

I am I am doing a DIR, I will go to cd lec 41 dir ok.

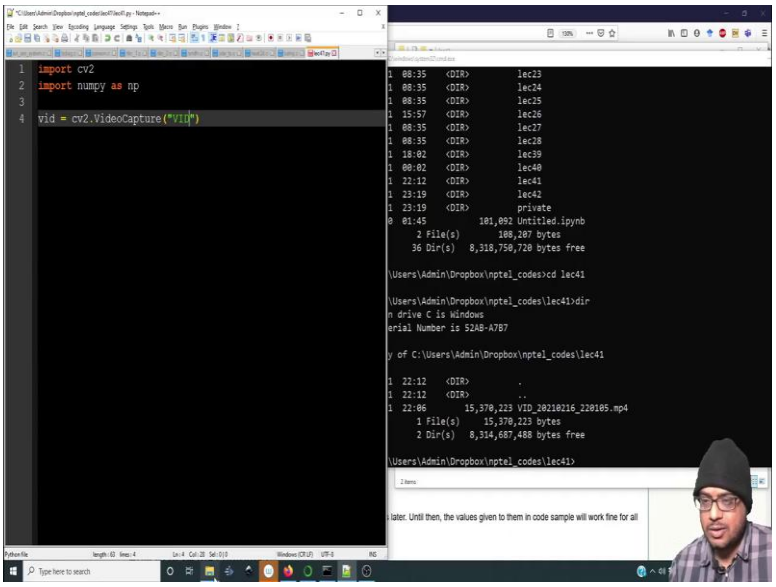
(Refer Slide Time: 05:04)



```
03-02-2021  08:35    <DIR>          lec23
03-02-2021  08:35    <DIR>          lec24
03-02-2021  08:35    <DIR>          lec25
11-02-2021  15:57    <DIR>          lec26
03-02-2021  08:35    <DIR>          lec27
03-02-2021  08:35    <DIR>          lec28
16-02-2021  18:02    <DIR>          lec39
19-02-2021  00:02    <DIR>          lec40
19-02-2021  22:12    <DIR>          lec41
18-02-2021  23:19    <DIR>          lec42
18-02-2021  23:19    <DIR>          private
26-11-2020  01:45           101,092 Untitled.ipynb
               2 File(s)        108,207 bytes
              36 Dir(s)   8,318,750,720 bytes free

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

(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>dir
 Volume in drive C is Windows
 Volume Serial Number is 52AB-A7B7

 Directory of C:\Users\Admin\Dropbox\nptel_codes\lec41

19-02-2021  22:12    <DIR>          .
19-02-2021  22:12    <DIR>          ..
16-02-2021  22:06        15,370,223 VID_20210216_220105.mp4
               1 File(s)     15,370,223 bytes
               2 Dir(s)   8,314,687,488 bytes free

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

So, it contains the file under question let me increase the size a bit so that you can ok. This is the local folder that we are working with alright fair enough. So, on the left I have opened up a new file. So, let us start writing the code.

(Refer Slide Time: 05:26)



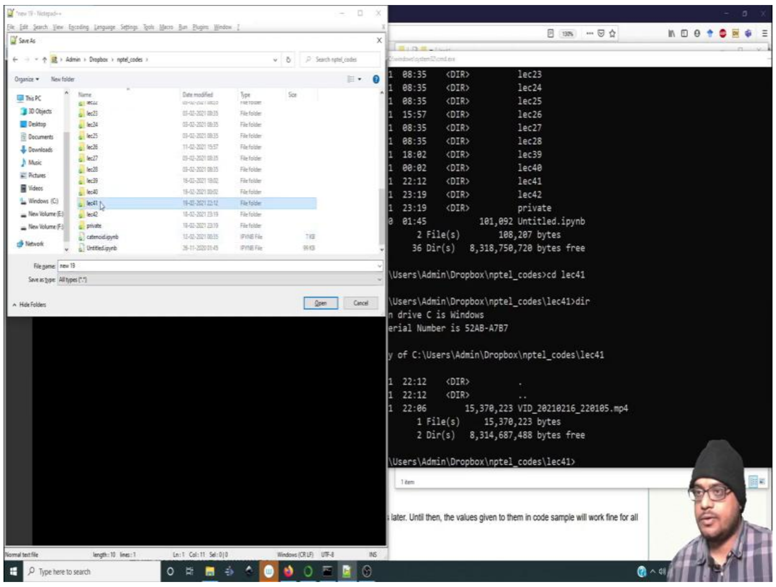
```python
import cv2
import numpy as np

vid = cv2.VideoCapture("VID")
```

```
1  08:35    <DIR>          lec23
1  08:35    <DIR>          lec24
1  08:35    <DIR>          lec25
1  15:57    <DIR>          lec26
1  08:35    <DIR>          lec27
1  08:35    <DIR>          lec28
1  18:02    <DIR>          lec39
1  00:02    <DIR>          lec40
1  22:12    <DIR>          lec41
1  23:19    <DIR>          lec42
1  23:19    <DIR>          private
0  01:45           101,092 Untitled.ipynb
       2 File(s)        108,207 bytes
      36 Dir(s)   8,318,750,720 bytes free

\Users\Admin\Dropbox\nptel_codes>cd lec41

\Users\Admin\Dropbox\nptel_codes\lec41>dir
n drive C is Windows
erial Number is 52AB-A7B7

y of C:\Users\Admin\Dropbox\nptel_codes\lec41

1  22:12    <DIR>          .
1  22:12    <DIR>          ..
1  22:06        15,370,223 VID_20210216_220105.mp4
       1 File(s)     15,370,223 bytes
       2 Dir(s)   8,314,687,488 bytes free

\Users\Admin\Dropbox\nptel_codes\lec41>
```

later. Until then, the values given to them in code sample will work fine for all

So, let us import cv2.

(Refer Slide Time: 05:30)

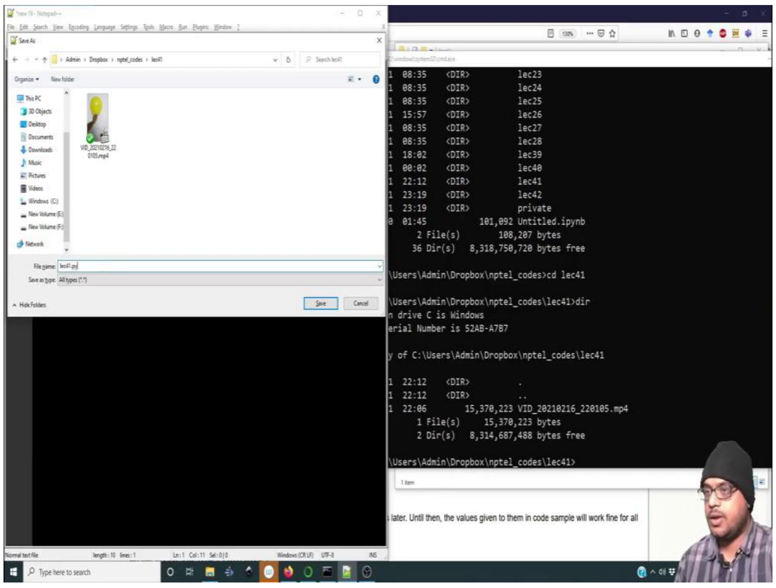


```
1  08:35    <DIR>          lec23
1  08:35    <DIR>          lec24
1  08:35    <DIR>          lec25
1  15:57    <DIR>          lec26
1  08:35    <DIR>          lec27
1  08:35    <DIR>          lec28
1  18:02    <DIR>          lec39
1  00:02    <DIR>          lec40
1  22:12    <DIR>          lec41
1  23:19    <DIR>          lec42
1  23:19    <DIR>          private
0  01:45           101,092 Untitled.ipynb
       2 File(s)        108,207 bytes
      36 Dir(s)   8,318,750,720 bytes free

\Users\Admin\Dropbox\nptel_codes>cd lec41

\Users\Admin\Dropbox\nptel_codes\lec41>dir
n drive C is Windows
erial Number is 52AB-A7B7

y of C:\Users\Admin\Dropbox\nptel_codes\lec41

1  22:12    <DIR>          .
1  22:12    <DIR>          ..
1  22:06        15,370,223 VID_20210216_220105.mp4
       1 File(s)     15,370,223 bytes
       2 Dir(s)   8,314,687,488 bytes free

\Users\Admin\Dropbox\nptel_codes\lec41>
```

later. Until then, the values given to them in code sample will work fine for all

So, cv2 is the name of the library. So, let me save it. I am going to save it in our folder lecture 41 like 41. py.

(Refer Slide Time: 05:39)



```
1  08:35    <DIR>          lec23
1  08:35    <DIR>          lec24
1  08:35    <DIR>          lec25
1  15:57    <DIR>          lec26
1  08:35    <DIR>          lec27
1  08:35    <DIR>          lec28
1  18:02    <DIR>          lec39
1  00:02    <DIR>          lec40
1  22:12    <DIR>          lec41
1  23:19    <DIR>          lec42
1  23:19    <DIR>          private
0  01:45           101,092 Untitled.ipynb
      2 File(s)        108,207 bytes
     36 Dir(s)   8,318,750,720 bytes free

\Users\Admin\Dropbox\nptel_codes>cd lec41

\Users\Admin\Dropbox\nptel_codes\lec41>dir
n drive C is Windows
erial Number is 52AB-A7B7

y of C:\Users\Admin\Dropbox\nptel_codes\lec41

1  22:12    <DIR>          .
1  22:12    <DIR>          ..
1  22:06        15,370,223 VID_20210216_220105.mp4
      1 File(s)     15,370,223 bytes
      2 Dir(s)   8,314,687,488 bytes free

\Users\Admin\Dropbox\nptel_codes\lec41>
```

later. Until then, the values given to them in code sample will work fine for all

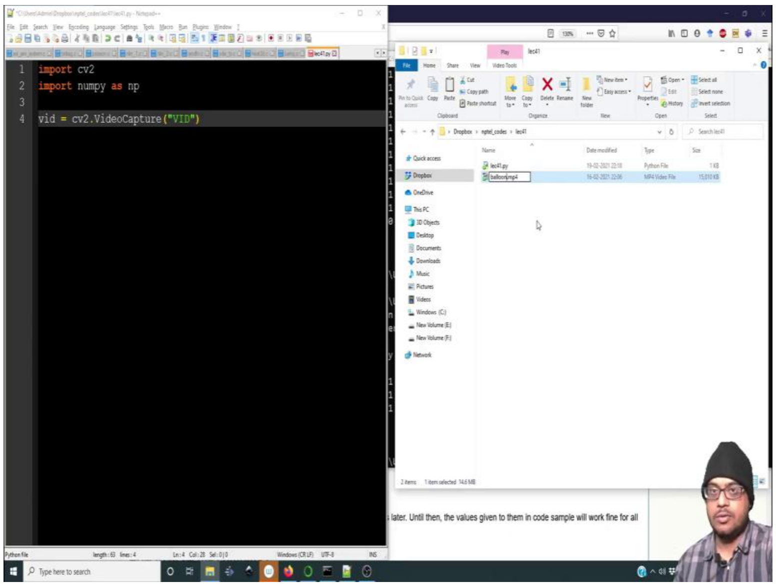
So, we are going to save it as .py, which indicates it is a Python py, alright. So, import cv2. This helps in importing the modules which contain the image processing toolbox. This is for image processing then we are going to import numpy as numpy as np import mat well, we do not need my plot living in this particular case. So, never mind that. So, we are going to just need cv2 and numpy, right.

Then what do we do? So, first things first we have video we have a dot mp 4 file. Now, you could do many things you could use ffmpeg and export each frame of the mp 4 file to a single image. Yeah, that is the perfectly fine way of going about things. In fact, when you do high speed imaging that is how you would obtain your raw files. You would obtain each frame as you go.

You would not obtain a single video file, you will obtain a bunch of files. Those will be static images, but in Python if you have a mp 4 file you can read it like we had read the dot wave file in the case of audio processing. So, we have imported cv2, now we will use the cv2 video capture to load the file. So, we are going to do the vid = cv2.VideoCapture("balloon.mp4").

.

(Refer Slide Time: 07:31)

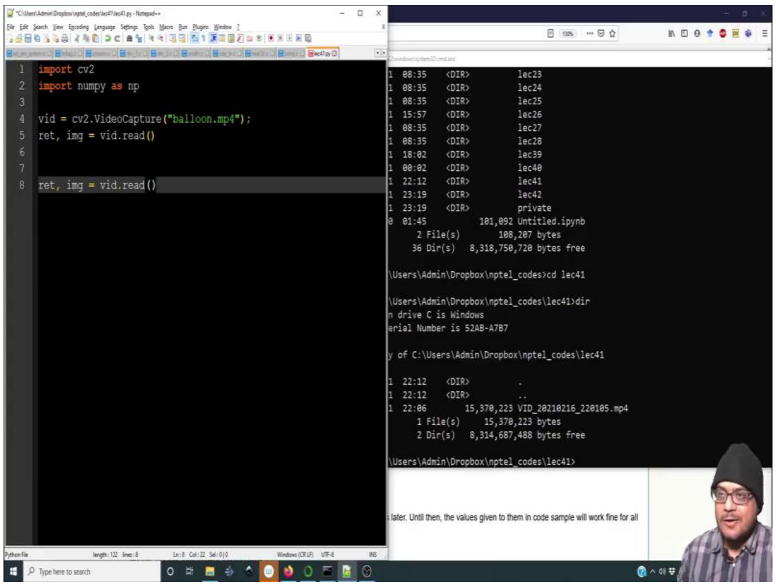


```python
import cv2
import numpy as np

vid = cv2.VideoCapture("VID")
```

balloon.mp4
later. Until then, the values given to them in code sample will work fine for all

So, the name of the file is ok VID, let me do one thing let me call it simply balloon, right.

(Refer Slide Time: 07:40)



```python
import cv2
import numpy as np

vid = cv2.VideoCapture("balloon.mp4");
ret, img = vid.read()


ret, img = vid.read()
```

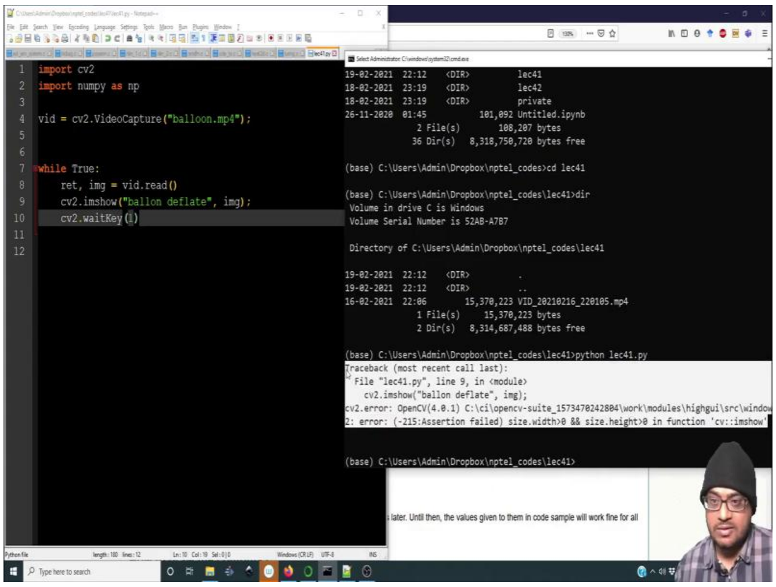
```
1  08:35    <DIR>          lec23
1  08:35    <DIR>          lec24
1  08:35    <DIR>          lec25
1  15:57    <DIR>          lec26
1  08:35    <DIR>          lec27
1  08:35    <DIR>          lec28
1  18:02    <DIR>          lec39
1  00:02    <DIR>          lec40
1  22:12    <DIR>          lec41
1  23:19    <DIR>          lec42
1  23:19    <DIR>          private
0  01:45           101,092 Untitled.ipynb
      2 File(s)        108,207 bytes
     36 Dir(s)   8,318,750,720 bytes free

\Users\Admin\Dropbox\nptel_codes>cd lec41

\Users\Admin\Dropbox\nptel_codes\lec41>dir
n drive C is Windows
erial Number is 52AB-A7B7

y of C:\Users\Admin\Dropbox\nptel_codes\lec41

1  22:12    <DIR>          .
1  22:12    <DIR>          ..
1  22:06        15,370,223 VID_20210216_220105.mp4
      1 File(s)     15,370,223 bytes
      2 Dir(s)   8,314,687,488 bytes free

\Users\Admin\Dropbox\nptel_codes\lec41>
```

later. Until then, the values given to them in code sample will work fine for all

So, I will call this balloon.mp4. Now, once I have this what I need to do is, I need to sort of read each frame of this object vid. So, how do I do that? I say ret, img = vid.read() and this will help me acquire each frame. So, actually once I do this it will only read the first frame.

If I redo the same command over here it will read the second frame. So, it will read a read image and go to the next frame. Once I call this again it will again read it and go to the next image. So, each time I do this well now you can imagine you can do this in a loop to loop over all the frames, but yeah let me do that. Let me first show you how it is true.

(Refer Slide Time: 08:37)



```python
import cv2
import numpy as np

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


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

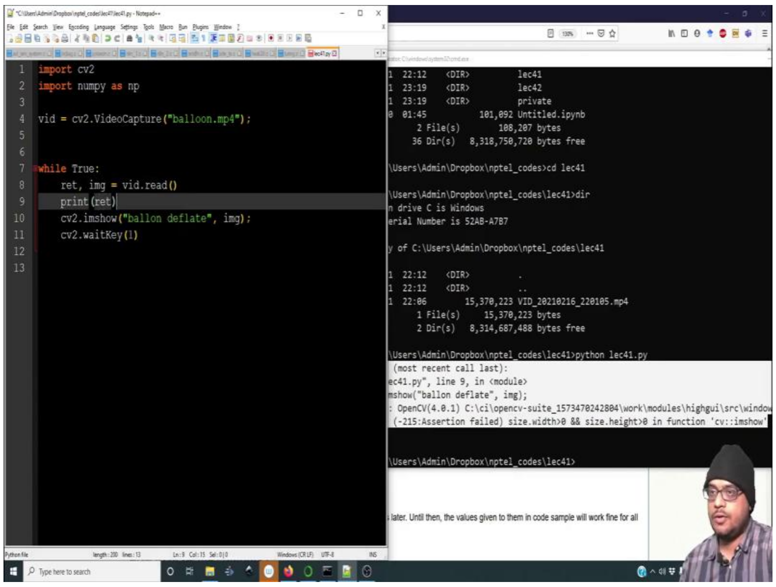
```
19-02-2021  22:12    <DIR>          lec41
18-02-2021  23:19    <DIR>          lec42
18-02-2021  23:19    <DIR>          private
26-11-2020  01:45           101,092 Untitled.ipynb
               2 File(s)        108,207 bytes
              36 Dir(s)   8,318,750,720 bytes free

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

(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>dir
 Volume in drive C is Windows
 Volume Serial Number is 52AB-A7B7

 Directory of C:\Users\Admin\Dropbox\nptel_codes\lec41

19-02-2021  22:12    <DIR>          .
19-02-2021  22:12    <DIR>          ..
16-02-2021  22:06        15,370,223 VID_20210216_220105.mp4
               1 File(s)     15,370,223 bytes
               2 Dir(s)   8,314,687,488 bytes free

(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
Traceback (most recent call last):
  File "lec41.py", line 9, in <module>
    cv2.imshow("ballon deflate", img);
cv2.error: OpenCV(4.0.1) C:\ci\opencv-suite_1573470242804\work\modules\highgui\src\window
2: error: (-215:Assertion failed) size.width>0 && size.height>0 in function 'cv::imshow'


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

later. Until then, the values given to them in code sample will work fine for all

So, while TRUE meaning while it is True and while one I mean this loop will go on forever, then what you will do is you will capture the image and now you will display the image. So, how do you display the image? So, you would do cv2.imshow (i). So, you need to give it a title balloon deflate and then you will type down the name of the frame. So, the name of the frame is img.

So, once you execute this you have to do cv2.waitKey 1 which is to ask the computer to wait for a while 1 millisecond before going to the next. So, let me save this. Let me go over here. Let me do python lec41. py ok. So, there you go. So, now, you will say that ok wait there is a there is a error over here. So, why this error? And, the reason is once it runs out of frame to read the return value will throw an error. So, let us do that.

(Refer Slide Time: 09:54)



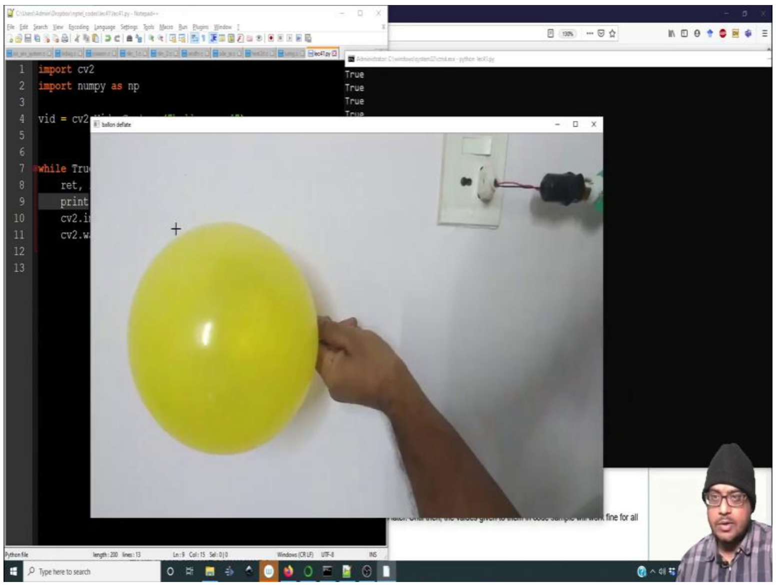
```python
import cv2
import numpy as np

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


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

```
1  22:12    <DIR>          lec41
1  23:19    <DIR>          lec42
1  23:19    <DIR>          private
0  01:45           101,092 Untitled.ipynb
      2 File(s)        108,207 bytes
     36 Dir(s)   8,318,750,720 bytes free

\Users\Admin\Dropbox\nptel_codes>cd lec41

\Users\Admin\Dropbox\nptel_codes\lec41>dir
n drive C is Windows
erial Number is 52AB-A7B7

y of C:\Users\Admin\Dropbox\nptel_codes\lec41

1  22:12    <DIR>          .
1  22:12    <DIR>          ..
1  22:06        15,370,223 VID_20210216_220105.mp4
      1 File(s)     15,370,223 bytes
      2 Dir(s)   8,314,687,488 bytes free

\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
 (most recent call last):
ec41.py", line 9, in <module>
mshow("ballon deflate", img);
: OpenCV(4.0.1) C:\ci\opencv-suite_1573470242804\work\modules\highgui\src\window
 (-215:Assertion failed) size.width>0 && size.height>0 in function 'cv::imshow'


\Users\Admin\Dropbox\nptel_codes\lec41>
```

later. Until then, the values given to them in code sample will work fine for all

Let me print the value of ret.

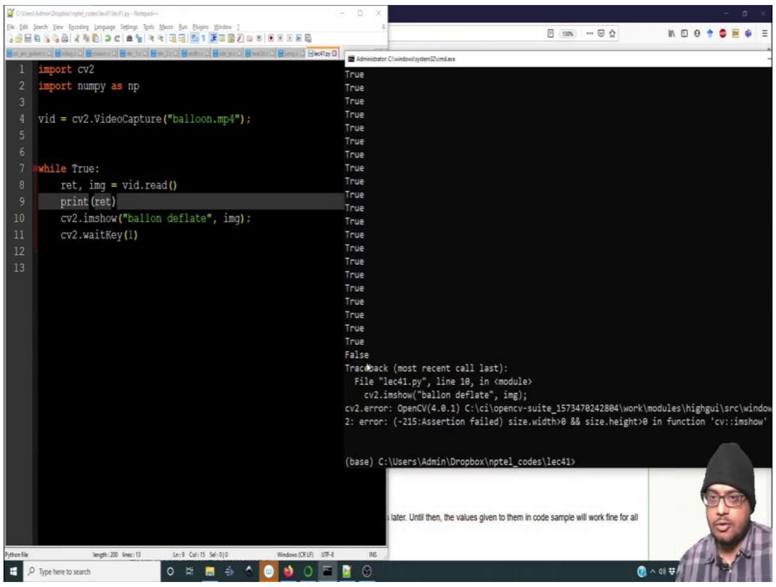
(Refer Slide Time: 09:59)



```
True
True
True
True
```

So, you will realize that it is true and suddenly it goes false.

(Refer Slide Time: 10:01)



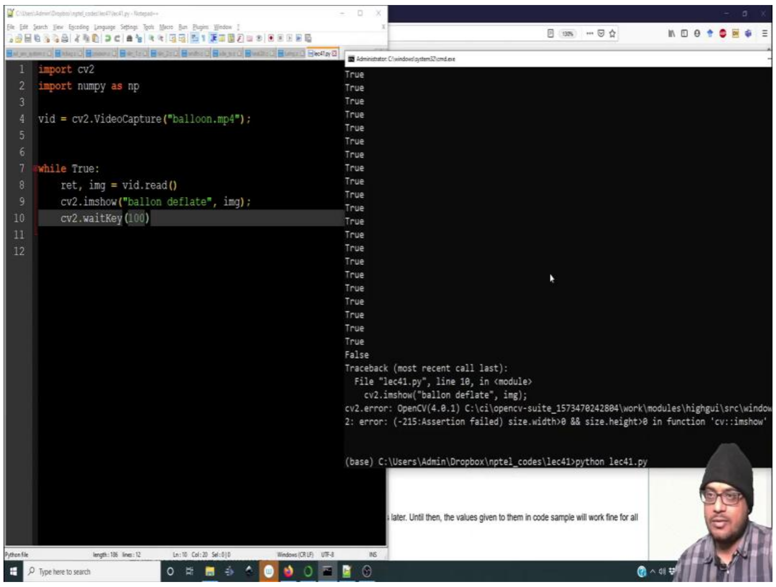
```
import cv2
import numpy as np

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


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

```
True
True
True
True
True
True
True
True
True
True
True
True
True
True
True
True
True
True
True
True
True
False
Traceback (most recent call last):
  File "lec41.py", line 10, in <module>
    cv2.imshow("ballon deflate", img);
cv2.error: OpenCV(4.0.1) C:\ci\opencv-suite_1573470242804\work\modules\highgui\src\window
2: error: (-215:Assertion failed) size.width>0 && size.height>0 in function 'cv::imshow'


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

later. Until then, the values given to them in code sample will work fine for all

So, then the loop exits that is a very hard way of stopping the loop, but in our case it checks out. It does not bother us in any way, it is fine ok. So, this is the basic loop that you will you are going to do. In case you want to increase the delay you can simply make it 100.

(Refer Slide Time: 10:22)



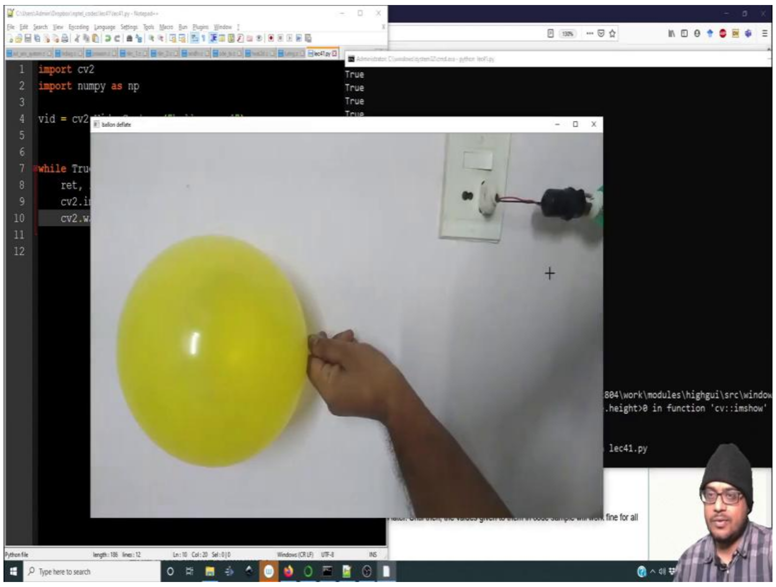
```
import cv2
import numpy as np

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


while True:
    ret, img = vid.read()
    cv2.imshow("ballon deflate", img);
    cv2.waitKey(100)
```

```
True
True
True
True
True
True
True
True
True
True
True
True
True
True
True
True
True
True
True
True
True
False
Traceback (most recent call last):
  File "lec41.py", line 10, in <module>
    cv2.imshow("ballon deflate", img);
cv2.error: OpenCV(4.0.1) C:\ci\opencv-suite_1573470242804\work\modules\highgui\src\window
2: error: (-215:Assertion failed) size.width>0 && size.height>0 in function 'cv::imshow'


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

later. Until then, the values given to them in code sample will work fine for all

So, then you will perceive the code running with a larger delay.

(Refer Slide Time: 10:30)



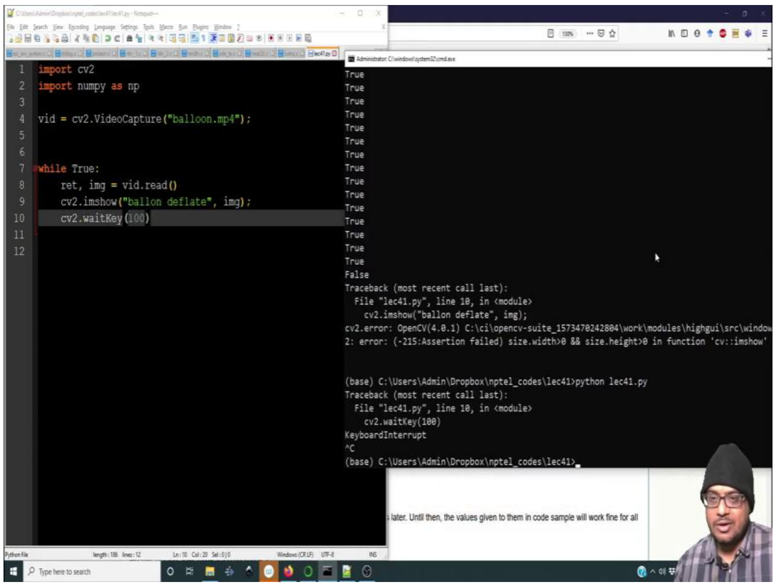
```
import cv2
import numpy as np

vid = cv2


while Tru
    ret, 
    cv2.i
    cv2.w
```

```
True
True
True
```

```
804\work\modules\highgui\src\window
.height>0 in function 'cv::imshow'

lec41.py
```

So, let me run this. So, look, that there is a larger delay in the point.

(Refer Slide Time: 10:34)

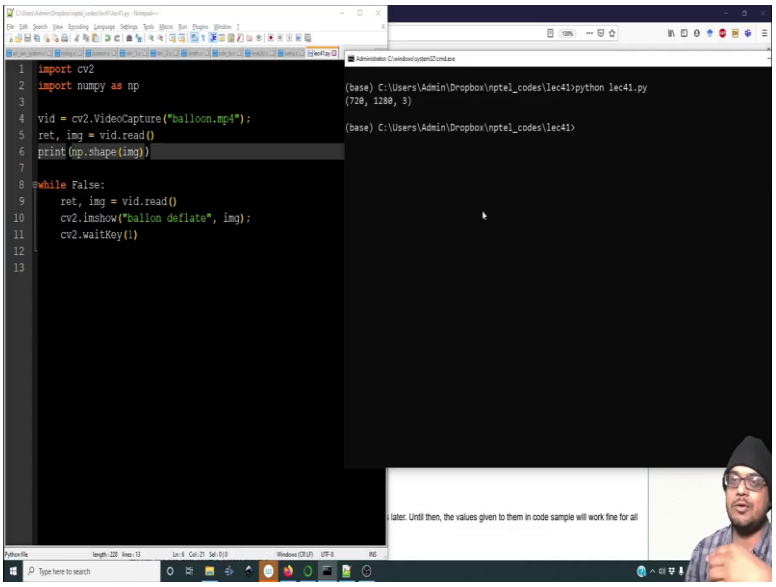


```
import cv2
import numpy as np

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


while True:
    ret, img = vid.read()
    cv2.imshow("ballon deflate", img);
    cv2.waitKey(100)
```

```
True
True
True
True
True
True
True
True
True
True
True
True
True
True
True
False
Traceback (most recent call last):
  File "lec41.py", line 10, in <module>
    cv2.imshow("ballon deflate", img);
cv2.error: OpenCV(4.0.1) C:\ci\opencv-suite_1573470242804\work\modules\highgui\src\window
2: error: (-215:Assertion failed) size.width>0 && size.height>0 in function 'cv::imshow'


(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
Traceback (most recent call last):
  File "lec41.py", line 10, in <module>
    cv2.waitKey(100)
KeyboardInterrupt
^C
(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>_
```

So, you can stop execution of this file by pressing control C. It sends a keyboard interrupt to the terminal and it close away. So, let me clear the screen, alright.

(Refer Slide Time: 10:47)



```python
import cv2
import numpy as np

vid = cv2.VideoCapture("balloon.mp4");
ret, img = vid.read()
print(np.shape(img))

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

```
(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
(720, 1280, 3)

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

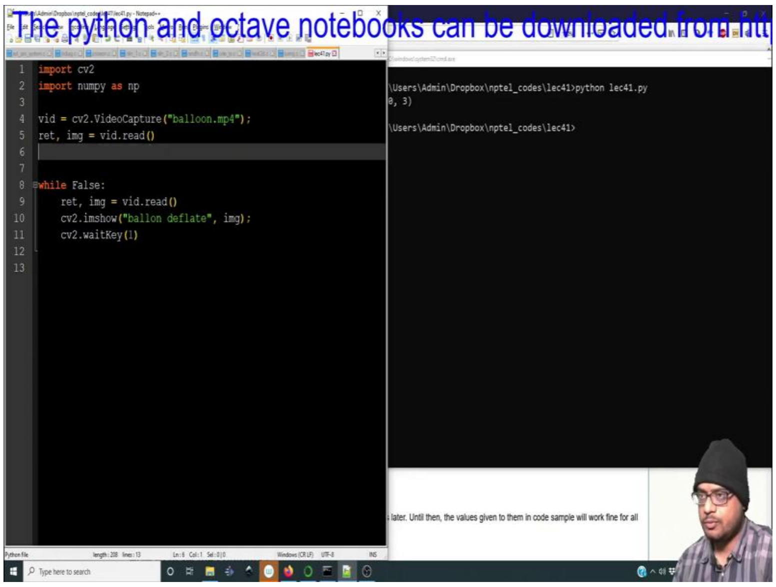
later. Until then, the values given to them in code sample will work fine for all

So, we are going to keep it as 1. We are going to keep 1 millisecond delay. Now, whatever code has to be done is going to be inside this, alright. So, now, before going into the code section of it, we must sort of understand what an image is, right. So, let me make this while False, so that we do not go into this loop.

Now, I am going to do this; I am going to do this. So, I am reading one frame and after reading this frame I am going to print out what actually img is or np.shape (img), so that we know what this object actually is alright. So, I have saved this file let me execute this code. So, the shape of img is 720 × 1280 × 3 meaning the camera that I have on my phone and I was recording it at 720p.

So, the frame size is 720 × 1280 and the 3 actually represent three channels. So, it has three channels, the three channels are b, g and r; blue, green and red. So, each image is composed of three stacks. So, one stack is the blue channel, one stack is the green channel and one stack is the red channel.

(Refer Slide Time: 12:28)


The python and octave notebooks can be downloaded from htt

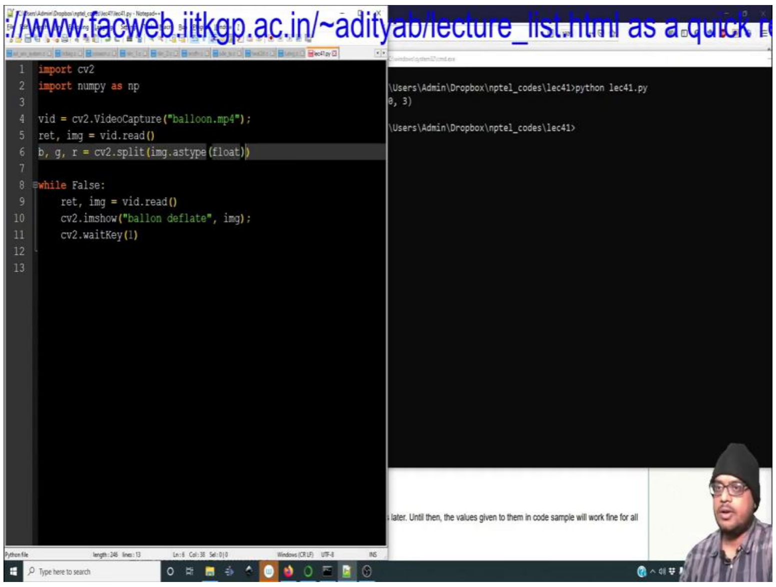
```
import cv2
import numpy as np

vid = cv2.VideoCapture("balloon.mp4");
ret, img = vid.read()


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

```
\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
0, 3)

\Users\Admin\Dropbox\nptel_codes\lec41>
```

later. Until then, the values given to them in code sample will work fine for all

So, if you have a photograph which is dominant in red, then the r channel will show 255. So, 255 means highest value of saturation. If you have something which is blue, the blue channel will show high degree of saturation; if you have intermediate colors you will have some linear combination of this colors, right.

(Refer Slide Time: 12:47)


://www.facweb.iitkgp.ac.in/~adityab/lecture_list.html as a quick t

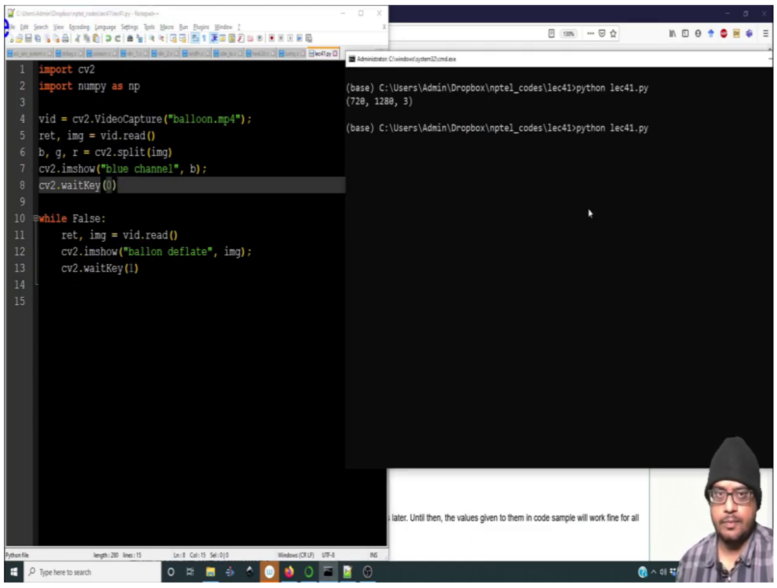
```
import cv2
import numpy as np

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

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

```
\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
0, 3)

\Users\Admin\Dropbox\nptel_codes\lec41>
```

later. Until then, the values given to them in code sample will work fine for all

So, now let me show you what these individual channels look like.

(Refer Slide Time: 12:47)



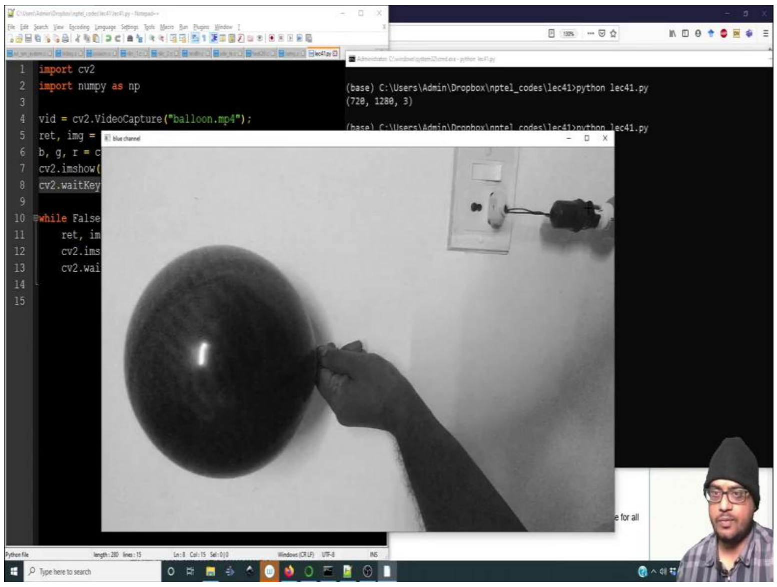
```
import cv2
import numpy as np

vid = cv2.VideoCapture("balloon.mp4");
ret, img = vid.read()
b, g, r = cv2.split(img)
cv2.imshow("blue channel", b);
cv2.waitKey(0)

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

```
(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
(720, 1280, 3)

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

later. Until then, the values given to them in code sample will work fine for all

So, let me do this. So, b, g, r = cv2.split. So, this cv2 dot split command it splits the image into the three channels. So, img and astype float and the reason ok let me not do this first. Let me just do this and let me do cv2.imshow this will be blue channel, this will be b. cv2.waitKey 0, alright. So, let me execute this.

(Refer Slide Time: 13:29)

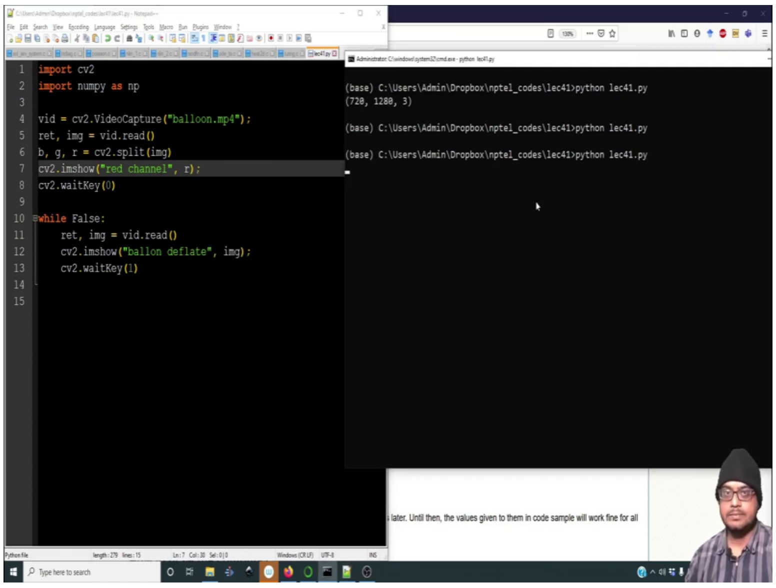


```
import cv2
import numpy as np

vid = cv2.VideoCapture("balloon.mp4");
ret, img =
b, g, r = c
cv2.imshow(
cv2.waitKey

while False
    ret, im
    cv2.ims
    cv2.wai
```

```
(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
(720, 1280, 3)

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

So, this is the first frame and this shows the blue channel. So, in the blue channel, the yellow color appears as black, ok. In the blue channel, the yellow color appears as black ok.

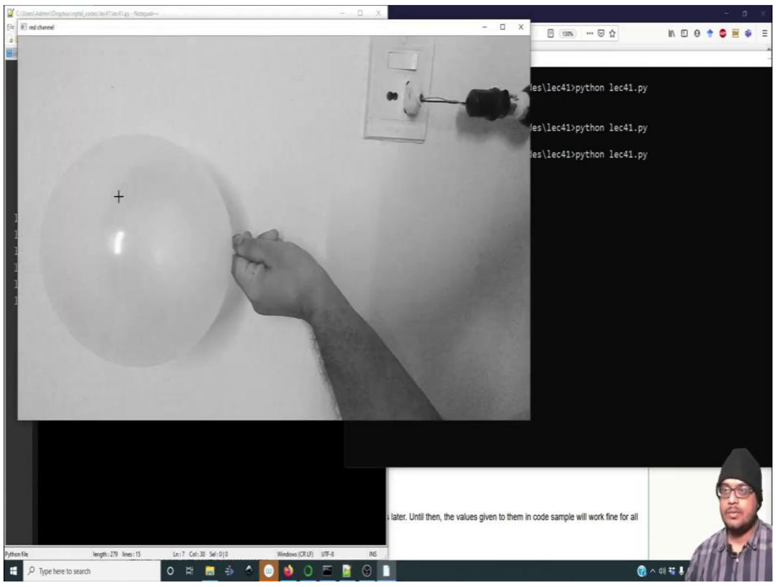
(Refer Slide Time: 13:42)



```
import cv2
import numpy as np

vid = cv2.VideoCapture("balloon.mp4");
ret, img = vid.read()
b, g, r = cv2.split(img)
cv2.imshow("red channel", r);
cv2.waitKey(0)

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

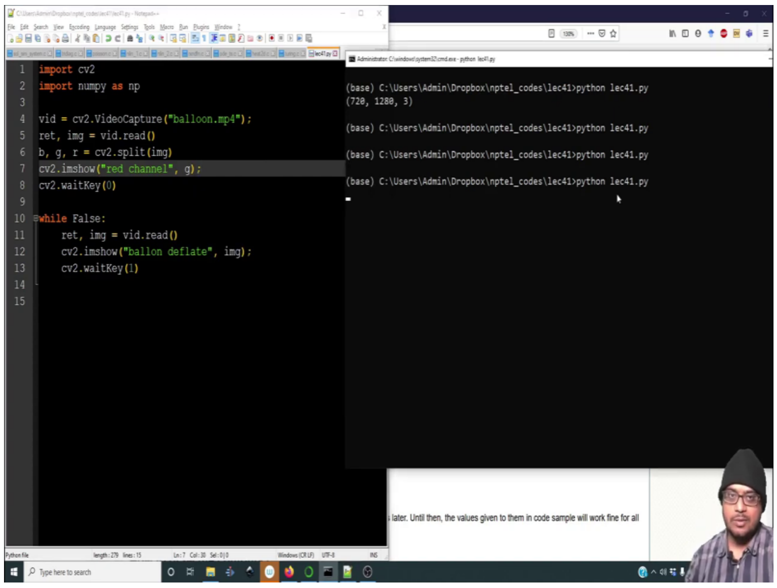
```
(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
(720, 1280, 3)

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

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

(Refer Slide Time: 13:52)



So, let me now print show the red channel. In the red channel the yellow color appears as white, alright.

(Refer Slide Time: 13:59)



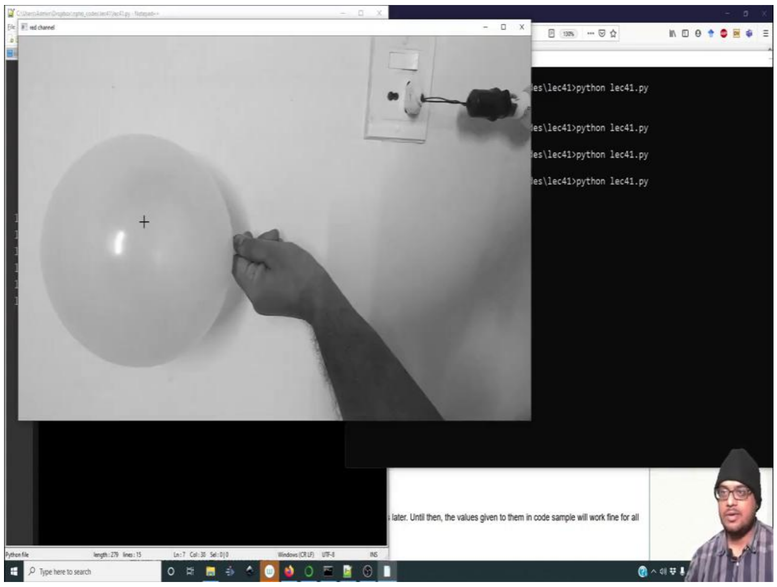
```
import cv2
import numpy as np

vid = cv2.VideoCapture("balloon.mp4");
ret, img = vid.read()
b, g, r = cv2.split(img)
cv2.imshow("red channel", g);
cv2.waitKey(0)

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

```
(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
(720, 1280, 3)

(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
```

later. Until then, the values given to them in code sample will work fine for all

In the green channel, do not bother with the title of the frame. So, I have saved this. I am going to run this.

(Refer Slide Time: 14:08)

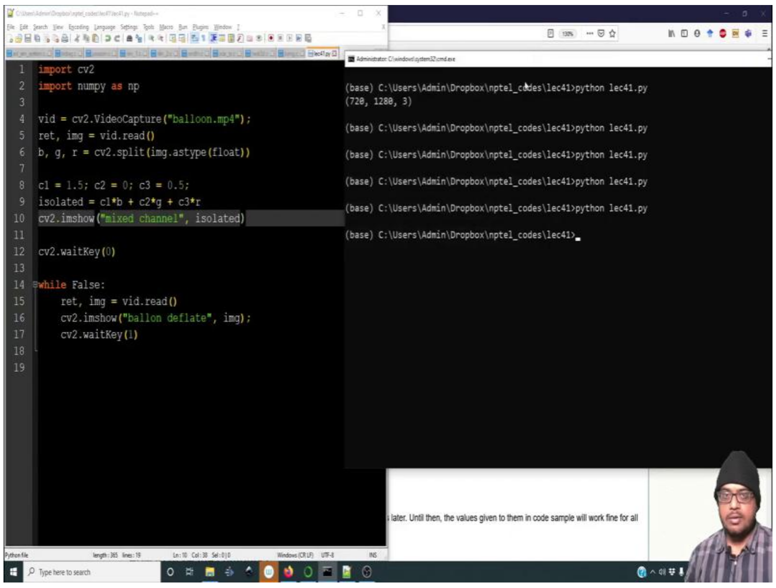


```
es\lec41>python lec41.py

es\lec41>python lec41.py
es\lec41>python lec41.py
es\lec41>python lec41.py
```

later. Until then, the values given to them in code sample will work fine for all

It also appears as white, but notice how the bulb color has changed anyway.

(Refer Slide Time: 14:14)



```python
import cv2
import numpy as np

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

c1 = 1.5; c2 = 0; c3 = 0.5;
isolated = c1*b + c2*g + c3*r
cv2.imshow("mixed channel", isolated)

cv2.waitKey(0)

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

```
(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
(720, 1280, 3)

(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>
```

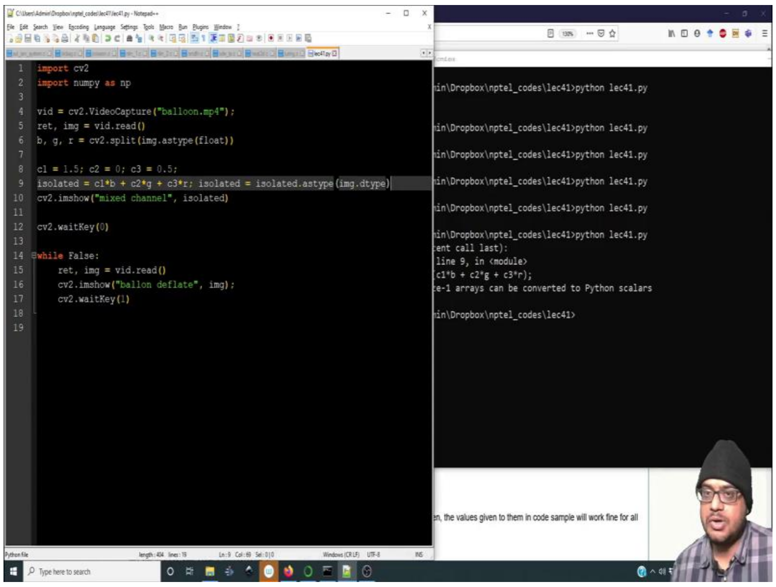
So, this is how we can split, but now what I want to do is I want to do some arithmetic on the image. So, what do I mean by doing some arithmetic on the image I will have some b, g and r. I would like to do some linear combination of b, g and r, so that the balloon is most prominent, the other things are not that prominent.

This means I am going to mix the channels and create a composite image. For that I must first export b, g and r as float this will help in doing the arithmetic later on. Now, once I have split the channels I am going to create a composite image. So, I am going to make isolated = c1*g + c2*b + c3*r, ok.

I am going to do this. Now, what is the purpose of doing this it will be clear. So, let me denote c1=1.5, c2 as 0 or I can write this c1 yeah, c1 = 1.5, c2 = 0, and c3 = 0.5. Now, what it is going to do it is going to take the blue channel amplified by 1.5 times and it is going to add it to the red channel by taking half of its magnitude.

And now, let us see how it looks. So, cv2.im cv2.imshow mixed channel isolated and in fact, before we show it as an image, well let us see I think this is going to throw an error. Yeah, it is not going to show anything and the reason is quite clear because this is. So, isolated is now of a data type float we must convert it from float to a data type. This is the data type image or integer.

(Refer Slide Time: 16:41)

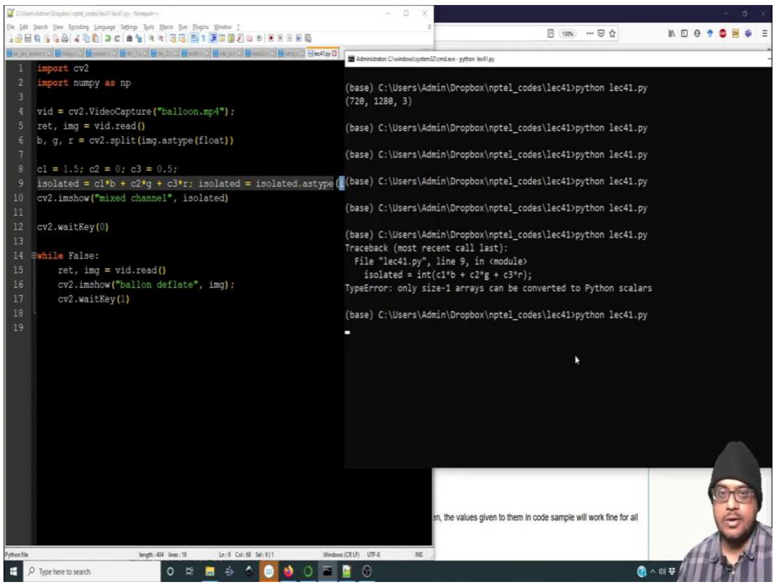


```python
import cv2
import numpy as np

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

c1 = 1.5; c2 = 0; c3 = 0.5;
isolated = c1*b + c2*g + c3*r; isolated = isolated.astype(img.dtype)
cv2.imshow("mixed channel", isolated)

cv2.waitKey(0)

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

```
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 9, in <module>
(c1*b + c2*g + c3*r);
e-1 arrays can be converted to Python scalars

in\Dropbox\nptel_codes\lec41>
```

So, we must convert it. So, we will say isolate or we can simply say int. Let us see whether this works ok. So, it is not going to work. So, we have to do it the hard way that is isolated = isolated.astype(img.dtype). So, it is the data type of an image ok. So, this particular line means whatever the data type of an image is you typecast isolated back to it.

(Refer Slide Time: 17:25)

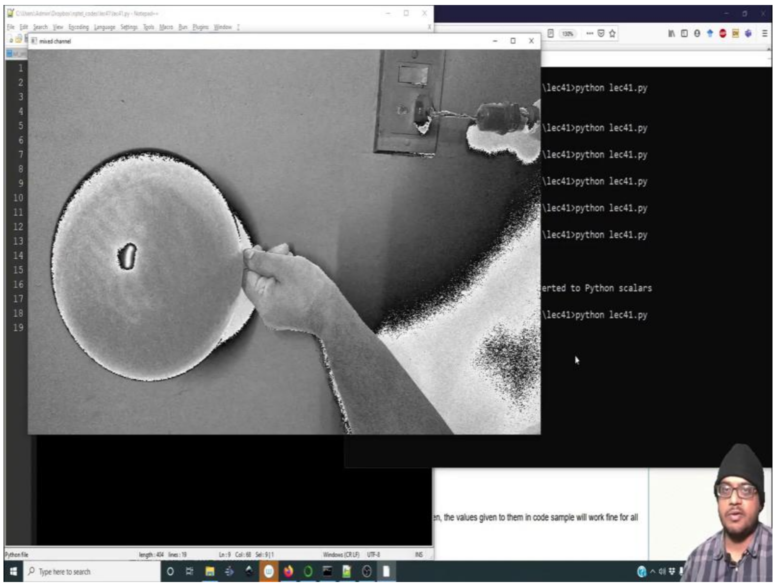


```python
import cv2
import numpy as np

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

c1 = 1.5; c2 = 0; c3 = 0.5;
isolated = c1*b + c2*g + c3*r; isolated = isolated.astype(i
cv2.imshow("mixed channel", isolated)

cv2.waitKey(0)

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

```
(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
(720, 1280, 3)

(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 9, in <module>
    isolated = int(c1*b + c2*g + c3*r);
TypeError: only size-1 arrays can be converted to Python scalars

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

Now, let us see whether it works.

(Refer Slide Time: 17:27)

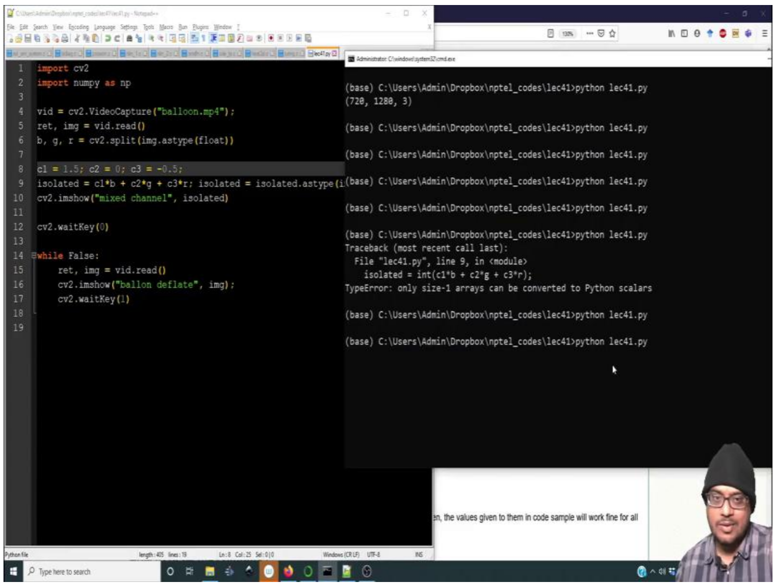


```
\lec41>python lec41.py

\lec41>python lec41.py

\lec41>python lec41.py

\lec41>python lec41.py

\lec41>python lec41.py

\lec41>python lec41.py


erted to Python scalars

\lec41>python lec41.py
```

n, the values given to them in code sample will work fine for all

Perfect, it works. So, now, what have I done? I have removed the green channel completely and if you remove the green channel completely you are going to see a bunch of things loose fading out. So, there is no channel over here because this was predominantly green ok. So, you sort of see how mixing the channels kind of work.

(Refer Slide Time: 17:47)



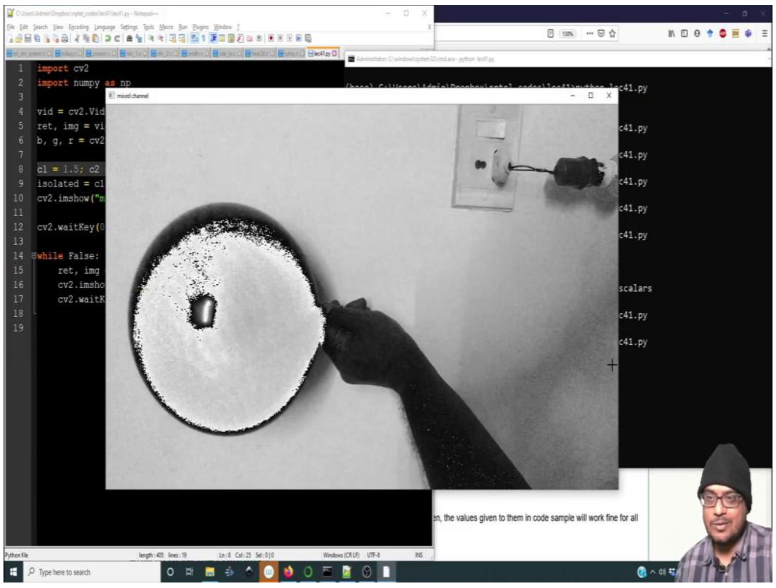
```python
import cv2
import numpy as np

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

c1 = 1.5; c2 = 0; c3 = -0.5;
isolated = c1*b + c2*g + c3*r; isolated = isolated.astype(i
cv2.imshow("mixed channel", isolated)

cv2.waitKey(0)

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

```
(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
(720, 1280, 3)

(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 9, in <module>
    isolated = int(c1*b + c2*g + c3*r);
TypeError: only size-1 arrays can be converted to Python scalars

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

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

n, the values given to them in code sample will work fine for all

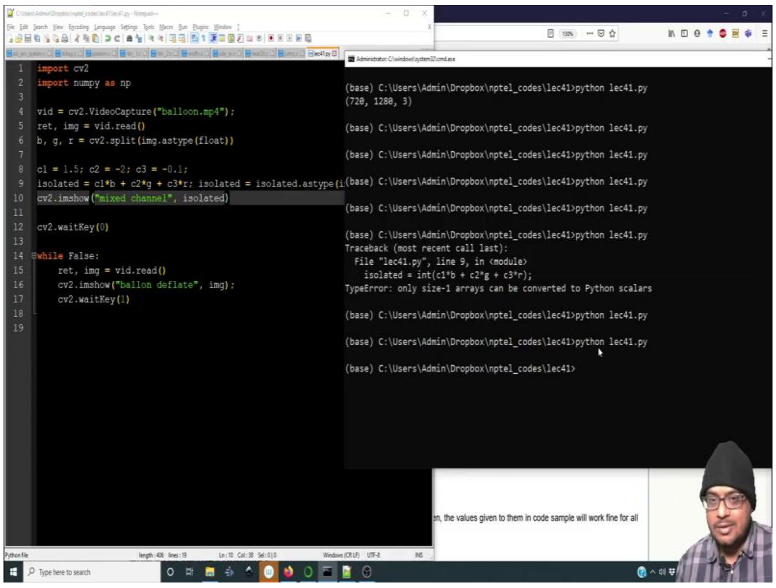
So, now, let me do this let me do c3 as minus.

(Refer Slide Time: 17:53)



Let me see how it looks. You see that we are slowly getting towards the edge ok. Forget about all this. The hand has become darker, but this is where the edge detection will going to work it is going to work really nicely.

(Refer Slide Time: 18:08)

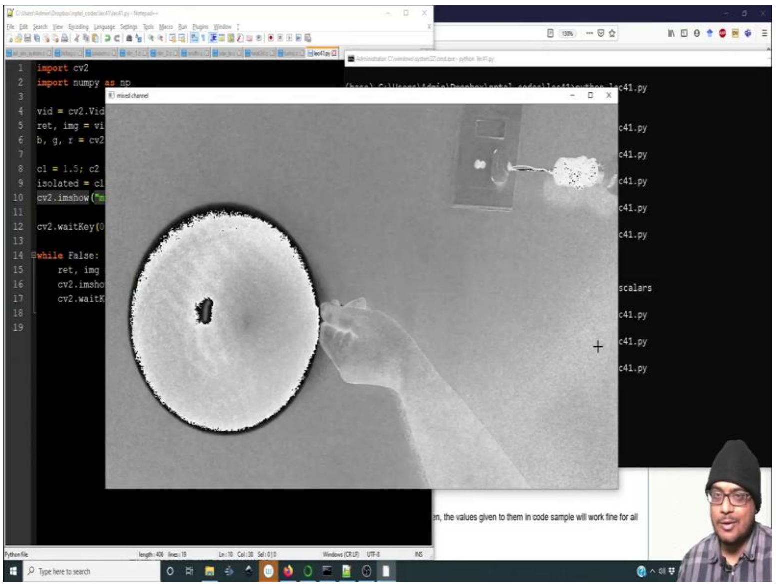


```
import cv2
import numpy as np

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

c1 = 1.5; c2 = -2; c3 = -0.1;
isolated = c1*b + c2*g + c3*r; isolated = isolated.astype(i
cv2.imshow("mixed channel", isolated)

cv2.waitKey(0)

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

```
(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
(720, 1280, 3)

(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 9, in <module>
    isolated = int(c1*b + c2*g + c3*r);
TypeError: only size-1 arrays can be converted to Python scalars

(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>
```

In fact, I have done the trial and error for this and if you do this, it is going to work out the best and this combination of channels was suggested to me by one of my students Shriyansh Darshan. He is a very talented person in all these kind of things and he helped me figure out the different biases of the channels ok.

(Refer Slide Time: 18:37)

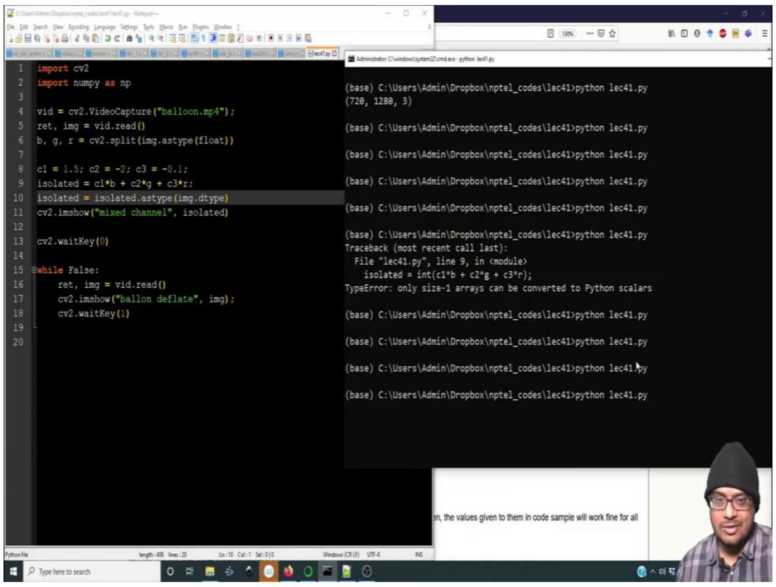


```
import cv2
import numpy as np

vid = cv2.Vid
ret, img = vi
b, g, r = cv2

c1 = 1.5; c2
isolated = c1
cv2.imshow("m

cv2.waitKey(0

while False:
    ret, img
    cv2.imsho
    cv2.waitK
```

```
c41.py
c41.py
c41.py
c41.py
c41.py
c41.py
scalars
c41.py
c41.py
c41.py
```

So, now let me run this and show you how it will look. So, once you do this, it looks as if that edge is very distinct, alright.

(Refer Slide Time: 18:49)

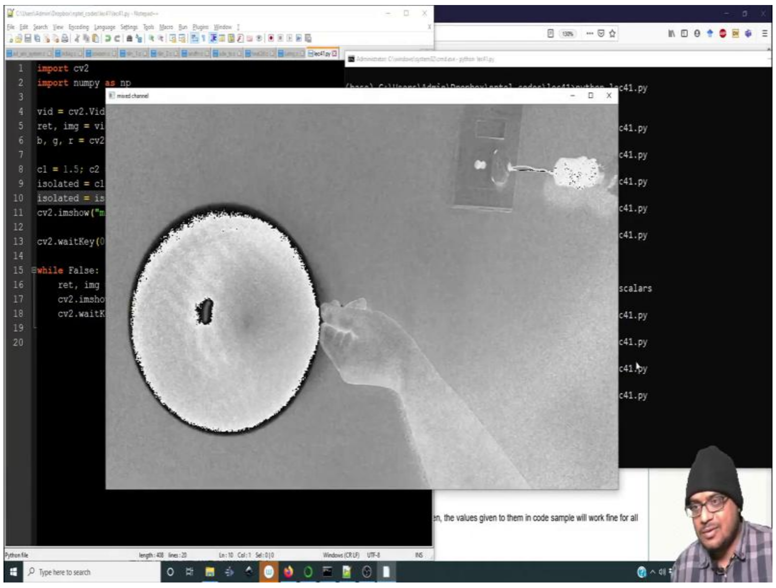


```
import cv2
import numpy as np

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

c1 = 1.5; c2 = -2; c3 = -0.1;
isolated = c1*b + c2*g + c3*r;
isolated = isolated.astype(img.dtype)
cv2.imshow("mixed channel", isolated)

cv2.waitKey(0)

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

```
(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
(720, 1280, 3)

(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 9, in <module>
    isolated = int(c1*b + c2*g + c3*r);
TypeError: only size-1 arrays can be converted to Python scalars

(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
```

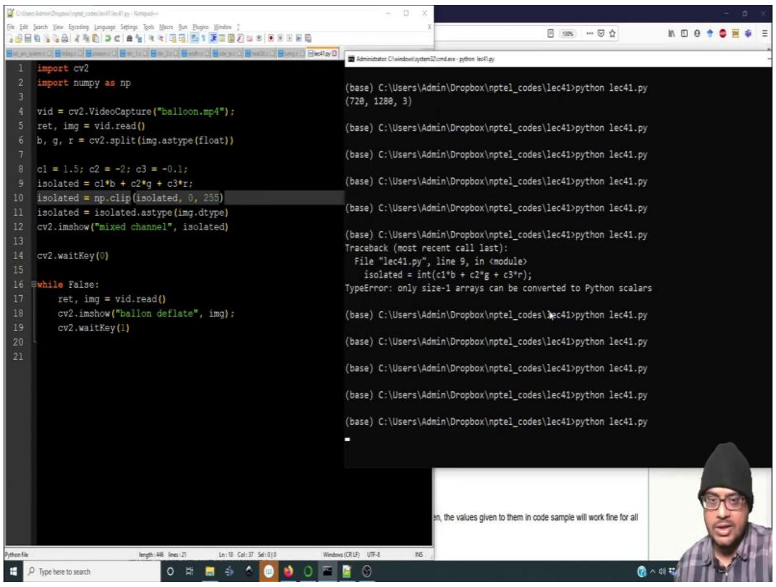
So, things are going in the right direction ok, alright.

(Refer Slide Time: 19:05)


mixed channel
en, the values given to them in code sample will work fine for all

So, actually before converting into sorry, from the image what it seemed like there was some saturation in the image like these areas are saturated.

(Refer Slide Time: 19:08)



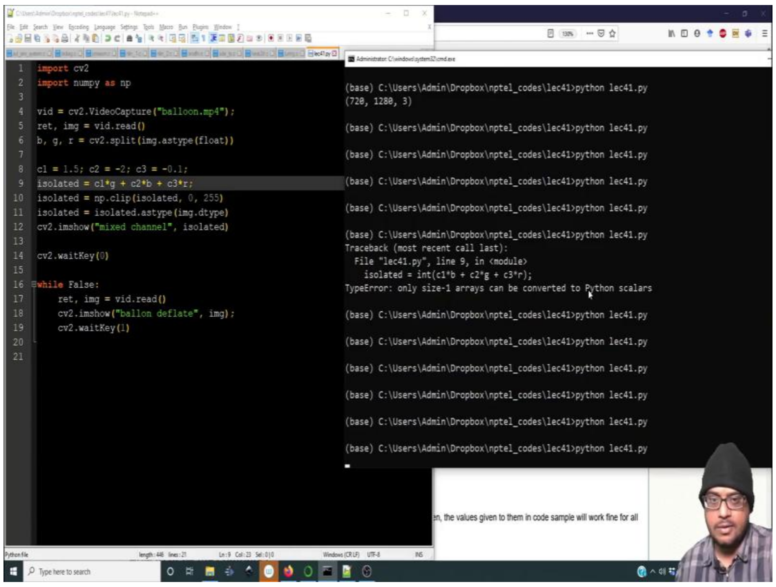
```
import cv2
import numpy as np

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

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

cv2.waitKey(0)

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

```
(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
(720, 1280, 3)

(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 9, in <module>
    isolated = int(c1*b + c2*g + c3*r);
TypeError: only size-1 arrays can be converted to Python scalars

(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
```

en, the values given to them in code sample will work fine for all

So, we can always clip the image to something between 0 and 255. So, typically the images the integer values are going from 0 to 255 meaning there is 256 levels of information ok. So, we are going to do that. So, isolated = np.clip(isolated, 0, 255) and we are going to clip isolated actually we are going to clip it before type casting it to the data type. So, we are going to clip it between 0 and 255. So, let us see what this gives us and this should give this gives nothing. I have put in the coefficients incorrectly. So g, b ok.

(Refer Slide Time: 20:20)



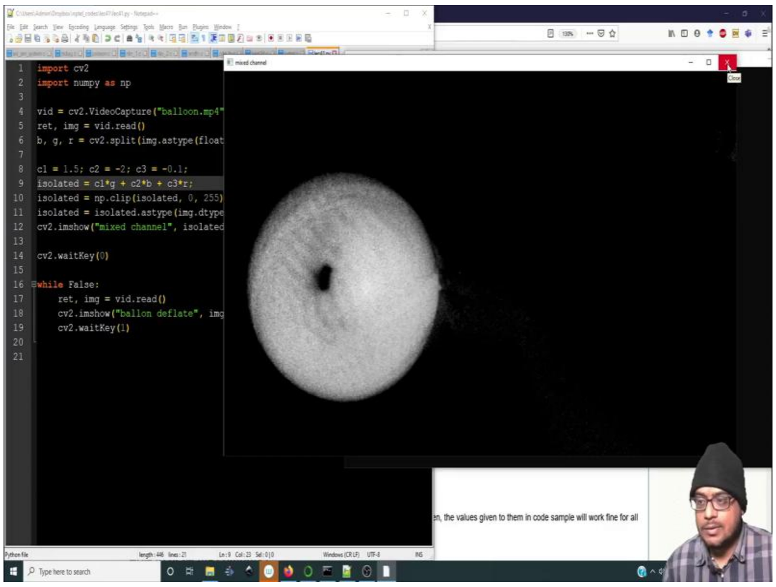
```python
import cv2
import numpy as np

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

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", isolated)

cv2.waitKey(0)

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

```
(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
(720, 1280, 3)

(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 9, in <module>
    isolated = int(c1*b + c2*g + c3*r);
TypeError: only size-1 arrays can be converted to Python scalars

(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
```

en, the values given to them in code sample will work fine for all

So, let us see.

(Refer Slide Time: 20:24)



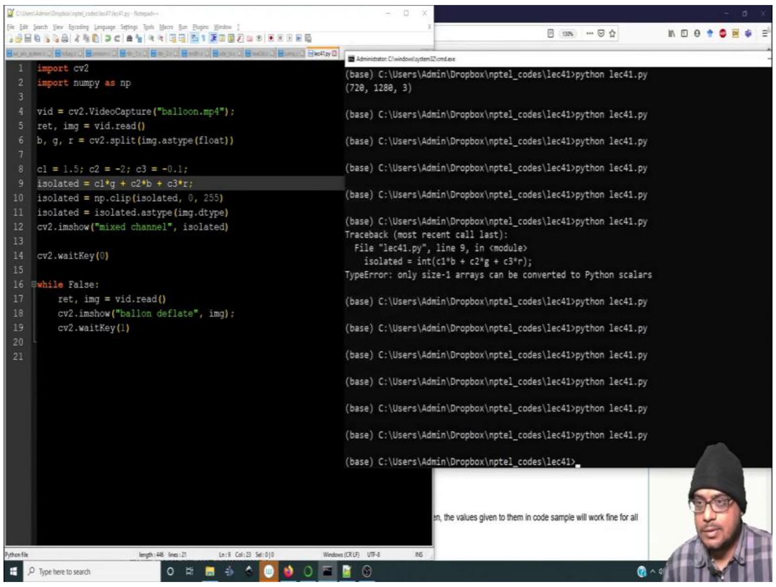
```python
import cv2
import numpy as np

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

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", isolated

cv2.waitKey(0)

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

en, the values given to them in code sample will work fine for all

Yeah, ok. So, once you mix those channels up this is what you obtain. You obtain an image where everything is sort of darkened out accepting for the balloon and the small light shadow of the fluorescent tube onto the image. Well, that is not going to be of a lot of concern to us because in the end it is going to work out quite well.

So, as you can see in that such a channel mixing gives you a nice sort of binary feel to it. This is not yet binary because the data type varies from 0 to 255 we have clipped it like this, but the important thing is once you threshold this image it is going to look much better.

(Refer Slide Time: 21:22)



```python
import cv2
import numpy as np

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

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", isolated)

cv2.waitKey(0)

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

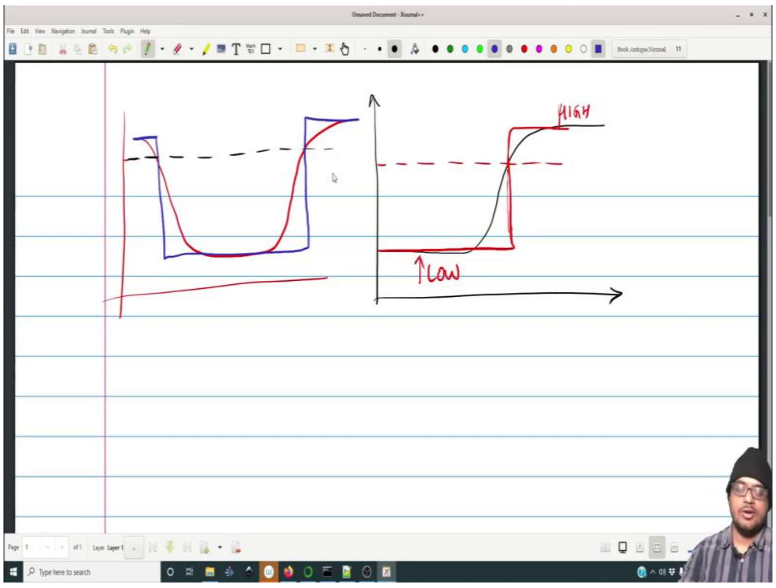
```
(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
(720, 1280, 3)

(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 9, in <module>
    isolated = int(c1*b + c2*g + c3*r);
TypeError: only size-1 arrays can be converted to Python scalars

(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

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

en, the values given to them in code sample will work fine for all

So, what I am going to do is I am going to remove this for now or I am going to keep it over here because we will require it later on. So, I have obtained an image which is which seems like it can help even a blind person figure out what the two things are. You can see white and you can see black.

So, that the boundary is the balloon. Now, what you need to do is you need to do two things first is you need to threshold an image. So, what is thresholding an image? So, let me (Refer Time: 22:05) show you what thresholding means.

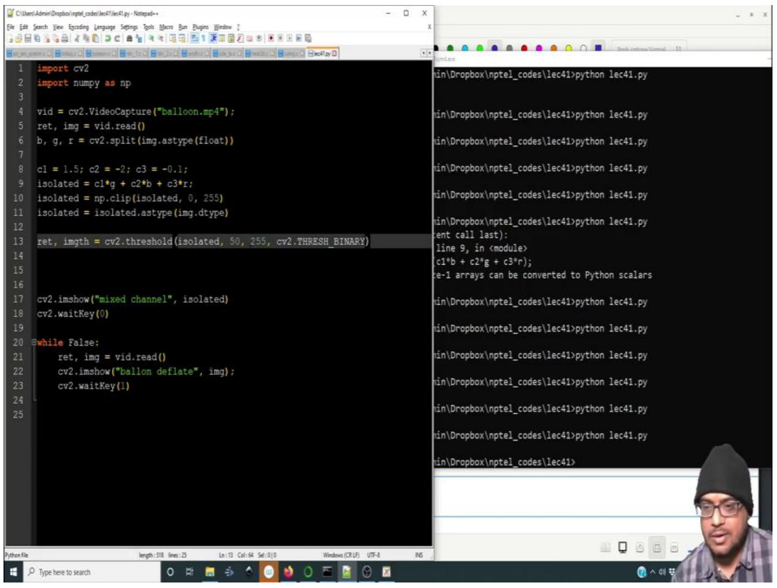
(Refer Slide Time: 22:08)



So, imagine you have and a signal like this ok and you say that this is my threshold value. So, what it will do is all the values below this it will put to 0, all the values above this threshold it will put to 1. So, this is effectively binarizing your image. It is going to convert whatever image you have to either low or high ok.

So, if you have a smooth variation like this, it will convert it into something which is sharp ok. So, imagine you have an image or earth signal that goes like this and if this is the threshold value ok, then what will happen? The signal will be looking like this ok that is what it is. You are going to convert it into binary, alright. So, let us go over here.

(Refer Slide Time: 23:25)



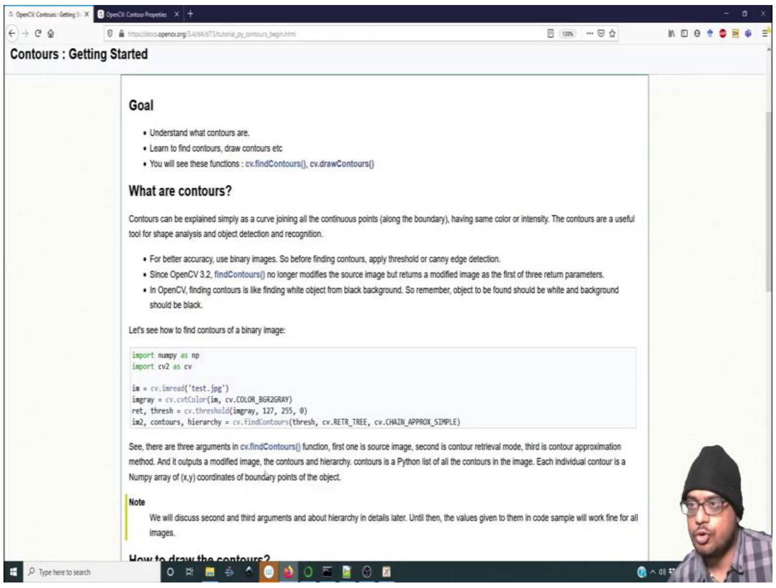
```python
import cv2
import numpy as np

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

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)

ret, imgth = cv2.threshold(isolated, 50, 255, cv2.THRESH_BINARY)



cv2.imshow("mixed channel", isolated)
cv2.waitKey(0)

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

```
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 9, in <module>
(c1*b + c2*g + c3*r);
e-1 arrays can be converted to Python scalars

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>
```

Now, let us do that thresholding. So, we are going to have ret, imgth = cv2.threshold. Now, you are going to pass isolated to it; the image that on which you want to apply the thresholding and you are going to give the lower value of the threshold and the where you want to threshold it to.

So, if I want to threshold it all the way to the top I can do that. I can say any value which is larger than 50 is going to be thresholded to 255 and the type of thresholding. So, we can have various kinds of thresholding and the type of thresholding is going to be THRESH BINARY meaning it is one eventually converted to 0s and 1s.

(Refer Slide Time: 24:16)



Contours : Getting Started

Goal

- Understand what contours are.
- Learn to find contours, draw contours etc
- You will see these functions : cv.findContours(), cv.drawContours()

What are contours?

Contours can be explained simply as a curve joining all the continuous points (along the boundary), having same color or intensity. The contours are a useful tool for shape analysis and object detection and recognition.

- For better accuracy, use binary images. So before finding contours, apply threshold or canny edge detection.
- Since OpenCV 3.2, findContours() no longer modifies the source image but returns a modified image as the first of three return parameters.
- In OpenCV, finding contours is like finding white object from black background. So remember, object to be found should be white and background should be black.

Let's see how to find contours of a binary image:

```
import numpy as np
import cv2 as cv

im = cv.imread('test.jpg')
imgray = cv.cvtColor(im, cv.COLOR_BGR2GRAY)
ret, thresh = cv.threshold(imgray, 127, 255, 0)
im2, contours, hierarchy = cv.findContours(thresh, cv.RETR_TREE, cv.CHAIN_APPROX_SIMPLE)
```

See, there are three arguments in cv.findContours() function, first one is source image, second is contour retrieval mode, third is contour approximation method. And it outputs a modified image, the contours and hierarchy. contours is a Python list of all the contours in the image. Each individual contour is a Numpy array of (x,y) coordinates of boundary points of the object.

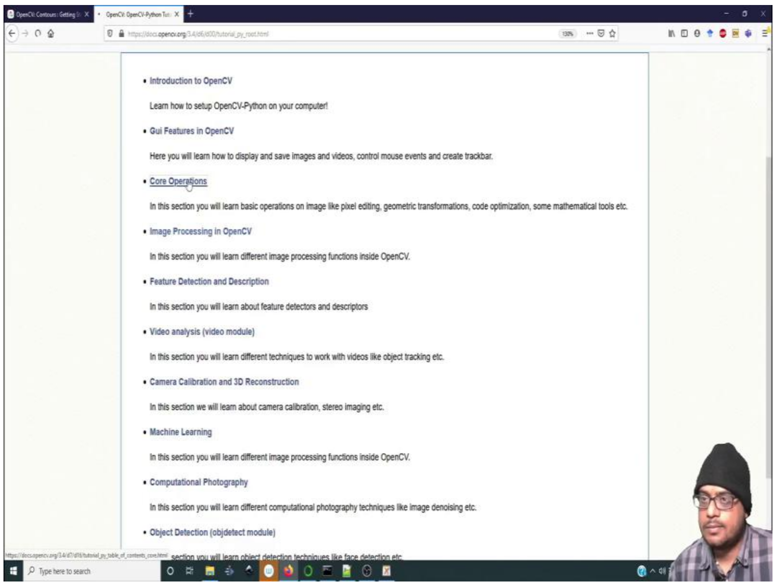
Note

We will discuss second and third arguments and about hierarchy in details later. Until then, the values given to them in code sample will work fine for all images.

How to draw the contours?

So, let me show you how they look.

(Refer Slide Time: 24:21)



- Introduction to OpenCV

  Learn how to setup OpenCV-Python on your computer!

- Gui Features in OpenCV

  Here you will learn how to display and save images and videos, control mouse events and create trackbar.

- Core Operations

  In this section you will learn basic operations on image like pixel editing, geometric transformations, code optimization, some mathematical tools etc.

- Image Processing in OpenCV

  In this section you will learn different image processing functions inside OpenCV.

- Feature Detection and Description

  In this section you will learn about feature detectors and descriptors

- Video analysis (video module)

  In this section you will learn different techniques to work with videos like object tracking etc.

- Camera Calibration and 3D Reconstruction

  In this section we will learn about camera calibration, stereo imaging etc.

- Machine Learning

  In this section you will learn different image processing functions inside OpenCV.

- Computational Photography

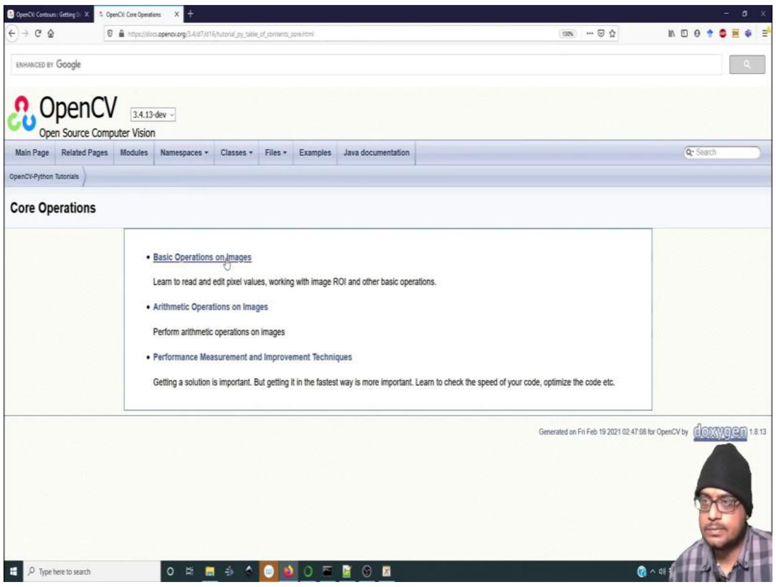
  In this section you will learn different computational photography techniques like image denoising etc.

- Object Detection (objdetect module)

  section you will learn object detection techniques like face detection etc.

So, let me just show you the documentation of this so that it is clear what thresholding means.

(Refer Slide Time: 24:32)



OpenCV 3.4.13-dev

Open Source Computer Vision

Main Page | Related Pages | Modules | Namespaces | Classes | Files | Examples | Java documentation

OpenCV-Python Tutorials

Core Operations

- Basic Operations on Images

  Learn to read and edit pixel values, working with image ROI and other basic operations.

- Arithmetic Operations on Images

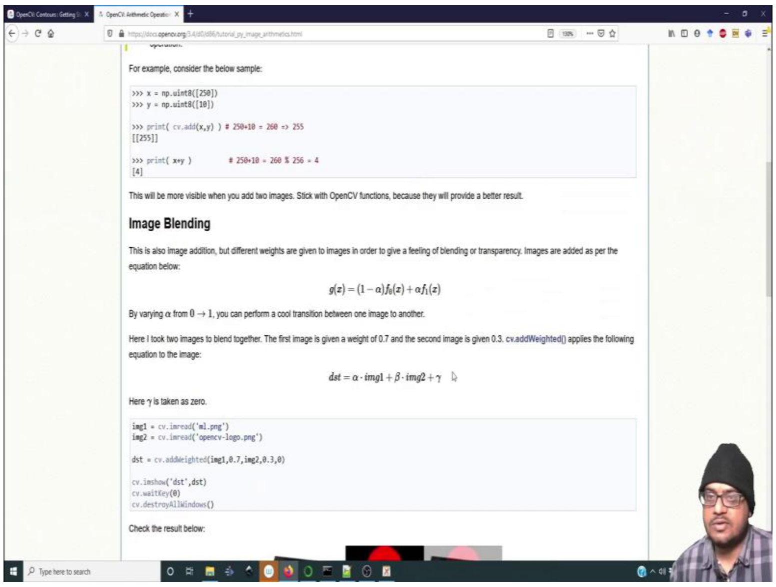
  Perform arithmetic operations on images

- Performance Measurement and Improvement Techniques

  Getting a solution is important. But getting it in the fastest way is more important. Learn to check the speed of your code, optimize the code etc.

Generated on Fri Feb 19 2021 02:47:08 for OpenCV by doxygen 1.8.13

(Refer Slide Time: 24:53)



For example, consider the below sample:

```
>>> x = np.uint8([250])
>>> y = np.uint8([10])

>>> print( cv.add(x,y) ) # 250+10 = 260 => 255
[[255]]

>>> print( x+y )          # 250+10 = 260 % 256 = 4
[4]
```

This will be more visible when you add two images. Stick with OpenCV functions, because they will provide a better result.

Image Blending

This is also image addition, but different weights are given to images in order to give a feeling of blending or transparency. Images are added as per the equation below:

$$g(x) = (1-\alpha)f_0(x) + \alpha f_1(x)$$

By varying α from $0 \rightarrow 1$, you can perform a cool transition between one image to another.

Here I took two images to blend together. The first image is given a weight of 0.7 and the second image is given 0.3. cv.addWeighted() applies the following equation to the image:

$$dst = \alpha \cdot img1 + \beta \cdot img2 + \gamma$$

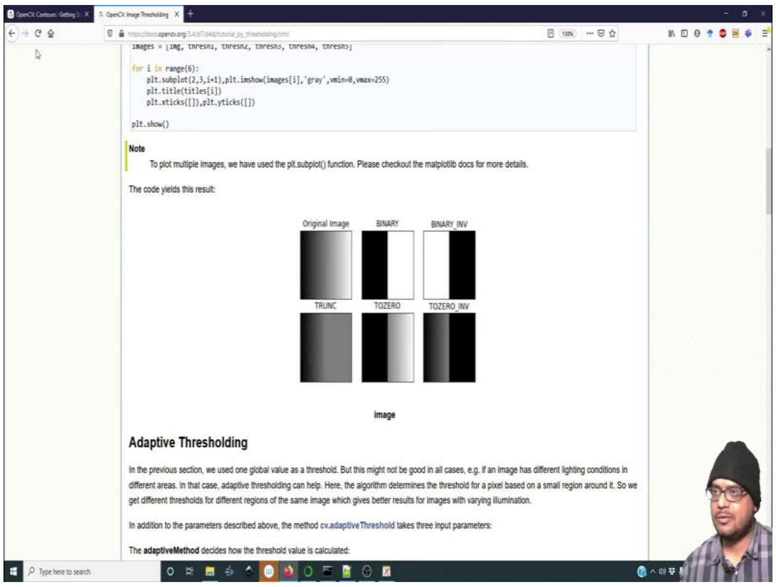
Here γ is taken as zero.

```
img1 = cv.imread('ml.png')
img2 = cv.imread('opencv-logo.png')

dst = cv.addWeighted(img1,0.7,img2,0.3,0)

cv.imshow('dst',dst)
cv.waitKey(0)
cv.destroyAllWindows()
```

Check the result below:

So, this is the channel mixing that we spoke about it is called as image blending. So, this is a bitwise operation we do not need that ok.

(Refer Slide Time: 25:18)



```
images = [img, thresh1, thresh2, thresh3, thresh4, thresh5]

for i in range(6):
    plt.subplot(2,3,i+1),plt.imshow(images[i],'gray',vmin=0,vmax=255)
    plt.title(titles[i])
    plt.xticks([]),plt.yticks([])

plt.show()
```

Note

To plot multiple images, we have used the plt.subplot() function. Please checkout the matplotlib docs for more details.

The code yields this result:

image

Adaptive Thresholding

In the previous section, we used one global value as a threshold. But this might not be good in all cases, e.g. if an image has different lighting conditions in different areas. In that case, adaptive thresholding can help. Here, the algorithm determines the threshold for a pixel based on a small region around it. So we get different thresholds for different regions of the same image which gives better results for images with varying illumination.

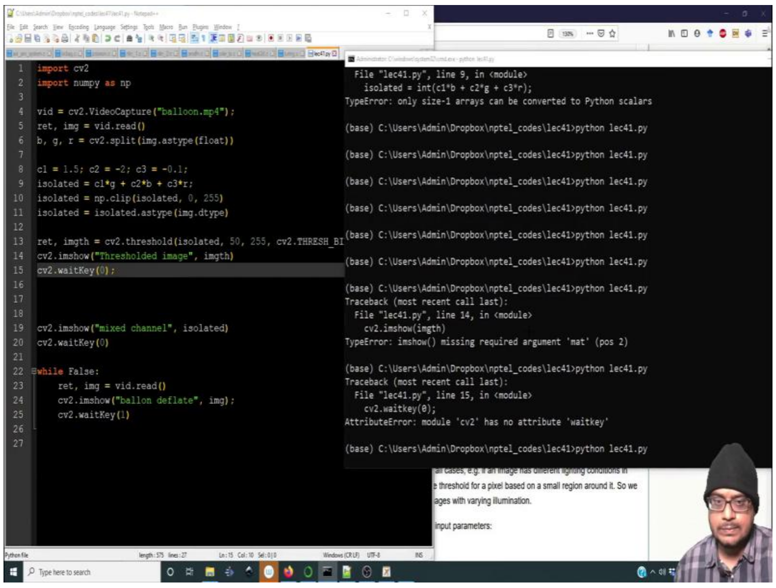
In addition to the parameters described above, the method cv.adaptiveThreshold takes three input parameters:

The **adaptiveMethod** decides how the threshold value is calculated:

So, this is what we are doing. So, if this is the original image. If you do it as a binary thresholding, if this is the threshold gray scale then all the gray scales below that will be put to white and all the gray scales above that will be put to black. This is what is going on and there is also a mode called binary inverted. So, it is going to take that threshold and invert it.

So, it is quite, it is quite useful in case you want to make masks or whatever. So, in order to make masks the useful threshold functions are TRUNC, TOZERO and TOZERO inverse, meaning if you put TOZERO it will put everything below a certain threshold to black, but above it is going to leave the image unchanged ok. These are also quite important in case you want to make masks. Well, so, here we going to convert it to binary.

(Refer Slide Time: 26:13)

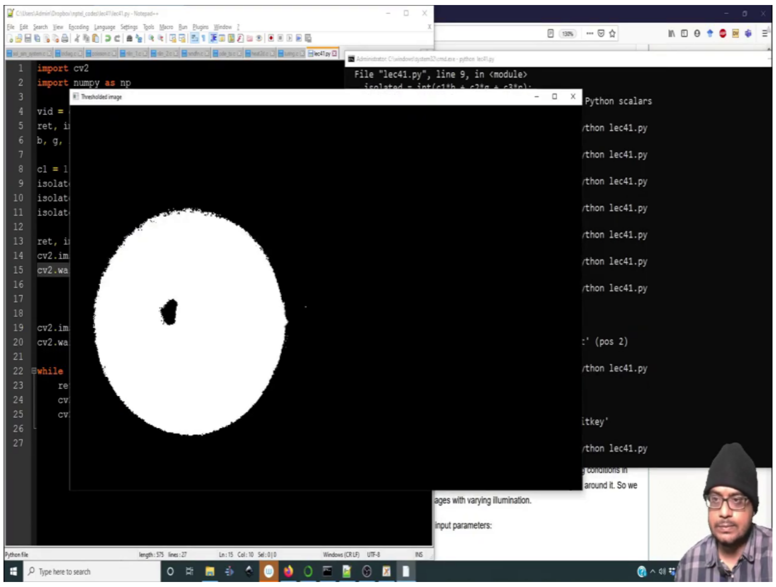


```python
import cv2
import numpy as np

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

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)

ret, imgth = cv2.threshold(isolated, 50, 255, cv2.THRESH_BI
cv2.imshow("Thresholded image", imgth)
cv2.waitKey(0);



cv2.imshow("mixed channel", isolated)
cv2.waitKey(0)

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

```
  File "lec41.py", line 9, in <module>
    isolated = int(c1*b + c2*g + c3*r);
TypeError: only size-1 arrays can be converted to Python scalars

(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

(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
Traceback (most recent call last):
  File "lec41.py", line 14, in <module>
    cv2.imshow(imgth)
TypeError: imshow() missing required argument 'mat' (pos 2)

(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
Traceback (most recent call last):
  File "lec41.py", line 15, in <module>
    cv2.waitkey(0);
AttributeError: module 'cv2' has no attribute 'waitkey'

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

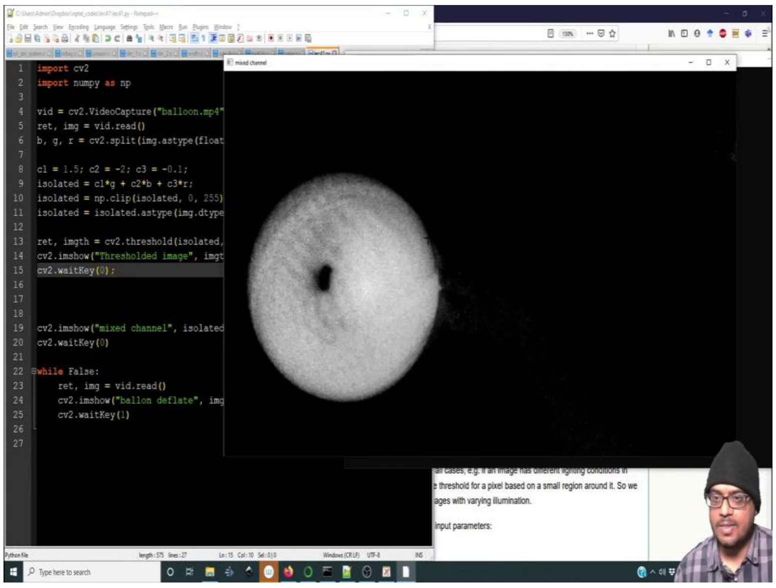
And, let me show you the image after converting it to binary, alright. So, let me execute the file. Oops I forgot to write the name this should be waitKey, alright.

(Refer Slide Time: 26:45)



So, this is the thresholded image.

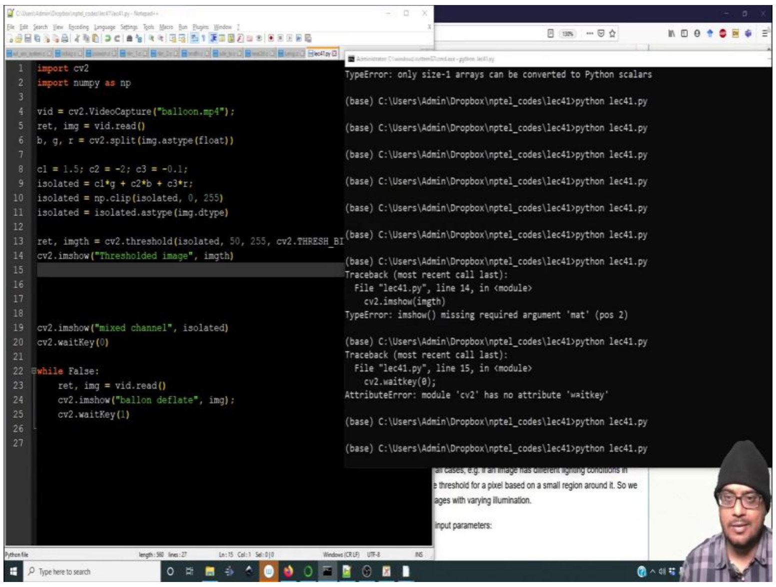
(Refer Slide Time: 26:49)



```
import cv2
import numpy as np

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

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

ret, imgth = cv2.threshold(isolated,
cv2.imshow("Thresholded image", imgt
cv2.waitKey(0);



cv2.imshow("mixed channel", isolated
cv2.waitKey(0)

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

This was the original image.

(Refer Slide Time: 26:52)

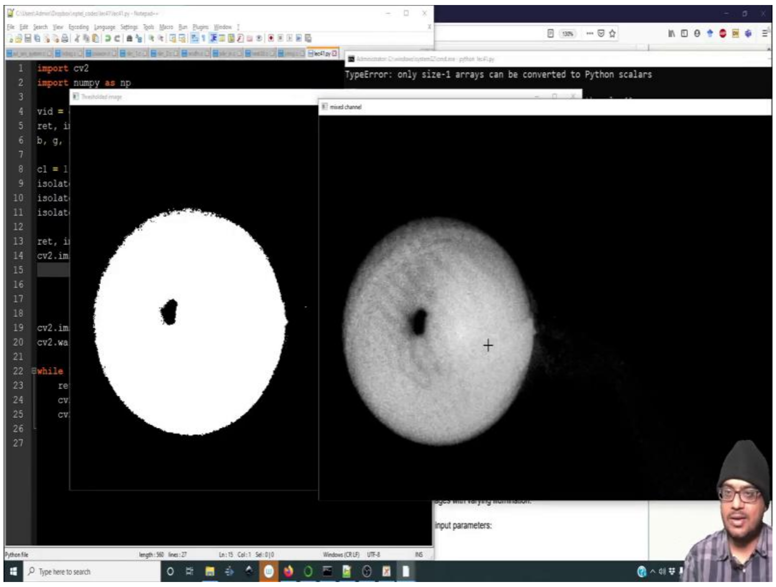


```
import cv2
import numpy as np

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

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)

ret, imgth = cv2.threshold(isolated, 50, 255, cv2.THRESH_BI
cv2.imshow("Thresholded image", imgth)




cv2.imshow("mixed channel", isolated)
cv2.waitKey(0)

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

```
TypeError: only size-1 arrays can be converted to Python scalars

(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

(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
Traceback (most recent call last):
  File "lec41.py", line 14, in <module>
    cv2.imshow(imgth)
TypeError: imshow() missing required argument 'mat' (pos 2)

(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
Traceback (most recent call last):
  File "lec41.py", line 15, in <module>
    cv2.waitkey(0);
AttributeError: module 'cv2' has no attribute 'waitkey'

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

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

In fact, let me remove this, so that the two windows are open side by side.

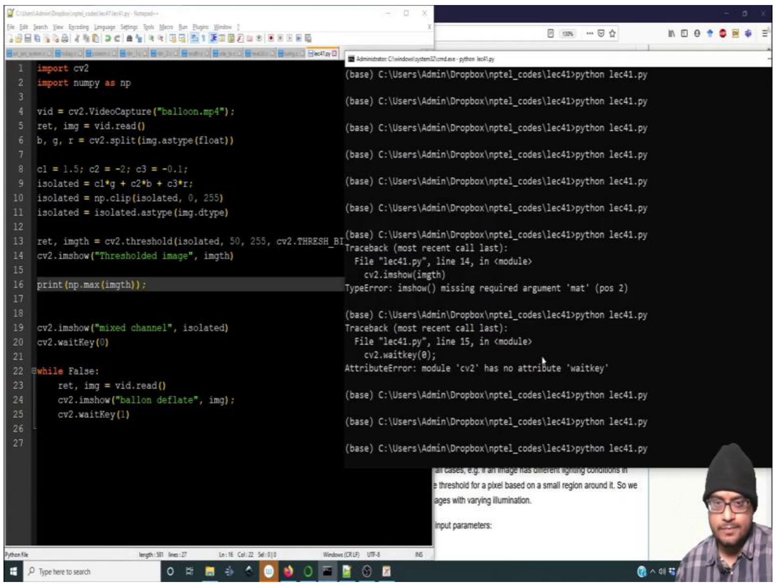
(Refer Slide Time: 26:57)


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

```
TypeError: only size-1 arrays can be converted to Python scalars
```

So, this is the mixed channel thing and this is when it is converted into binary, alright.

(Refer Slide Time: 27:06)


```
import cv2
import numpy as np

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

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)

ret, imgth = cv2.threshold(isolated, 50, 255, cv2.THRESH_BI
cv2.imshow("Thresholded image", imgth)

print(np.max(imgth));


cv2.imshow("mixed channel", isolated)
cv2.waitKey(0)

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

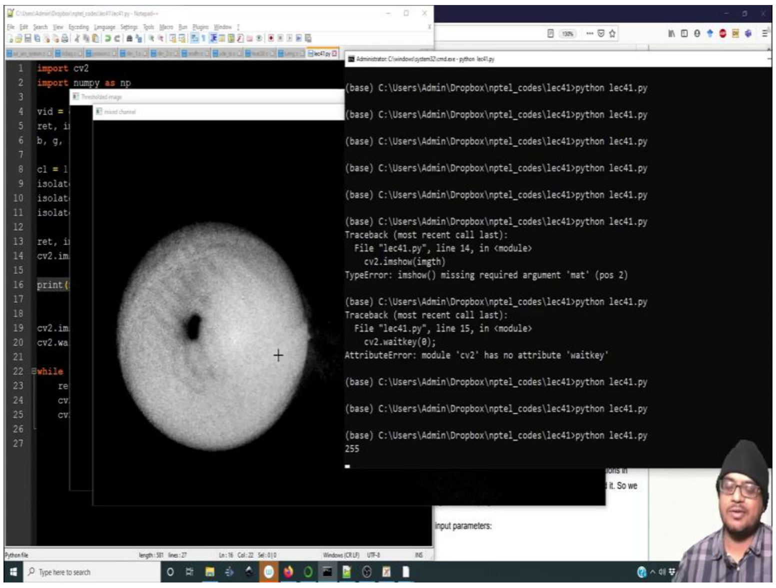
```
(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py

(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

(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
Traceback (most recent call last):
  File "lec41.py", line 14, in <module>
    cv2.imshow(imgth)
TypeError: imshow() missing required argument 'mat' (pos 2)

(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
Traceback (most recent call last):
  File "lec41.py", line 15, in <module>
    cv2.waitkey(0);
AttributeError: module 'cv2' has no attribute 'waitkey'

(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
```

e threshold for a pixel based on a small region around it. So we
ages with varying illumination.

input parameters:

So, let me close the two windows. Let me in-fact, show you what the value of image threshold will be or the maximum value. So, let us say print np.max imgth. So, let me run this.

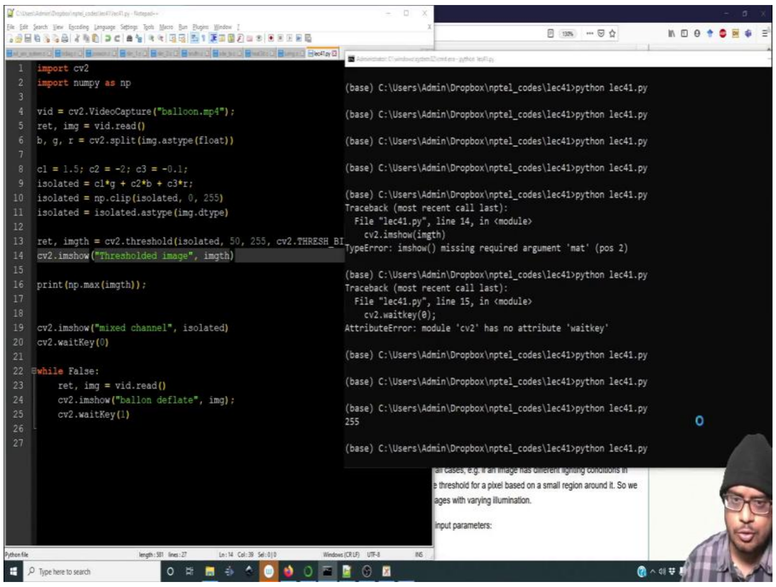
(Refer Slide Time: 27:21)



```
(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 14, in <module>
    cv2.imshow(imgth)
TypeError: imshow() missing required argument 'mat' (pos 2)

(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
Traceback (most recent call last):
  File "lec41.py", line 15, in <module>
    cv2.waitkey(0);
AttributeError: module 'cv2' has no attribute 'waitkey'

(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
255
```

So, it is 255. So, it is converted it to max and taken the other thing to 0. It is binary because there is only two levels that is 0 or 255 great.

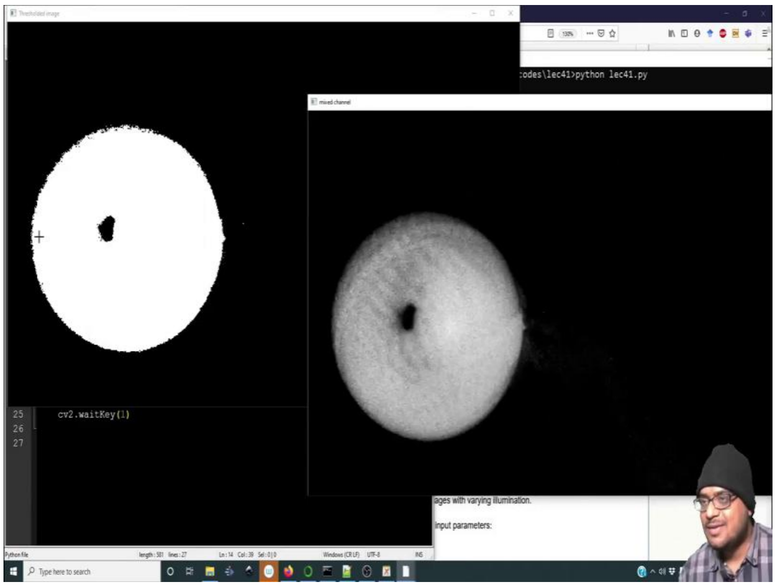
(Refer Slide Time: 27:36)



```python
import cv2
import numpy as np

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

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)

ret, imgth = cv2.threshold(isolated, 50, 255, cv2.THRESH_BI
cv2.imshow("Thresholded image", imgth)

print(np.max(imgth));


cv2.imshow("mixed channel", isolated)
cv2.waitKey(0)

while False:
    ret, img = vid.read()
    cv2.imshow("ballon deflate", img);
    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
Traceback (most recent call last):
  File "lec41.py", line 14, in <module>
    cv2.imshow(imgth)
TypeError: imshow() missing required argument 'mat' (pos 2)

(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
Traceback (most recent call last):
  File "lec41.py", line 15, in <module>
    cv2.waitkey(0);
AttributeError: module 'cv2' has no attribute 'waitkey'

(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
255

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

(Refer Slide Time: 27:42)

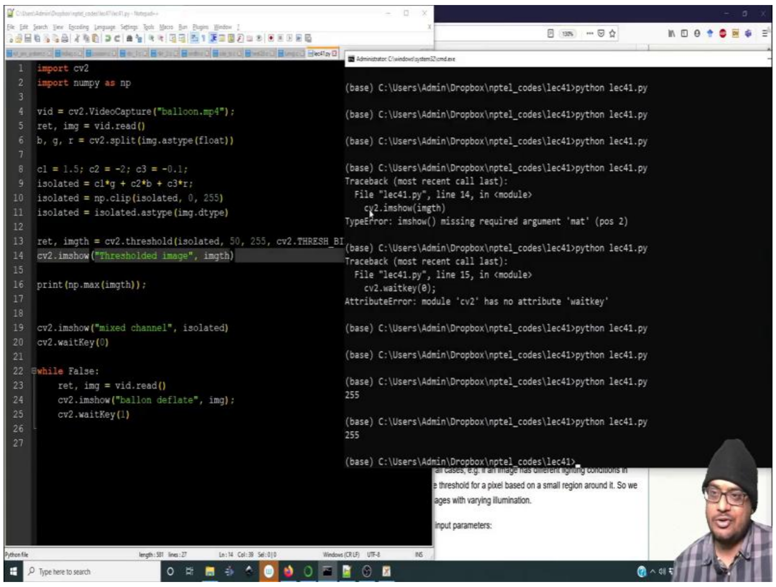


```
cv2.waitKey(1)
```

```
codes\lec41>python lec41.py
```

So, well there is a small thing I would like you to look over here is this little hazy boundary. We do not; we do not want that hazy boundary; we want it to be nice and smooth. So, in case you want to do that you must do what is called as a blurring operation on the image ok.

(Refer Slide Time: 27:59)

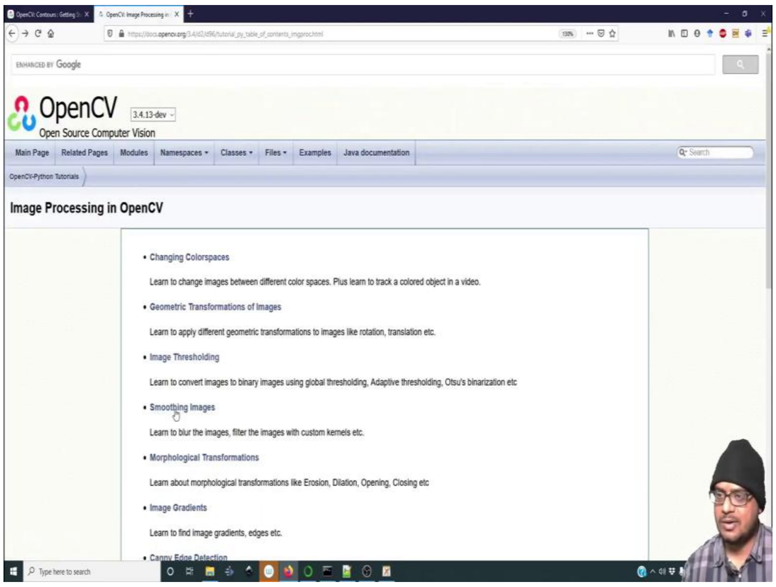


```python
import cv2
import numpy as np

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

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)

ret, imgth = cv2.threshold(isolated, 50, 255, cv2.THRESH_BI
cv2.imshow("Thresholded image", imgth)

print(np.max(imgth));


cv2.imshow("mixed channel", isolated)
cv2.waitKey(0)

while False:
    ret, img = vid.read()
    cv2.imshow("ballon deflate", img);
    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
Traceback (most recent call last):
  File "lec41.py", line 14, in <module>
    cv2.imshow(imgth)
TypeError: imshow() missing required argument 'mat' (pos 2)

(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
Traceback (most recent call last):
  File "lec41.py", line 15, in <module>
    cv2.waitkey(0);
AttributeError: module 'cv2' has no attribute 'waitkey'

(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
255

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

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

So, let me keep it like this, let me blur the image, alright. So, this is the thresholded image. Before thresholding we are going to apply a median blur where there is various kinds of blurs that exist. There is a Gaussian blur, there is a median blur. So, let me show you all the functions.

(Refer Slide Time: 28:27)



OpenCV
Open Source Computer Vision

Image Processing in OpenCV

- Changing Colorspaces

  Learn to change images between different color spaces. Plus learn to track a colored object in a video.
- Geometric Transformations of Images

  Learn to apply different geometric transformations to images like rotation, translation etc.
- Image Thresholding

  Learn to convert images to binary images using global thresholding, Adaptive thresholding, Otsu's binarization etc
- Smoothing Images

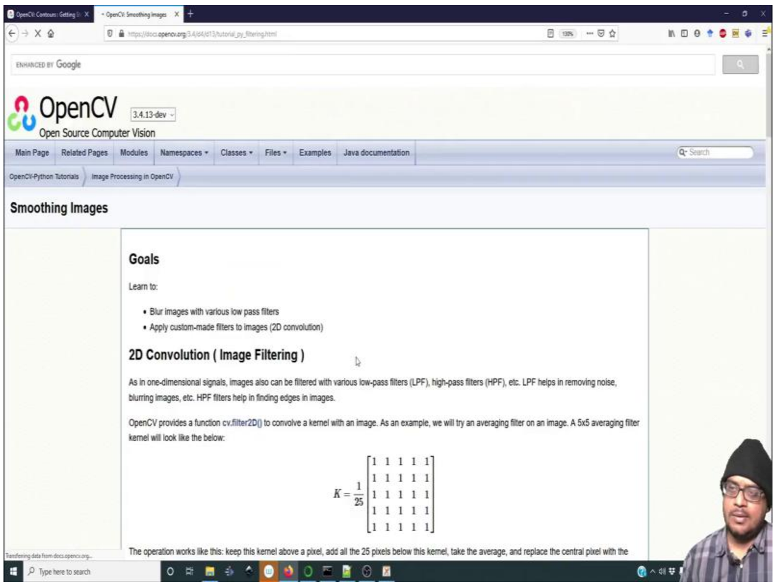
  Learn to blur the images, filter the images with custom kernels etc.
- Morphological Transformations

  Learn about morphological transformations like Erosion, Dilation, Opening, Closing etc
- Image Gradients

  Learn to find image gradients, edges etc.
- Canny Edge Detection

So, I am showing it through this because it is easier to show all the functions.

(Refer Slide Time: 28:30)



OpenCV
Open Source Computer Vision

Smoothing Images

Goals

Learn to:

- Blur images with various low pass filters
- Apply custom-made filters to images (2D convolution)

2D Convolution (Image Filtering)

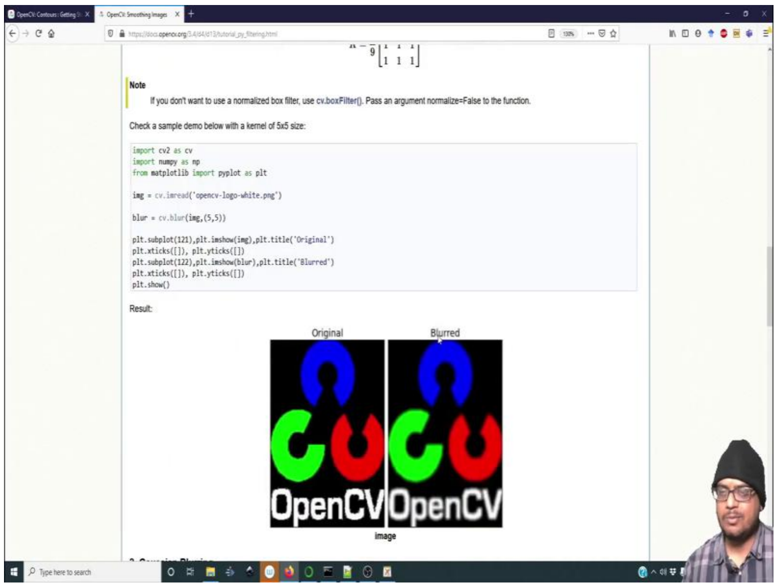
As in one-dimensional signals, images also can be filtered with various low-pass filters (LPF), high-pass filters (HPF), etc. LPF helps in removing noise, blurring images, etc. HPF filters help in finding edges in images.

OpenCV provides a function cv.filter2D() to convolve a kernel with an image. As an example, we will try an averaging filter on an image. A 5x5 averaging filter kernel will look like the below:

$$K = \frac{1}{25}\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \end{bmatrix}$$

The operation works like this: keep this kernel above a pixel, add all the 25 pixels below this kernel, take the average, and replace the central pixel with the

(Refer Slide Time: 28:31)



Note

If you don't want to use a normalized box filter, use cv.boxFilter(). Pass an argument normalize=False to the function.

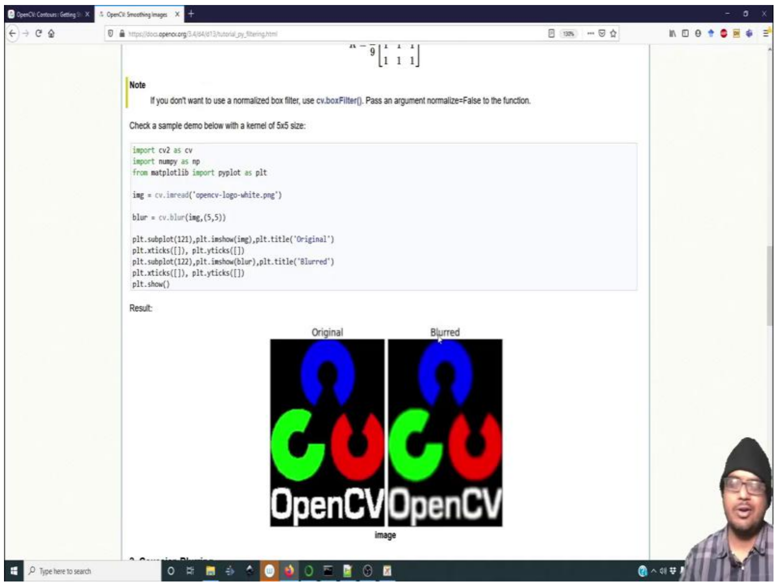
Check a sample demo below with a kernel of 5x5 size:

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

img = cv.imread('opencv-logo-white.png')

blur = cv.blur(img,(5,5))

plt.subplot(121),plt.imshow(img),plt.title('Original')
plt.xticks([]), plt.yticks([])
plt.subplot(122),plt.imshow(blur),plt.title('Blurred')
plt.xticks([]), plt.yticks([])
plt.show()
```

Result:

So, if this is the original image the blurring will cause it to become like this.

(Refer Slide Time: 28:39)



Note

If you don't want to use a normalized box filter, use cv.boxFilter(). Pass an argument normalize=False to the function.

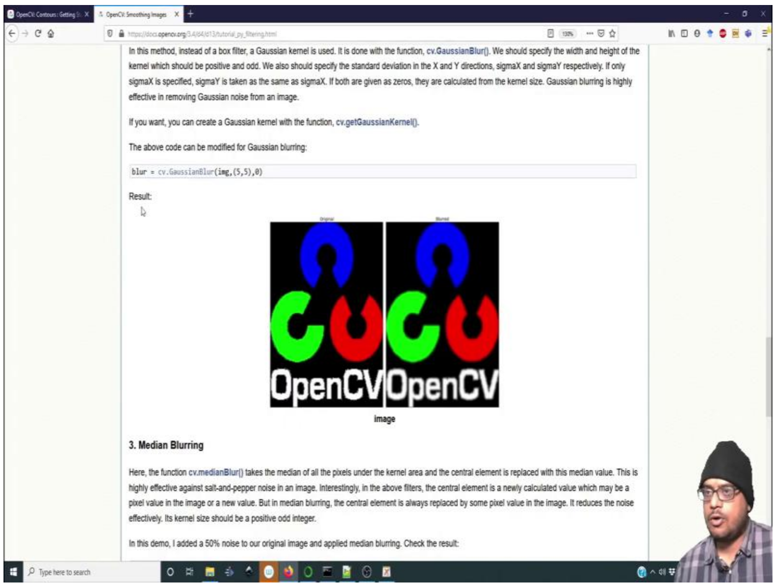
Check a sample demo below with a kernel of 5x5 size:

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

img = cv.imread('opencv-logo-white.png')

blur = cv.blur(img,(5,5))

plt.subplot(121),plt.imshow(img),plt.title('Original')
plt.xticks([]), plt.yticks([])
plt.subplot(122),plt.imshow(blur),plt.title('Blurred')
plt.xticks([]), plt.yticks([])
plt.show()
```

Result:

I am trying to show you the workflow how you go around doing it.

(Refer Slide Time: 28:43)



In this method, instead of a box filter, a Gaussian kernel is used. It is done with the function, cv.GaussianBlur(). We should specify the width and height of the kernel which should be positive and odd. We also should specify the standard deviation in the X and Y directions, sigmaX and sigmaY respectively. If only sigmaX is specified, sigmaY is taken as the same as sigmaX. If both are given as zeros, they are calculated from the kernel size. Gaussian blurring is highly effective in removing Gaussian noise from an image.

If you want, you can create a Gaussian kernel with the function, cv.getGaussianKernel().

The above code can be modified for Gaussian blurring:

```
blur = cv.GaussianBlur(img,(5,5),0)
```

Result:

image

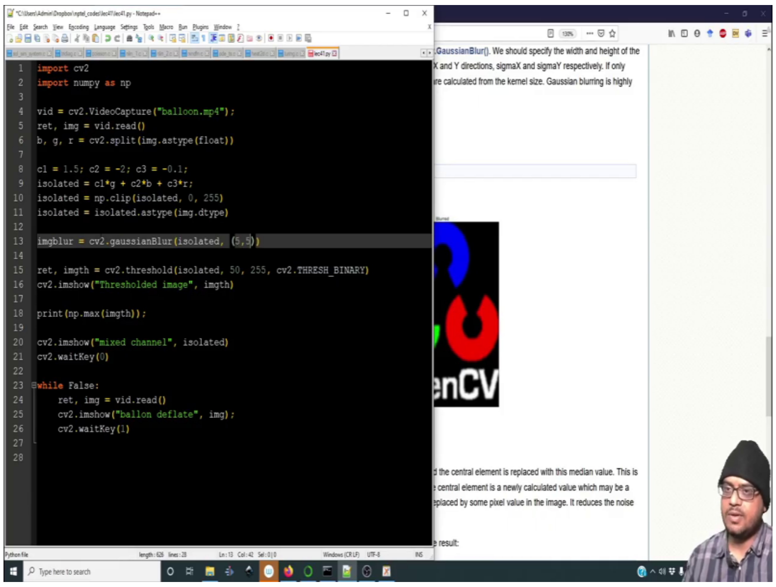
3. Median Blurring

Here, the function cv.medianBlur() takes the median of all the pixels under the kernel area and the central element is replaced with this median value. This is highly effective against salt-and-pepper noise in an image. Interestingly, in the above filters, the central element is a newly calculated value which may be a pixel value in the image or a new value. But in median blurring, the central element is always replaced by some pixel value in the image. It reduces the noise effectively. Its kernel size should be a positive odd integer.

In this demo, I added a 50% noise to our original image and applied median blurring. Check the result:

So, Gaussian blur is a very popular way of doing it and it takes a Gaussian kernel and does the blurring for you, ok.

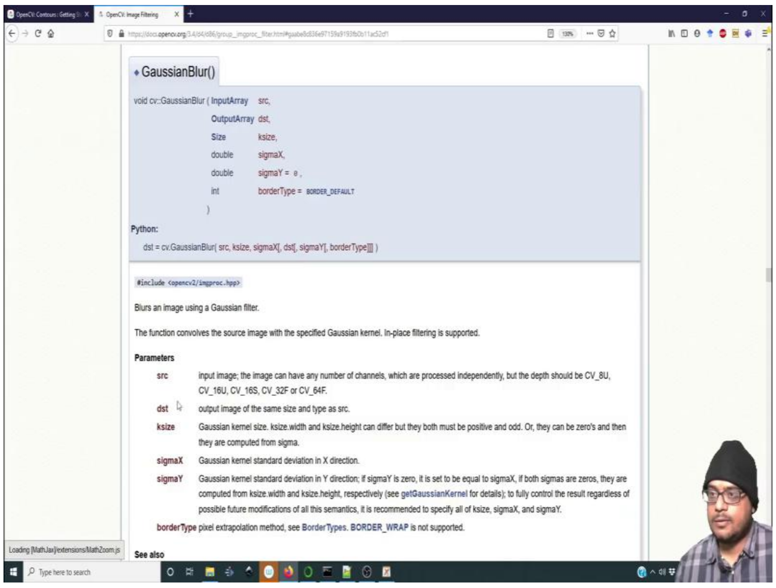
(Refer Slide Time: 28:54)



```
import cv2
import numpy as np

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

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)

imgblur = cv2.gaussianBlur(isolated, (5,5))

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

print(np.max(imgth));

cv2.imshow("mixed channel", isolated)
cv2.waitKey(0)

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

So, we are going to say imgblur = cv2.GaussianBlur. We are going to pass the image, we are going to pass the kernel size. So, what all things you need to pass? You need to pass the image on which you want to do it, you want to also do the GaussianBlur the kernel the size of the blur ok.

(Refer Slide Time: 29:17)



GaussianBlur()

```
void cv::GaussianBlur ( InputArray   src,
                        OutputArray  dst,
                        Size         ksize,
                        double       sigmaX,
                        double       sigmaY = 0,
                        int          borderType = BORDER_DEFAULT
                      )
```

Python:

```
dst = cv.GaussianBlur( src, ksize, sigmaX[, dst[, sigmaY[, borderType]]] )
```

`#include <opencv2/imgproc.hpp>`

Blurs an image using a Gaussian filter.

The function convolves the source image with the specified Gaussian kernel. In-place filtering is supported.

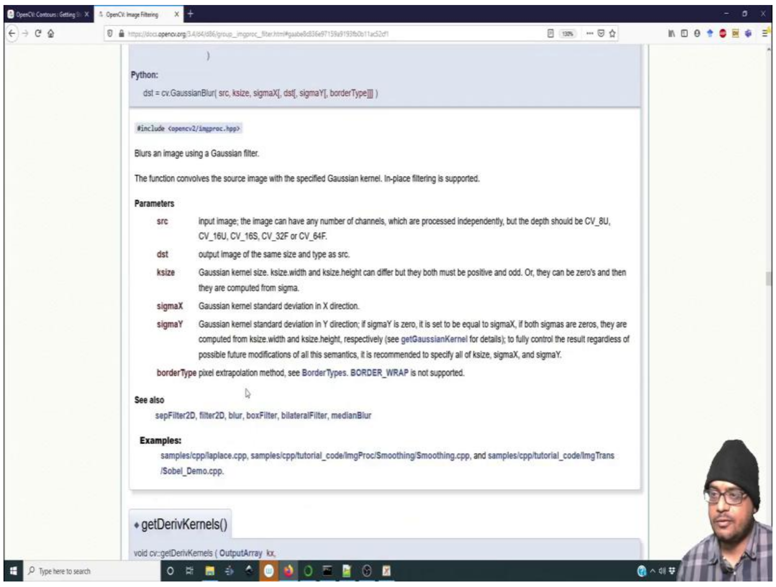
Parameters

src	input image; the image can have any number of channels, which are processed independently, but the depth should be CV_8U, CV_16U, CV_16S, CV_32F or CV_64F.
dst	output image of the same size and type as src.
ksize	Gaussian kernel size. ksize.width and ksize.height can differ but they both must be positive and odd. Or, they can be zero's and then they are computed from sigma.
sigmaX	Gaussian kernel standard deviation in X direction.
sigmaY	Gaussian kernel standard deviation in Y direction; if sigmaY is zero, it is set to be equal to sigmaX, if both sigmas are zeros, they are computed from ksize.width and ksize.height, respectively (see getGaussianKernel for details); to fully control the result regardless of possible future modifications of all this semantics, it is recommended to specify all of ksize, sigmaX, and sigmaY.
borderType	pixel extrapolation method, see BorderTypes. BORDER_WRAP is not supported.

See also

So, the parameters. So, this is the C++ version of it.

(Refer Slide Time: 29:24)



```
)
```

Python:

```
dst = cv.GaussianBlur( src, ksize, sigmaX[, dst[, sigmaY[, borderType]]] )
```

`#include <opencv2/imgproc.hpp>`

Blurs an image using a Gaussian filter.

The function convolves the source image with the specified Gaussian kernel. In-place filtering is supported.

Parameters

src	input image; the image can have any number of channels, which are processed independently, but the depth should be CV_8U, CV_16U, CV_16S, CV_32F or CV_64F.
dst	output image of the same size and type as src.
ksize	Gaussian kernel size. ksize.width and ksize.height can differ but they both must be positive and odd. Or, they can be zero's and then they are computed from sigma.
sigmaX	Gaussian kernel standard deviation in X direction.
sigmaY	Gaussian kernel standard deviation in Y direction; if sigmaY is zero, it is set to be equal to sigmaX, if both sigmas are zeros, they are computed from ksize.width and ksize.height, respectively (see getGaussianKernel for details); to fully control the result regardless of possible future modifications of all this semantics, it is recommended to specify all of ksize, sigmaX, and sigmaY.
borderType	pixel extrapolation method, see BorderTypes. BORDER_WRAP is not supported.

See also

sepFilter2D, filter2D, blur, boxFilter, bilateralFilter, medianBlur

Examples:

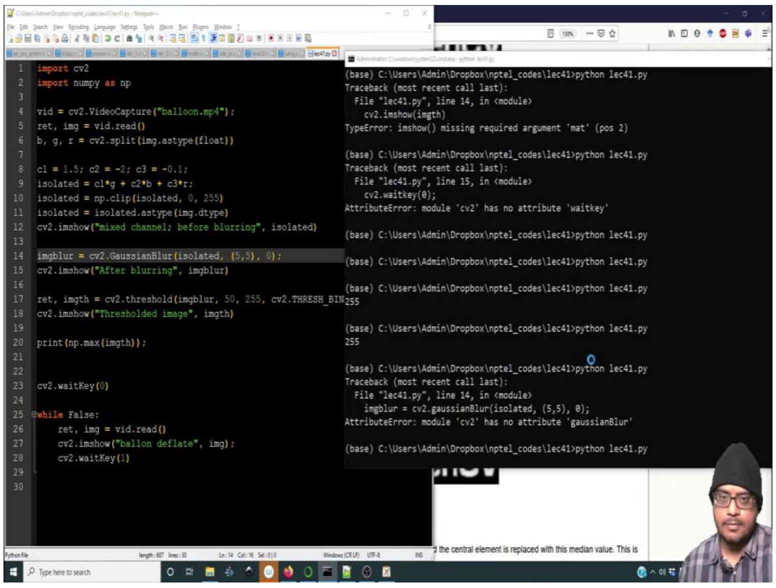
samples/cpp/laplace.cpp, samples/cpp/tutorial_code/ImgProc/Smoothing/Smoothing.cpp, and samples/cpp/tutorial_code/ImgTrans/Sobel_Demo.cpp.

getDerivKernels()

```
void cv::getDerivKernels ( OutputArray  kx,
```

But, the C version also works in a similar fashion ok, alright. So, based on the kernel size it is going to figure out what the standard deviations are going to be and typically we keep it 0 to keep everything normalized, alright.

(Refer Slide Time: 29:52)

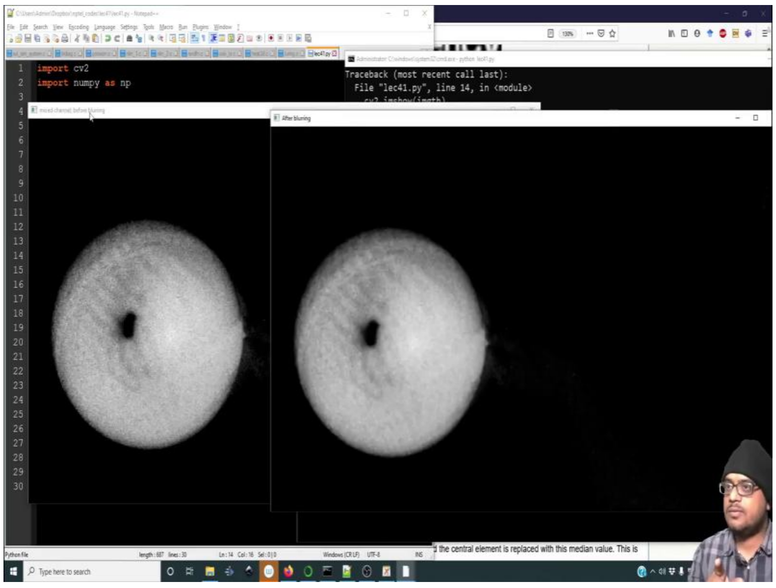


```python
import cv2
import numpy as np

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

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)

imgblur = cv2.GaussianBlur(isolated, (5,5), 0);
cv2.imshow("After blurring", imgblur)

ret, imgth = cv2.threshold(imgblur, 50, 255, cv2.THRESH_BIN

cv2.imshow("Thresholded image", imgth)

print(np.max(imgth));


cv2.waitKey(0)

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

```
(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
Traceback (most recent call last):
  File "lec41.py", line 14, in <module>
    cv2.imshow(imgth)
TypeError: imshow() missing required argument 'mat' (pos 2)

(base) C:\Users\Admin\Dropbox\nptel_codes\lec41>python lec41.py
Traceback (most recent call last):
  File "lec41.py", line 15, in <module>
    cv2.waitkey(0);
AttributeError: module 'cv2' has no attribute 'waitkey'

(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
255

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

(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, (5,5), 0);
AttributeError: module 'cv2' has no attribute 'gaussianBlur'

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

So, isolated, (5,5), 0. So, once it is done we going to well we can show the blurred image also. So, let me do that. So, let me say cv2.imshow("After blurring", imgblur). Let me take this over here makes channel before blurring this is after blurring, this is after threshold. Now, a threshold will be actually applied on imgblur, alright. So, let me go over here. This g should be a capital G ok, yeah.

(Refer Slide Time: 30:43)

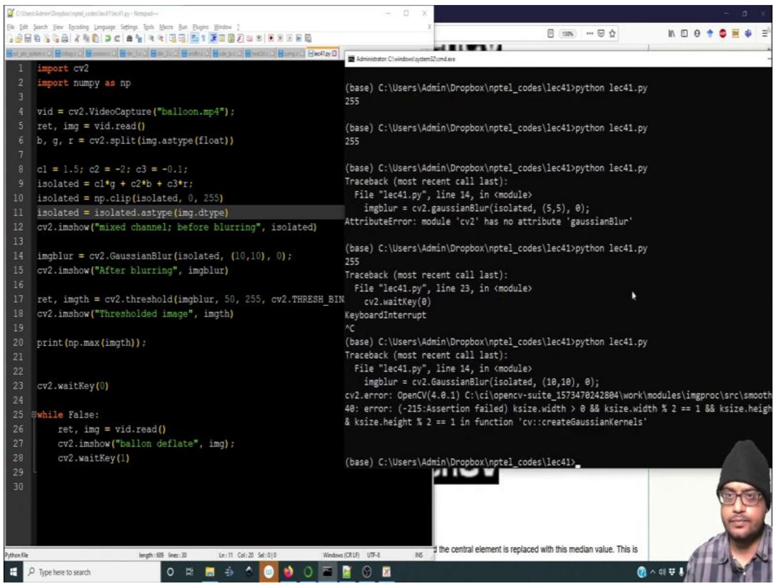


```python
import cv2
import numpy as np
```

```
Traceback (most recent call last):
  File "lec41.py", line 14, in <module>
```

So, look at the two images. So, this is before blurring and this is after blurring. So, the edges seem to have smoothened out.

(Refer Slide Time: 30:58)

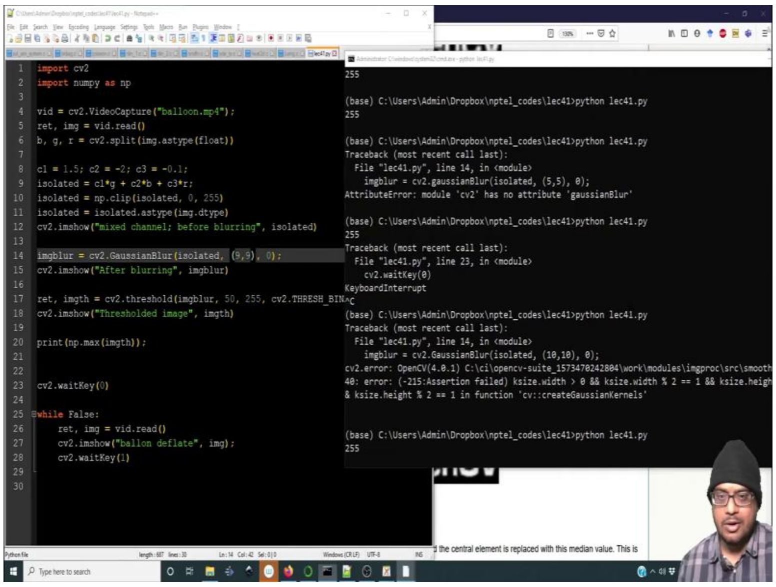


```python
import cv2
import numpy as np

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

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)

imgblur = cv2.GaussianBlur(isolated, (10,10), 0);
cv2.imshow("After blurring", imgblur)

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

print(np.max(imgth));


cv2.waitKey(0)

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

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

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

(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, (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>
```

In fact, let me increase the size of the kernel to 10 × 10. So, you have a larger area of blurring I have to close the windows you need to give it an odd number.

(Refer Slide Time: 31:22)

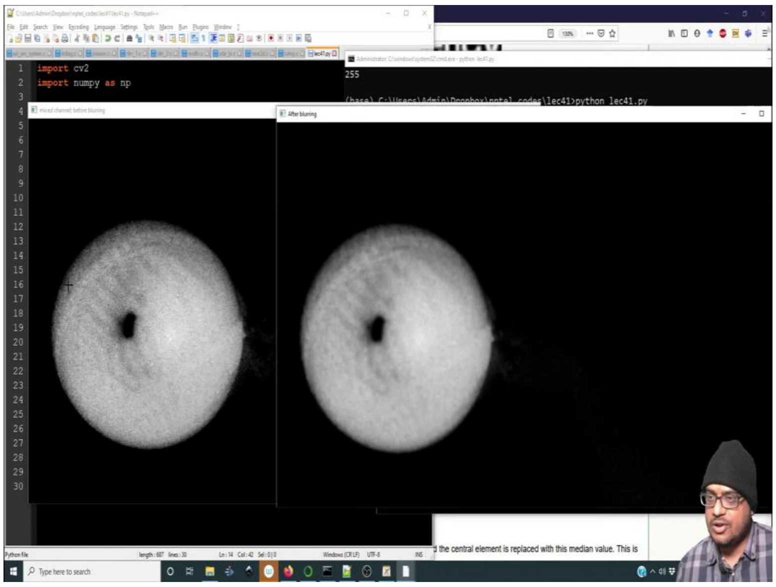


```python
import cv2
import numpy as np

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

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)

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

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

print(np.max(imgth));


cv2.waitKey(0)

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

```
255

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

(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, (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
```

So, you need to give it an odd number, you cannot give it an even number because it is going to center and so, if it is 1 it is centered, if it is 3, then the nearest neighbours are taken if it is 5 then two on this side, two on this side, two above, two below ok that is why it needs to be an odd number ok.

(Refer Slide Time: 31:41)



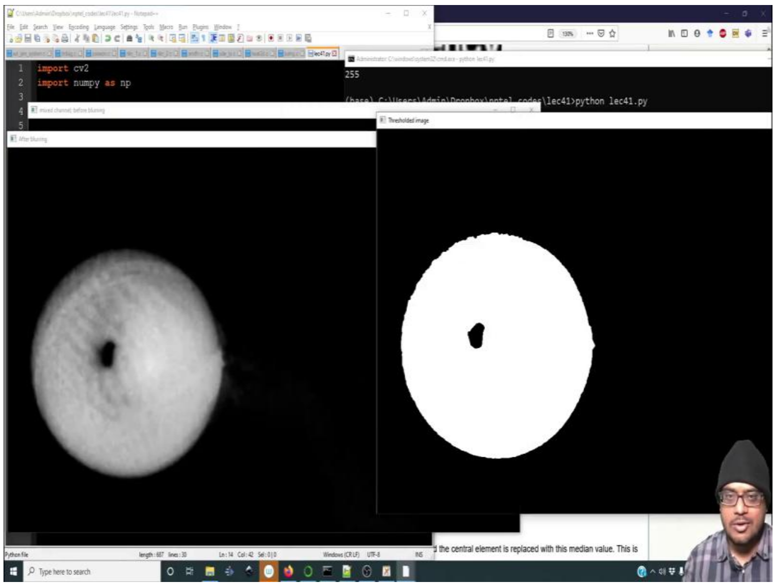
```
import cv2
import numpy as np
```

```
255

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

After blurring
d the central element is replaced with this median value. This is
Type here to search

So, this is the original image, this is the blurred image. So, all those freckles on the boundary because of the bad lighting sort of smoothened out.

(Refer Slide Time: 31:52)



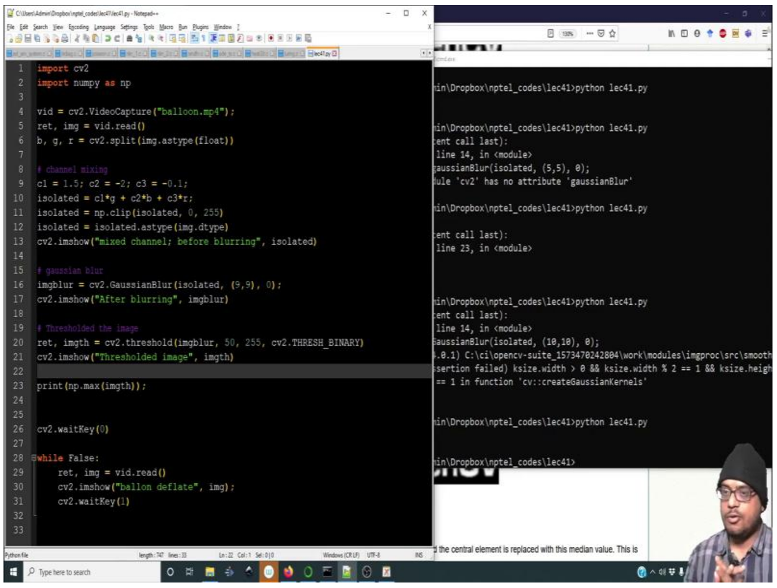
```
import cv2
import numpy as np
```

```
255

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

After blurring
Thresholded image
d the central element is replaced with this median value. This is
Type here to search

Once it is smoothened out you get a nice contour well there are some things going on, but anyway you can actually do a medium blur and get rid of that as well but, we do not bother so much about that. We will get a good answer regardless ok.

(Refer Slide Time: 32:03)



```python
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_BINARY)
cv2.imshow("Thresholded image", imgth)

print(np.max(imgth));


cv2.waitKey(0)

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

```

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


in\Dropbox\nptel_codes\lec41>python lec41.py
ent call last):
 line 14, in <module>
aussianBlur(isolated, (5,5), 0);
ule 'cv2' has no attribute 'gaussianBlur'

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

ent call last):
 line 23, in <module>



in\Dropbox\nptel_codes\lec41>python lec41.py
ent call last):
 line 14, in <module>
aussianBlur(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>
```

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

So, so far we have seen the following. We have seen this is the channel mixing, once we have mixed the channels we have performed a gaussian blur. After doing a gaussian blur, we have thresholded the image.

These are the three basic things we have done. In the next video, we are going to move ahead on this and we are going to combine all this in the form of a continuous evaluation of all the frames. So, with this I end this particular lecture, I will see you next time. Bye.