Applying ML algorithms for prediction in non-IT fields

Supervisor: Dr. Mrad Mohamed Azouz

Author: Mohamed Khaled, BSc student at BME

email: mohamedkhaledelb@gmail.com

Neptun code: QRSSBO

The goal of the project:

The aim of this project is to develop an image processing application that will be able to identify different droplets using its edges in its three different phases: Developing, just before detachment from the dropper and after detachment.. and be able to identify the next dropper in the same way.

When successfully identifying the droplet we need to create functions that calculate the area and the perimeter of the droplet.

Tools used: OpenCV with Python

What is opencv ?

OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library, which is aimed at real-time computer vision.

The library is cross-platform it can support Python, Java , C++ etc.

It was originally developed by intel and it is free for use under the open-source BSD license.

OpenCV's application areas include:

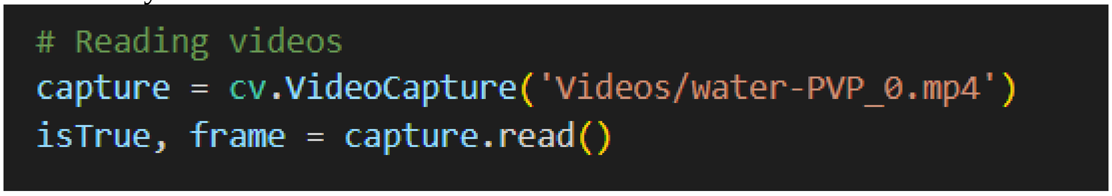
- 2D and 3D feature toolkits
- Facial recognition
- Huma-computer interaction
- Gesture recognition
- Motion understanding
- Object identification
- Segmentation and recognition
- Motion tracking

What is image processing ?:

It is the process of transforming an image to a digital form and performing certain operations to get some useful info from it. The image processing system usually treats all the images as a 2D signals when applying certain predetermined signal processing methods.

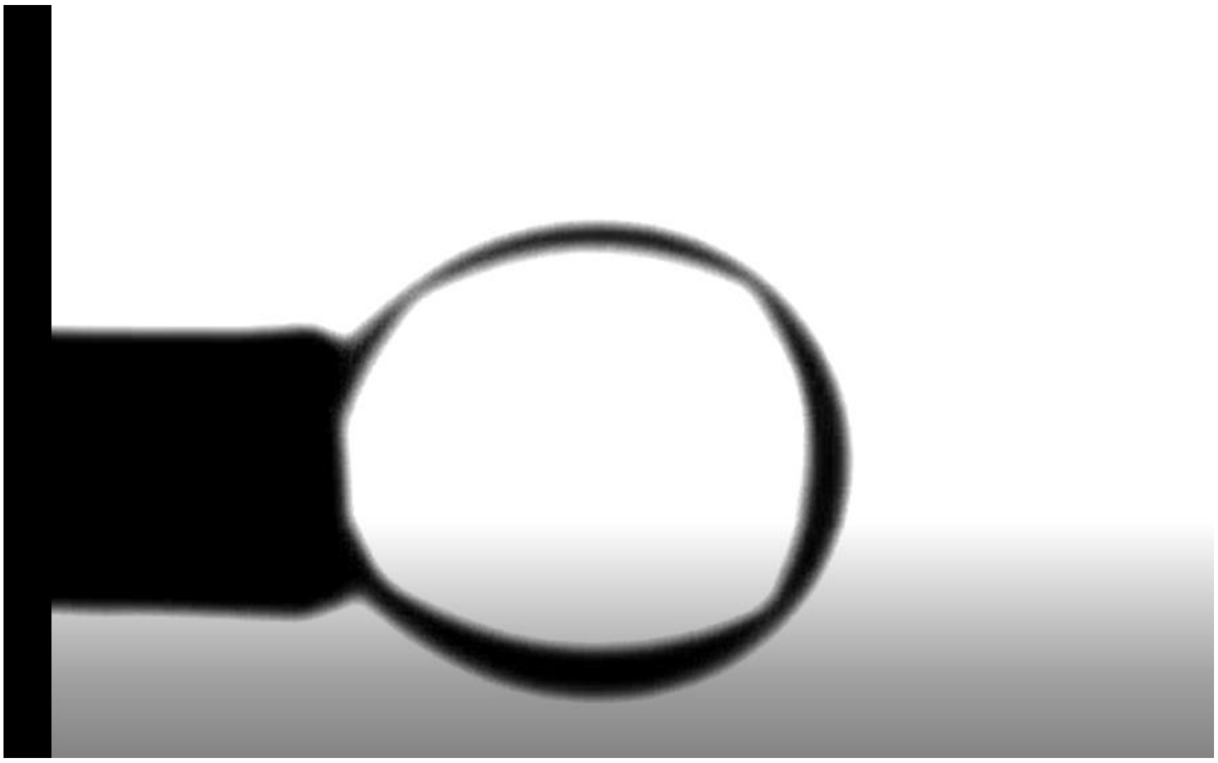
Coding and implementation steps and details

First we need to read the video from the device for that we will use the following code in the screenshot which basically read the video frame by frame , we need to pass the path to the video to the (VideoCapture) method, then use the read method which basically return two things , first Boolean value true or false depending on if it successfully read the frame or not and the frame itself.



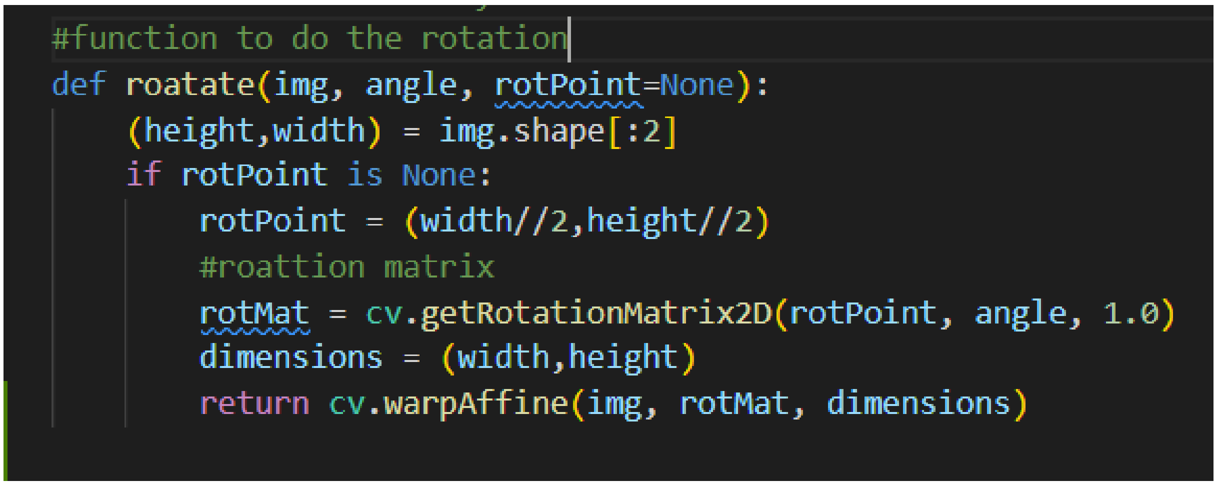
```
# Reading videos
capture = cv.VideoCapture('Videos/water-PVP_0.mp4')
isTrue, frame = capture.read()
```

In the screenshot below this is how the droplets drop from the dropper they are dropping in horizontal way which is not logic so, we should do some rotation to all the video frames.



So, how the rotation function works ?
It takes three parameters, first one is the source image or frame we want to rotate and the second one is the angle of rotation and the third is the rotation point and since it is none we will rotate around the center.

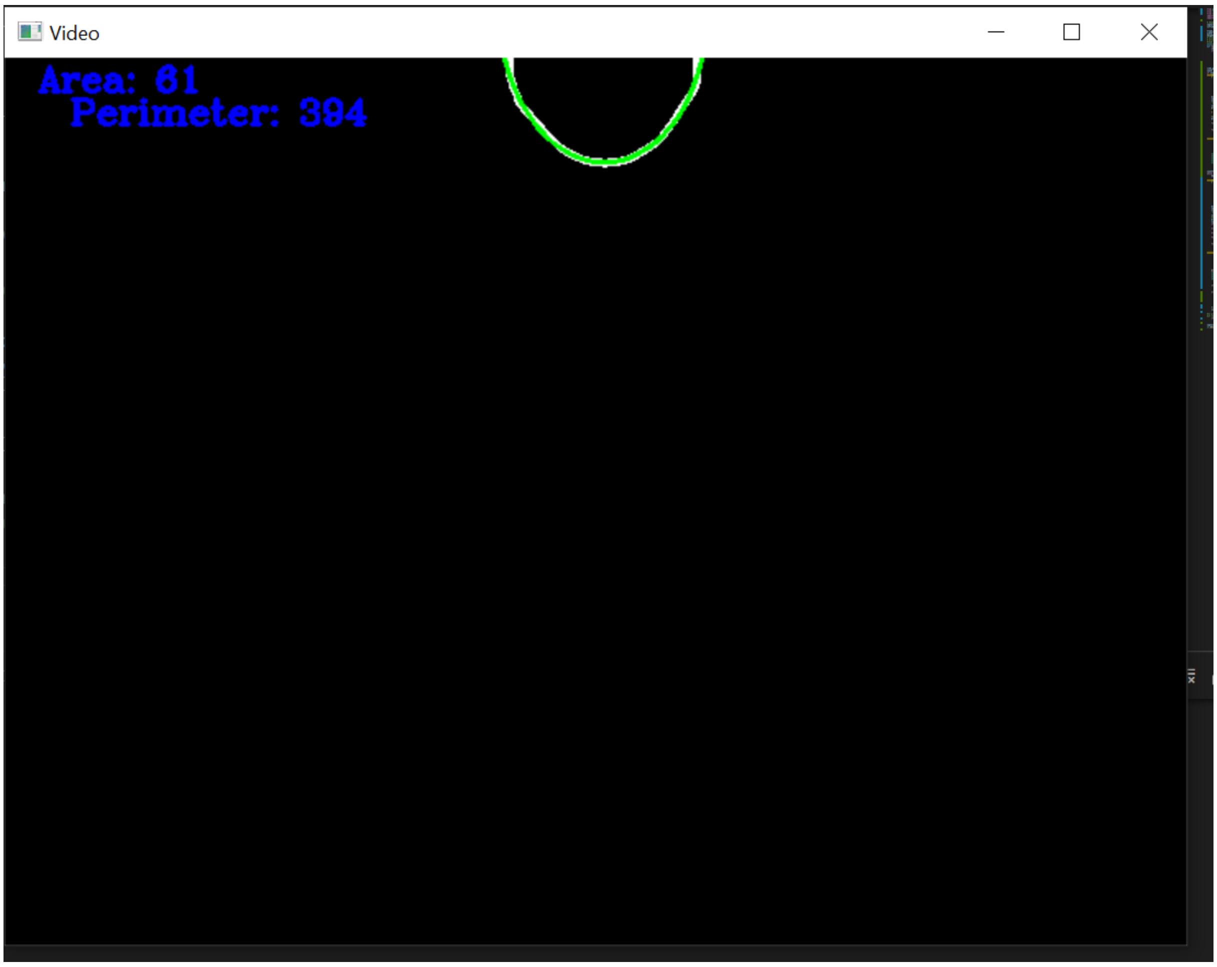
Then we need the rotation matrix after that we return the rotated image using cv.warpAffine method.



```python
#function to do the rotation
def roatate(img, angle, rotPoint=None):
    (height,width) = img.shape[:2]
    if rotPoint is None:
        rotPoint = (width//2,height//2)
        #roattion matrix
        rotMat = cv.getRotationMatrix2D(rotPoint, angle, 1.0)
        dimensions = (width,height)
        return cv.warpAffine(img, rotMat, dimensions)
```

How does the application work ?

Our application starts by reading the first frame only of the video and displays the area and the perimeter of the contour, then if you want to continue reading the rest of frames press letter ‘c’ on the keyboard , if you wish to stop at any moment just press space, if you want to terminate and close the application press ‘d’.



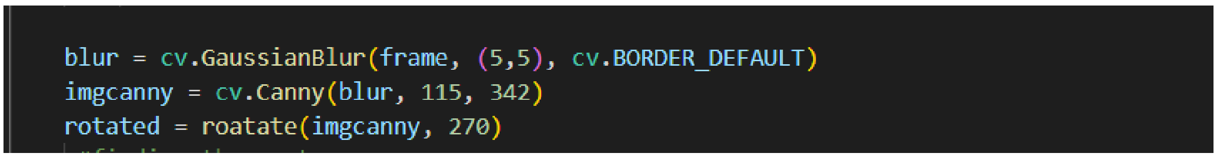
Continue implementation details:-

After reading the video frames I noticed the existence of some noise -some dots on the screen that will affect our result.

I tried some solutions for example by drawing on them but this emerge to be wrong but after doing some bluring to the video frames those dots disappeared.

I used cv.GuassianBlur function for this purpose , this function takes three parameters,

First one is the source image or frame second is the kernel size the third one is the standard deviation.



```
blur = cv.GaussianBlur(frame, (5,5), cv.BORDER_DEFAULT)
imgcanny = cv.Canny(blur, 115, 342)
rotated = roatate(imgcanny, 270)
```

Then I used canny edge detector to find the edges of the video, using cv.Canny method, This method takes three parameters also , first one is the source image and second and the third are the thresholds values, after some trials the value of the upper threshold and lower threshold were 115 and 342 respectively.

After that we need to do the rotation using rotate function that we did previously with angle 270.

Contours detection

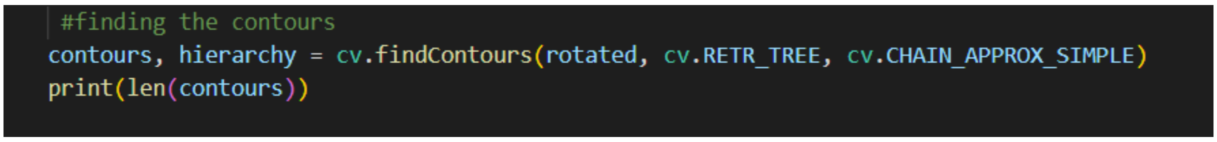
the next step is to find the contours of the frames from which we will be getting the data we need such as the perimeter and the area.

But first what are the contours ?

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

For a better accuracy we should use the binary images, so first we should apply thresholds or canny edge detection as we did.

Finding contours is like finding white object from black background.



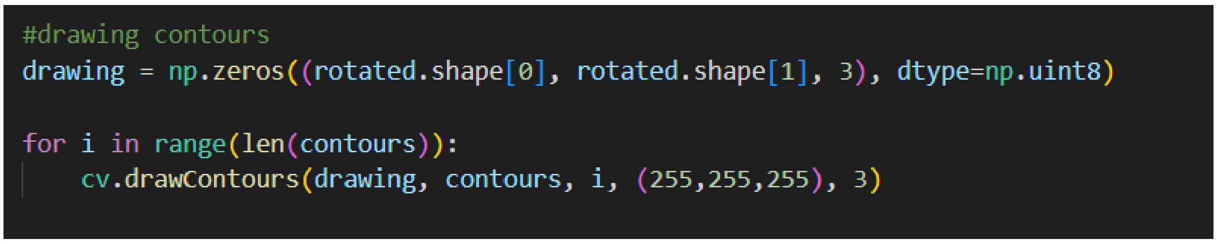
```python
#finding the contours
contours, hierarchy = cv.findContours(rotated, cv.RETR_TREE, cv.CHAIN_APPROX_SIMPLE)
print(len(contours))
```

We used cv.findContours method to get the contours , it takes three parameters, first one is the source image or frame, second one is the mode and the third is the approximation method.

This method returns two things , contours and hierarchies, the second one is out of our scope, but the contours is a python list of all the contpurs in the image. Each individual contour is a Numpy array of (x,y) coordinates of the boundary points of the object.

After finding the contours we have to draw them.

How to draw the contours ?

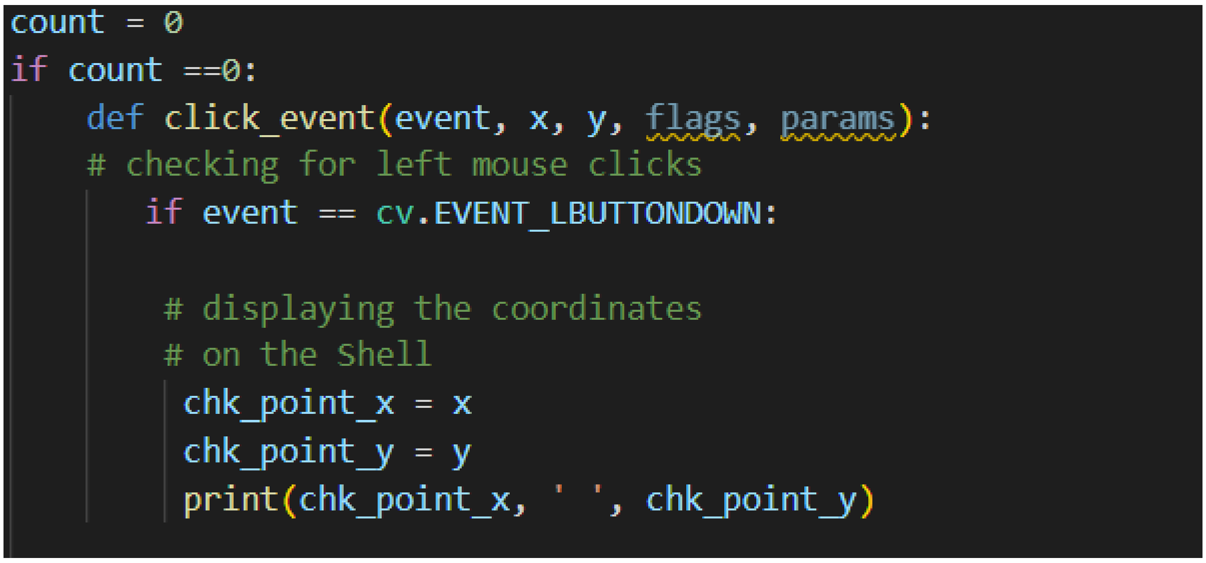


```python
#drawing contours
drawing = np.zeros((rotated.shape[0], rotated.shape[1], 3), dtype=np.uint8)

for i in range(len(contours)):
    cv.drawContours(drawing, contours, i, (255,255,255), 3)
```

First I used the np.zeros method which will return a new array of the given shape and type filled with zeros.

After that I created a for loop to iterate through the given contours and draw them using cv.drawContours method, and we pass the needed parameters to it which are the frame, the contours , the index of the contour, the color and the thickness.

After drawing the contours I tried to remove the small remaining static part in order not to affect the calculation so much, that is why I created this click_event function which will store the position of the mouse click in a variable and remove the points above this point from the contours but it does not work correctly for me so there will be a small error in the calculation.



```python
count = 0
if count ==0:
    def click_event(event, x, y, flags, params):
    # checking for left mouse clicks
       if event == cv.EVENT_LBUTTONDOWN:

        # displaying the coordinates
        # on the Shell
         chk_point_x = x
         chk_point_y = y
         print(chk_point_x, ' ', chk_point_y)
```

Calculating data:-

As we stated we will start by reading only the first frame and we will read the next ones by pressing any button then we can stop by pressing space.

So, first we will calculate the required data in the first frame and we will follow mostly the same technique in the next frames.

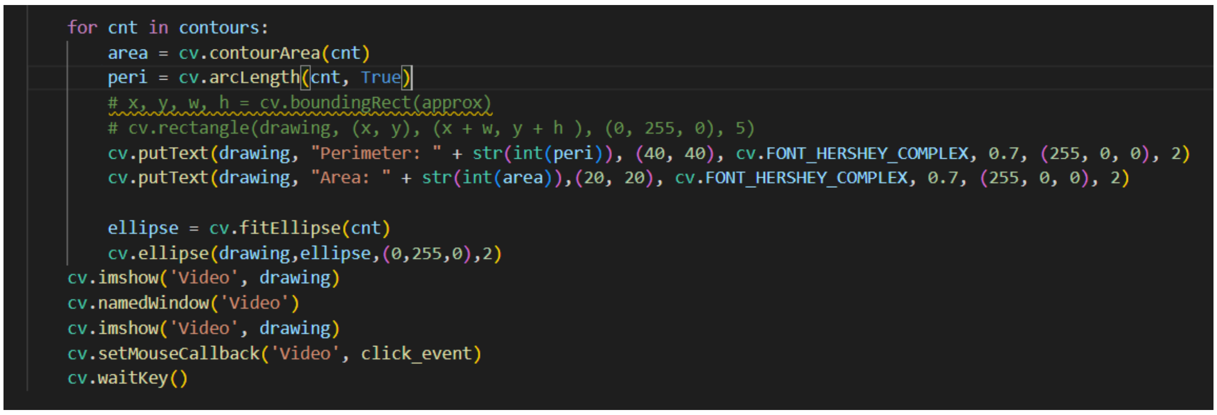
So, I used cv.contourAarea(cnt) and I passed the contour as a parameter this will give us the area bounded by the contours.

And I used cv.arcLength(cnt, True) to get the length or the perimeter of the contours , the second argument specify whether the shape is a closed contour or it is just a curve.

After this I did a contour approximation to the shape.

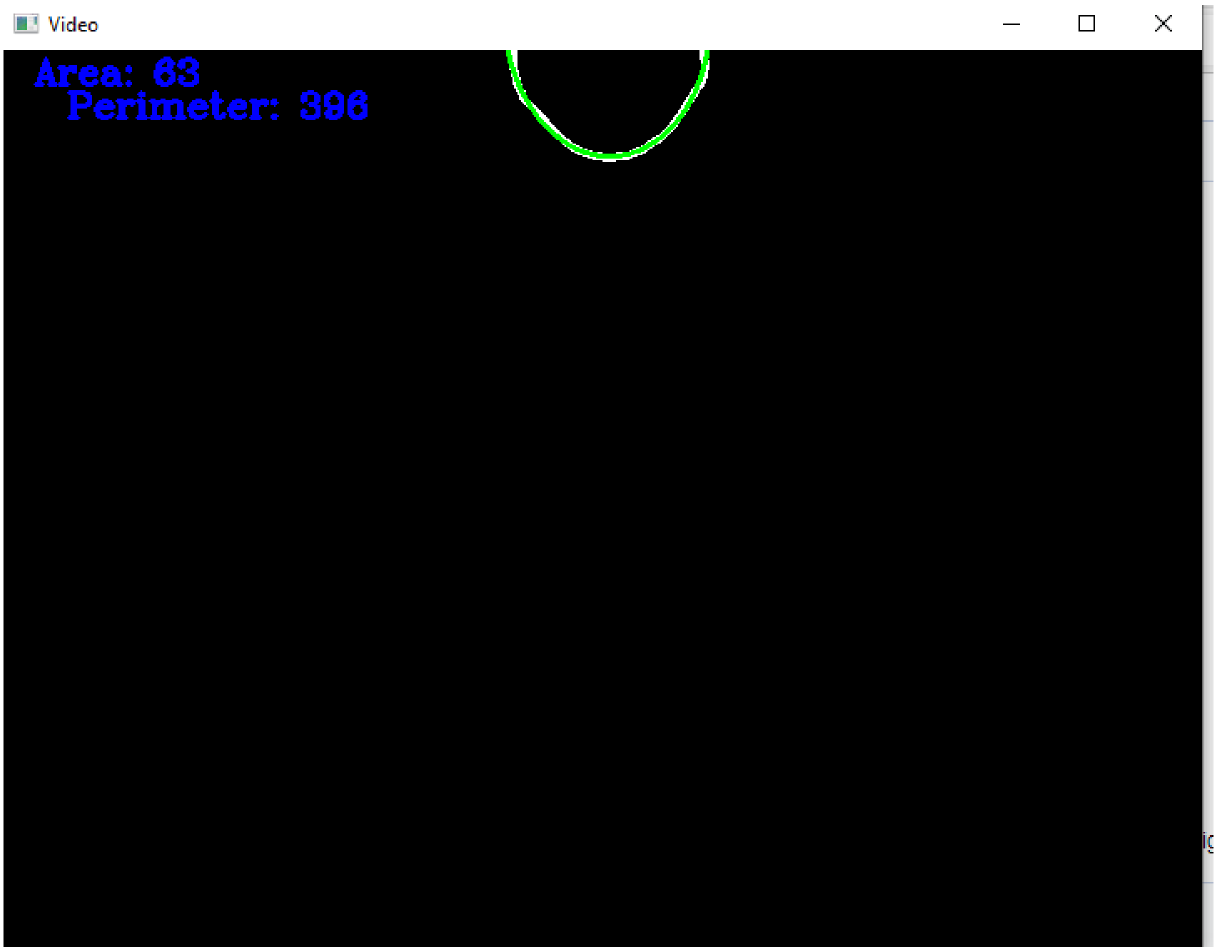
What contour approximation does ?

It approximate the contour to a shape with less number of vertices depending upon the precision we specify, I tried first to approximate it to a rectangle but it makes more sense to approximate it to ellipse.



```
    for cnt in contours:
        area = cv.contourArea(cnt)
        peri = cv.arcLength(cnt, True)
        # x, y, w, h = cv.boundingRect(approx)
        # cv.rectangle(drawing, (x, y), (x + w, y + h ), (0, 255, 0), 5)
        cv.putText(drawing, "Perimeter: " + str(int(peri)), (40, 40), cv.FONT_HERSHEY_COMPLEX, 0.7, (255, 0, 0), 2)
        cv.putText(drawing, "Area: " + str(int(area)),(20, 20), cv.FONT_HERSHEY_COMPLEX, 0.7, (255, 0, 0), 2)

        ellipse = cv.fitEllipse(cnt)
        cv.ellipse(drawing,ellipse,(0,255,0),2)
    cv.imshow('Video', drawing)
    cv.namedWindow('Video')
    cv.imshow('Video', drawing)
    cv.setMouseCallback('Video', click_event)
    cv.waitKey()
```

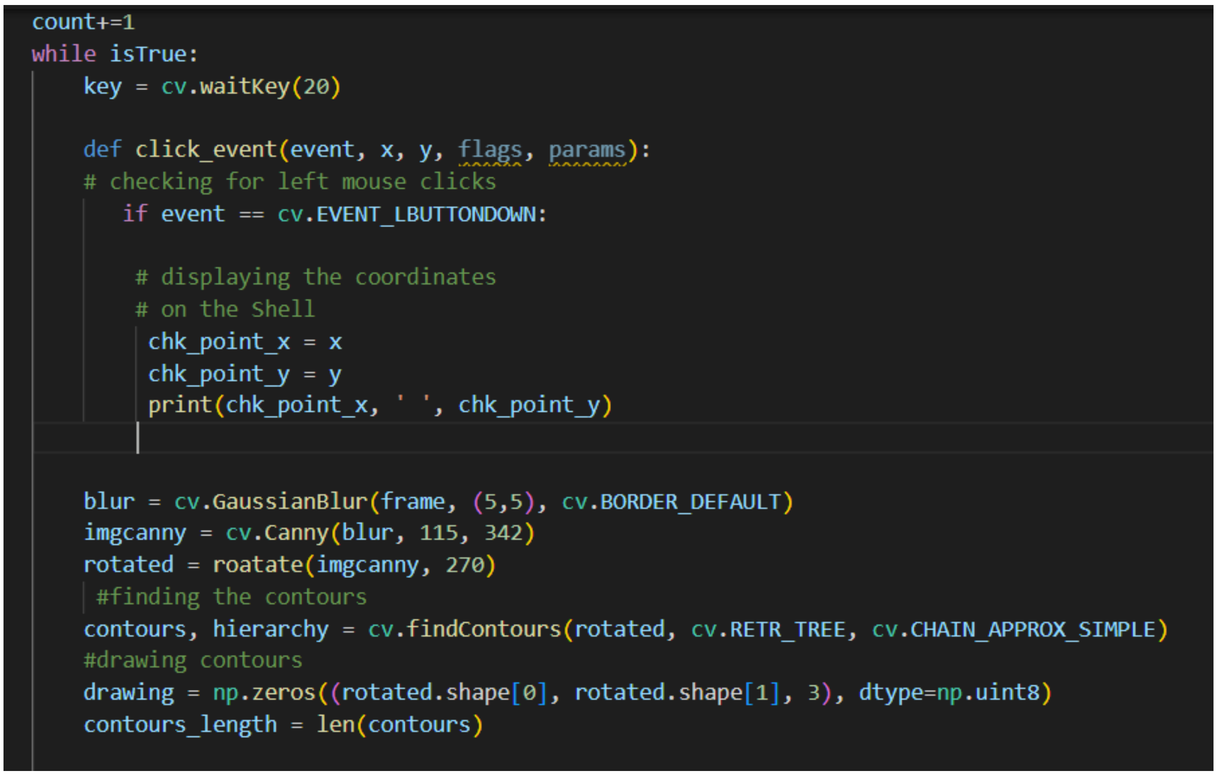
Then we print the area and the perimeter to the console using cv.putText method.



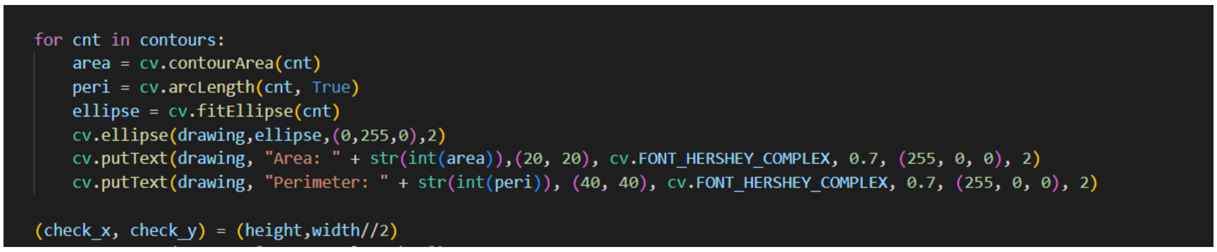
```
Area: 63
  Perimeter: 396
```

Then I increased the counter to indicate that we are moving to the next frame which will be inside a while loop that keeps reading the video till it ends or we can force it to end by pressing ‘d’ or simply stop running the application.

Then inside the loop the code was mostly the same as we did in the first frame except for determining the state of the droplet whether it is developing or detached.



```
count+=1
while isTrue:
    key = cv.waitKey(20)

    def click_event(event, x, y, flags, params):
    # checking for left mouse clicks
        if event == cv.EVENT_LBUTTONDOWN:

          # displaying the coordinates
          # on the Shell
            chk_point_x = x
            chk_point_y = y
            print(chk_point_x, ' ', chk_point_y)


    blur = cv.GaussianBlur(frame, (5,5), cv.BORDER_DEFAULT)
    imgcanny = cv.Canny(blur, 115, 342)
    rotated = roatate(imgcanny, 270)
     #finding the contours
    contours, hierarchy = cv.findContours(rotated, cv.RETR_TREE, cv.CHAIN_APPROX_SIMPLE)
    #drawing contours
    drawing = np.zeros((rotated.shape[0], rotated.shape[1], 3), dtype=np.uint8)
    contours_length = len(contours)
```



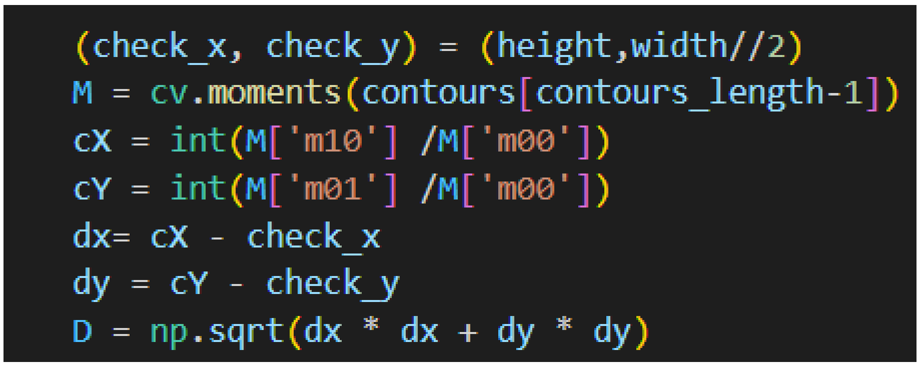
```
for cnt in contours:
    area = cv.contourArea(cnt)
    peri = cv.arcLength(cnt, True)
    ellipse = cv.fitEllipse(cnt)
    cv.ellipse(drawing,ellipse,(0,255,0),2)
    cv.putText(drawing, "Area: " + str(int(area)),(20, 20), cv.FONT_HERSHEY_COMPLEX, 0.7, (255, 0, 0), 2)
    cv.putText(drawing, "Perimeter: " + str(int(peri)), (40, 40), cv.FONT_HERSHEY_COMPLEX, 0.7, (255, 0, 0), 2)

(check_x, check_y) = (height,width//2)
```

How to determine the state of the droplet ?:

My idea was to find a point in the bottom of the frame which is (height, width//2), then we look for the distance between this point and the Centroid of the contour.

How to find the centroid of the contours ?:

For this we used cv.moments menthod , image moments helps us to calculate some features like center of mass of the object, area of the object etc.

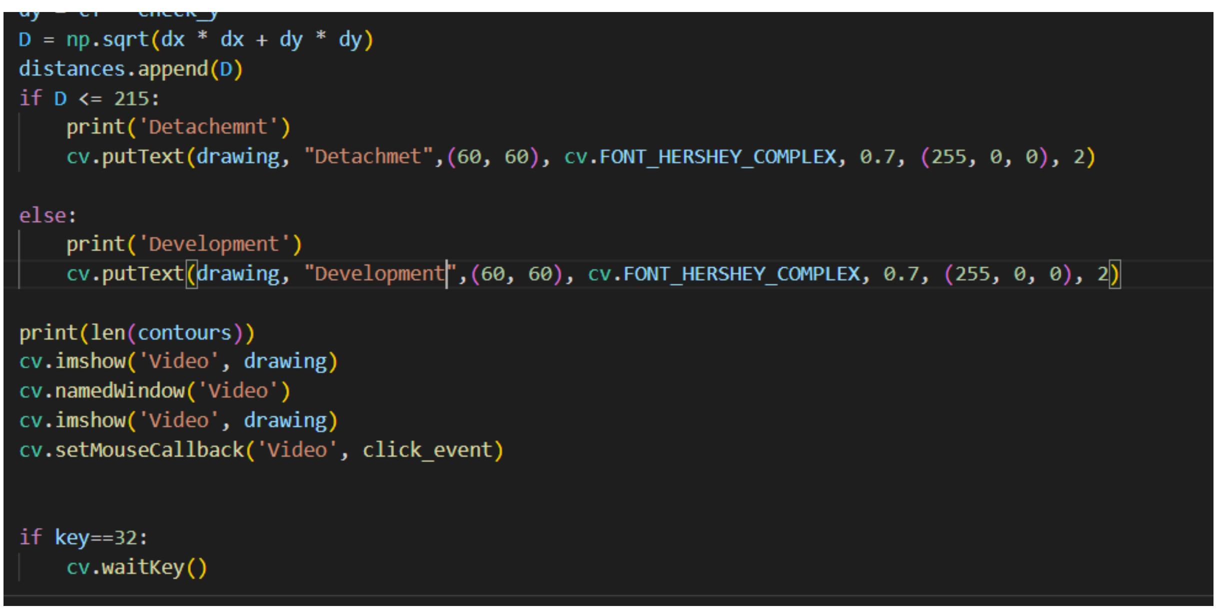


```python
(check_x, check_y) = (height,width//2)
M = cv.moments(contours[contours_length-1])
cX = int(M['m10'] /M['m00'])
cY = int(M['m01'] /M['m00'])
dx= cX - check_x
dy = cY - check_y
D = np.sqrt(dx * dx + dy * dy)
```

We can calculate the centroid using this calculation given by the relations, $C_x = M_{10}/M_{00}$ and $C_y = M_{01}/M_{00}$ where M00 is the area of the contour.

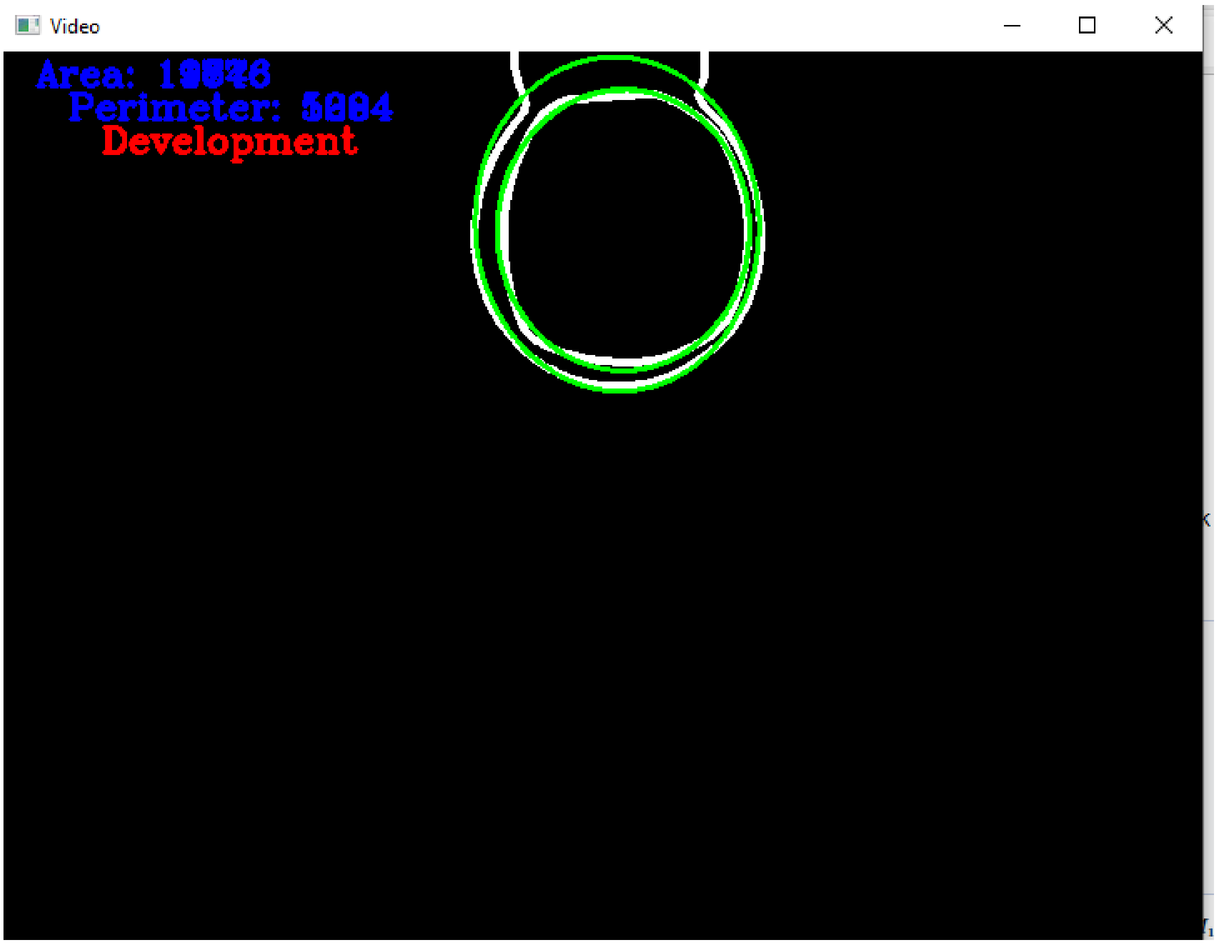
After that we have two points so we can easily get the distance between them mathematically by calculating the square root of the difference between X and Y coordinates of the two points, then we append those distances to a list called distances.

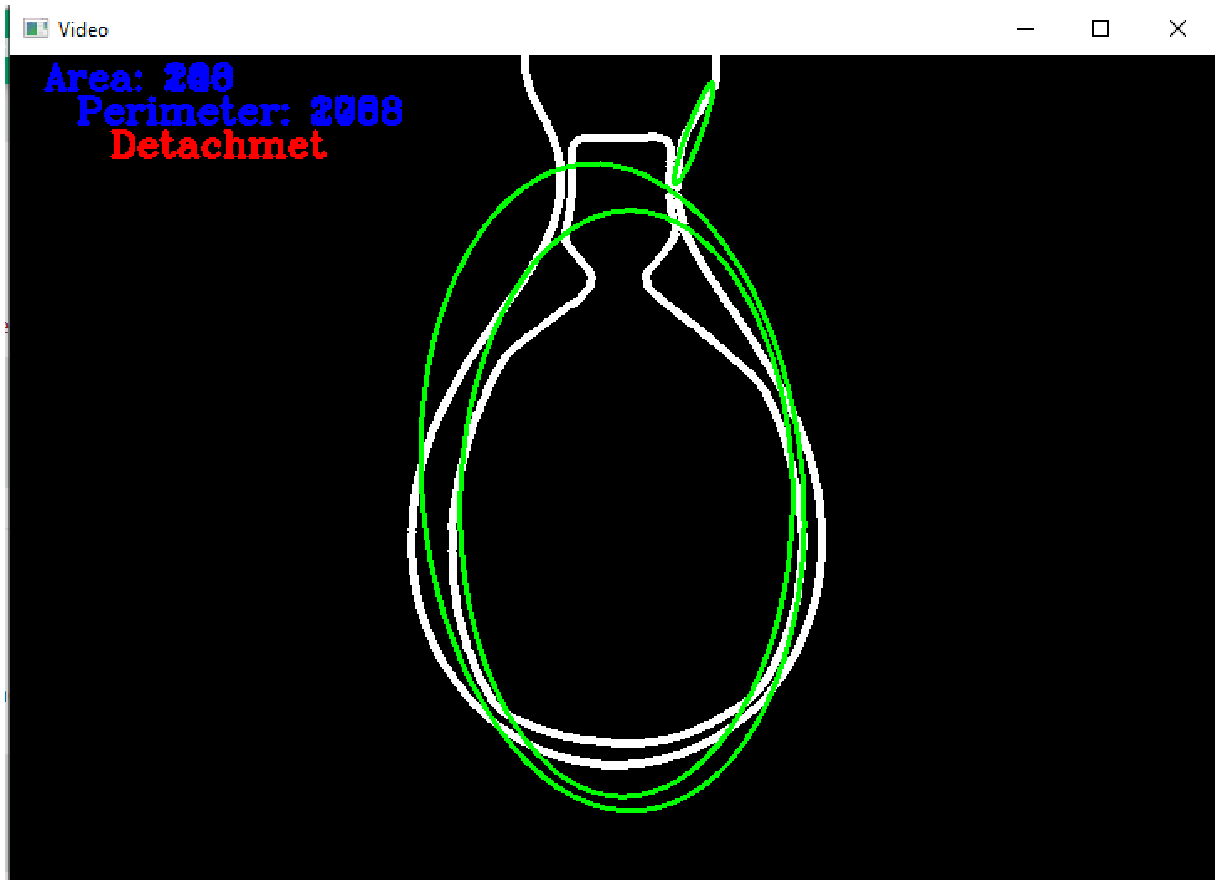
Then what I noticed is that when the distance is approximately 215 then the droplet is detached so I checked for this value to determine the state of the droplet.



```python
D = np.sqrt(dx * dx + dy * dy)
distances.append(D)
if D <= 215:
    print('Detachemnt')
    cv.putText(drawing, "Detachmet",(60, 60), cv.FONT_HERSHEY_COMPLEX, 0.7, (255, 0, 0), 2)

else:
    print('Development')
    cv.putText(drawing, "Development",(60, 60), cv.FONT_HERSHEY_COMPLEX, 0.7, (255, 0, 0), 2)

print(len(contours))
cv.imshow('Video', drawing)
cv.namedWindow('Video')
cv.imshow('Video', drawing)
cv.setMouseCallback('Video', click_event)


if key==32:
    cv.waitKey()
```

And this is the output of the application





Things to be improved

I think there could be a more general and precise way for calculating the information but this what I reached out

Also there could be a more precise way to determine the state of the droplets I tried it with distances calculation but there might be a better way.