## Algoritmo de detección de bloques

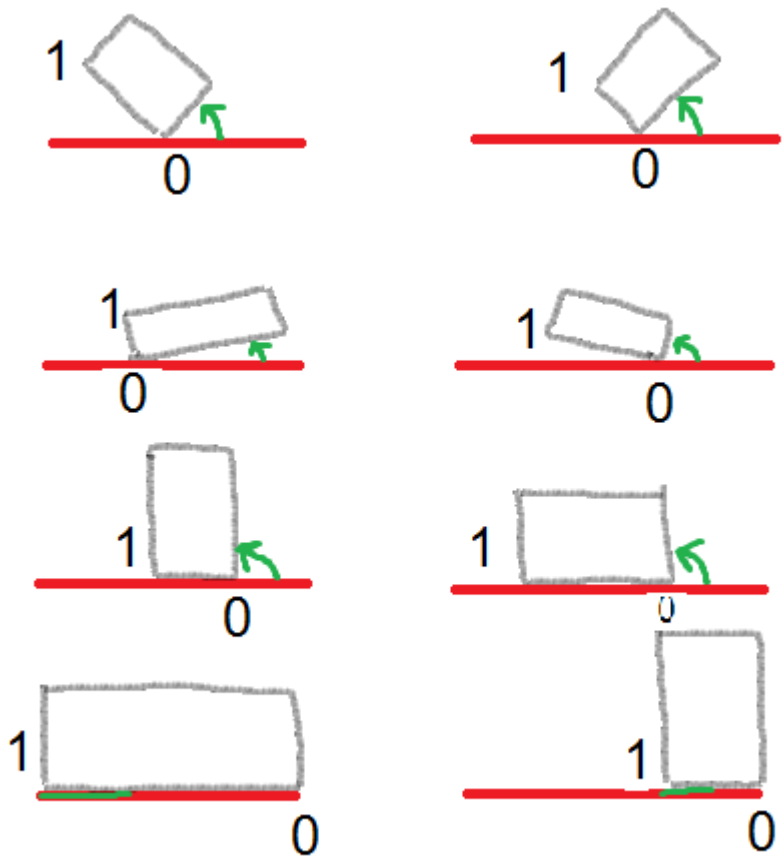
En primer lugar se le realizará un preprocesamiento a la imagen con el objetivo de destacar los bloques lo mayor posible. Este consta de 5 partes:

- filtro de mediana: eliminaar cualquier ruido presente
- Pasaje de imagen a escala de grises: de BGR a grises
- Binarización global por el método de otsu: en este caso los bloques destacan bastante del fondo, tanto su color como sus bordes, por lo que es favorable aplicar esta binarización
- Apertura: regularizar las formas
- Cierre: eliminar los contornos pequeños

A continuación, nos valemos de la función $findContours$ para hallar todos los contorno presentes en la imagen. Luego se le aplica a cada contorno la función *minAreaRect*, que devuelve el mínimo rectangulo orientado que contiene al contorno. De esta manera, se obtiene un parámetro al cual es mucho más facil aplicarle condiciones para corroborar si se trata del bloque buscado, además esta función devuelve entre otras propiedades, el centro del rectangulo y el ancho y largo.

Para entender la orientacion, y como la maneja la funcion *minAreaRect* hay que tener en cuenta los siguientes datos:

- el vértice más bajo es el vértice 0, luego el 1,2 y 3 le siguen en forma horaria
- La funcion calcula el angulo midiendo a contrareloj desde la horizontal hasta el primer borde que encuentre del rectangulo
- El ángulo varia de 0 a -90




[46]:
```
#Si queremos que las imágenes sean mostradas en una ventana emergente quitar el␣
 ↪inline
%matplotlib inline

# OpenCV-Python utiliza NumPy para el manejo de imágenes
import numpy as np
# cv2 es el módulo python para acceder a OpenCV
import cv2 as cv
# Usamos las poderosas herramientas de graficación de matplotlib para mostrar␣
 ↪imágenes, perfiles, histogramas, etc
import matplotlib.pyplot as plt
# para acceder a la ruta de las imagenes
import glob
```

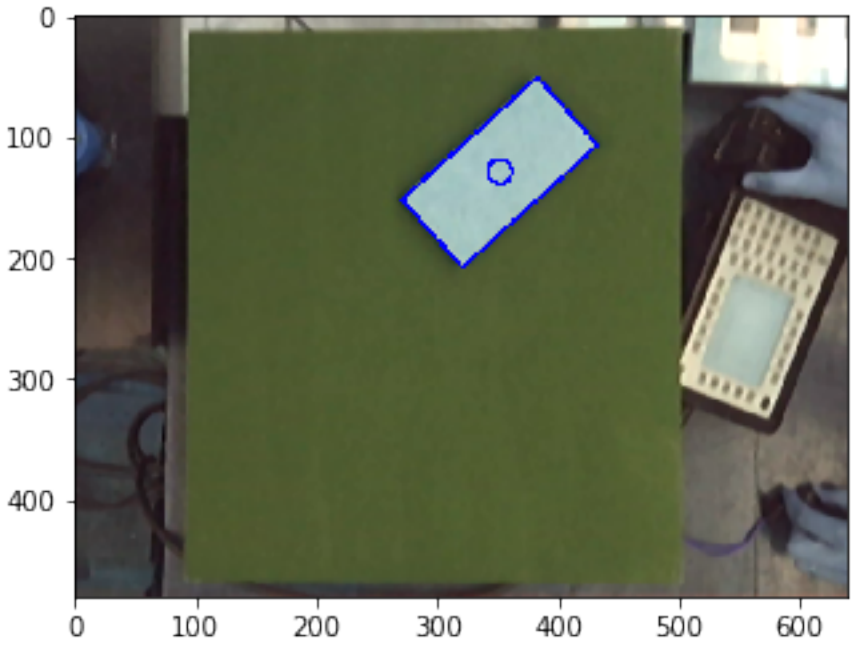
[49]:
```
# Carpeta con las fotos:
calib_fnames = glob.glob('C:/Users/pichichus/Desktop/Facultad/FMVR/TPFINAL/
 ↪imagenes/img_bloques/*.png')
#calib_fnames = glob.glob('C:/Users/pichichus/Desktop/Facultad/FMVR/TPFINAL/
 ↪imagenes/img_bloques_desafio/*.jpg')
```

[50]:
```
centros = list()
vertices=list()
orientacion = list()
i=1
for img in calib_fnames:
    print('Imagen{}'.format(i))
    #Leo la imagen
    img = cv.imread(img)

    #ROI
    #img= img[15:600,80:500,:]

    #aplico filtro suaviza
    img = cv.medianBlur(img,5)

    #paso a gris
    imggray = cv.cvtColor(img, cv.COLOR_BGR2GRAY)

    #binarizacion global, otsu (el contraste entre elementos es alto)
    ret, img_bin = cv.threshold(imggray,30,255,cv.THRESH_BINARY+cv.THRESH_OTSU)




    #regularizar las formas
    #apertura

    kernel = np.ones ((1,1), np.uint8)
    apertura = cv.morphologyEx(img_bin, cv.MORPH_OPEN, kernel)


    #cierre
    kernel = np.ones ((2,2), np.uint8)
    cierre = cv.morphologyEx(apertura, cv.MORPH_CLOSE, kernel)


    #hallo los contornos
    im2, contours, hierarchy = cv.findContours(cierre, cv.RETR_TREE, cv.
↪CHAIN_APPROX_TC89_L1)


    #recorro contornos hallados y verifico si corresponden sus caracteristicas a␣
↪las que se esperarían de un rectangulo
    for cnt in contours:
        rect = cv.minAreaRect(cnt)
        centro = rect[0]
        dimension=rect[1]
```

```
        rotacion = rect[2]
        area = cv.contourArea(cnt)


        if((dimension[0]*dimension[1] != 0)  and (area/
↪(dimension[0]*dimension[1])>0.95) and area>10000):
            box = cv.boxPoints(rect)
            #d(p1,p2)=sqrt((x2-x1)**2+(y2-y1)**2)
            x0,y0= box[0]
            x1,y1=box[1]
            x3,y3=box[3]
            if(np.sqrt((x3-x0)**2+(y3-y0)**2)<np.sqrt((x1-x0)**2+(y1-y0)**2)):
                rotacion = rotacion-90
            rotacion = abs(rotacion)
            print("Encontré bloque!")
            print('Área: {} - Centro: {} - Dimensiones: {} - Rotación: {}'.
↪format(area,centro,dimension,rotacion))
            cv.circle(img, (int(centro[0]),int(centro[1])), 10,(0,0,255),2)




            vertices.append(box)
            box = np.int0(box)

            cv.drawContours(img,[box],0,(0,0,255),2)
            centros.append(centro)
            dimensiones.append(dimension)
            orientacion.append(rotacion)
            plt.figure(i)
            plt.imshow(img)
            plt.show()







    i= i+1
%store dimensiones
%store centros
%store vertices
%store orientacion
```

Imagen1

```
Encontré bloque!
Área: 11120.0 - Centro: (352.4909973144531, 129.95880126953125) - Dimensiones:
(150.36459350585938, 75.43861389160156) - Rotación: 42.031761169433594
```




```
Imagen2
Encontré bloque!
Área: 11149.0 - Centro: (171.67794799804688, 145.92718505859375) - Dimensiones:
(75.98509979248047, 150.09153747558594) - Rotación: 114.37646865844727
```

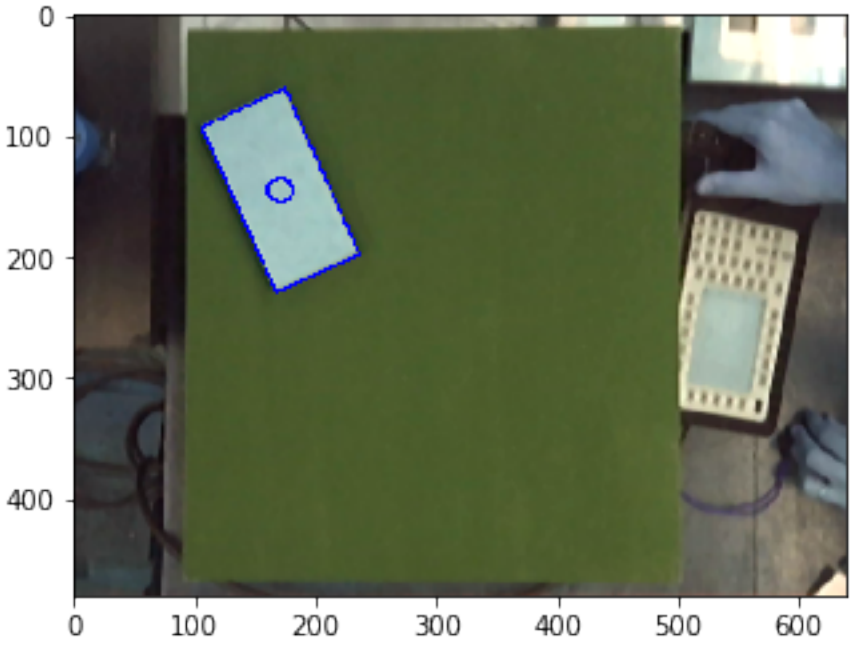



Imagen3
Encontré bloque!
Área: 11348.5 - Centro: (226.46409606933594, 130.07369995117188) - Dimensiones: (149.01434326171875, 77.37078857421875) - Rotación: 51.49934387207031

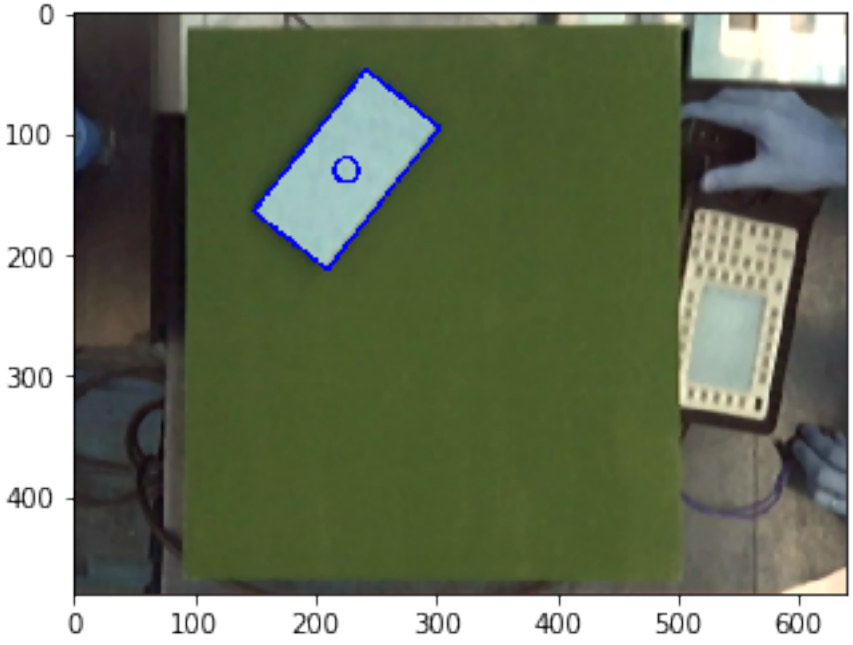


Imagen4
Encontré bloque!
Área: 11137.5 - Centro: (270.3270568847656, 130.83819580078125) - Dimensiones:
(75.28987884521484, 150.3024139404297) - Rotación: 101.0702018737793

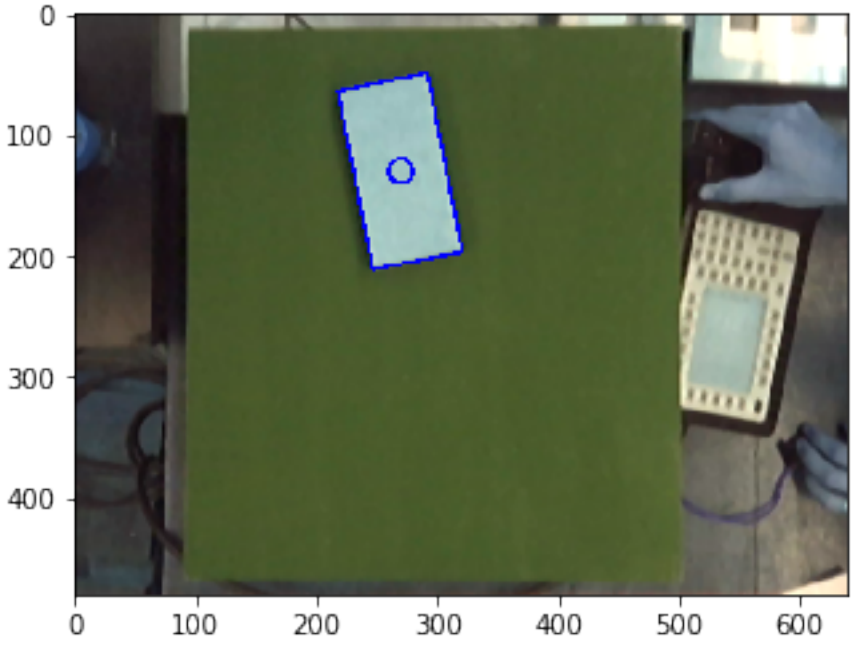



Imagen5
Encontré bloque!
Área: 11215.5 - Centro: (366.16229248046875, 126.82916259765625) - Dimensiones:
(151.77186584472656, 75.65897369384766) - Rotación: 73.03385162353516

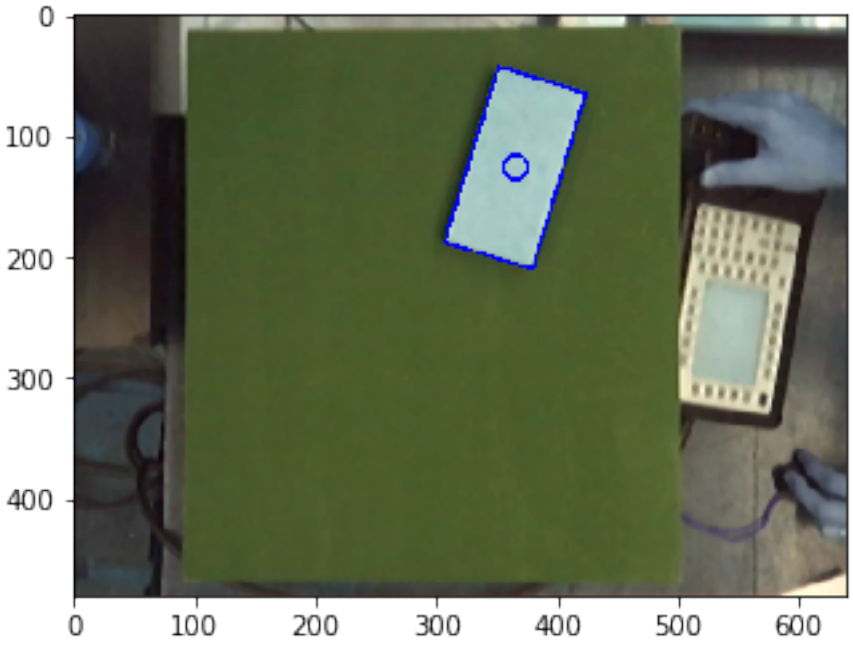



Imagen6
Encontré bloque!
Área: 11065.0 - Centro: (305.78240966796875, 226.9926300048828) - Dimensiones: (75.6281509399414, 149.31309509277344) - Rotación: 147.80426788330078

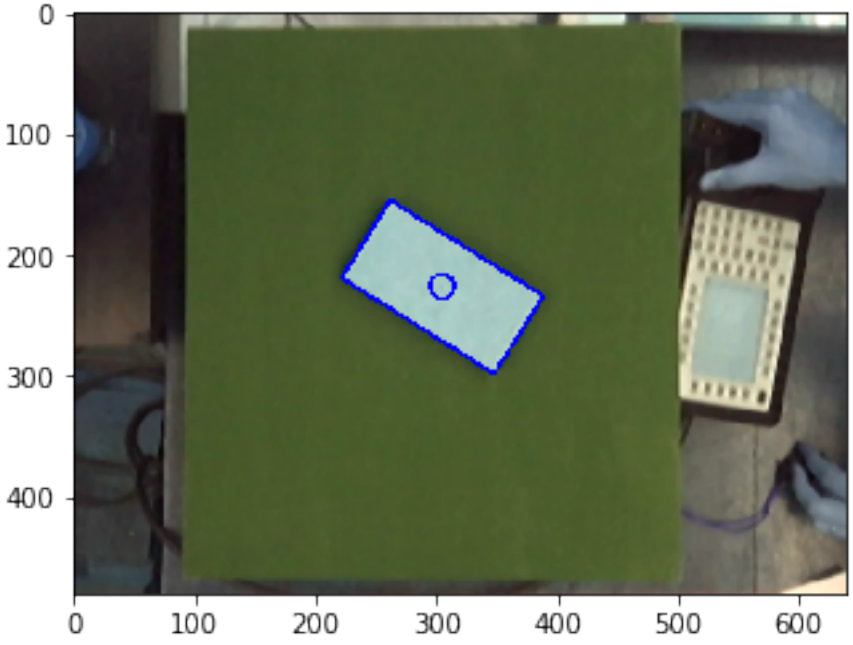


Imagen7
Encontré bloque!
Área: 11237.5 - Centro: (375.8201904296875, 319.88763427734375) - Dimensiones:
(76.31983947753906, 150.73167419433594) - Rotación: 147.9946174621582

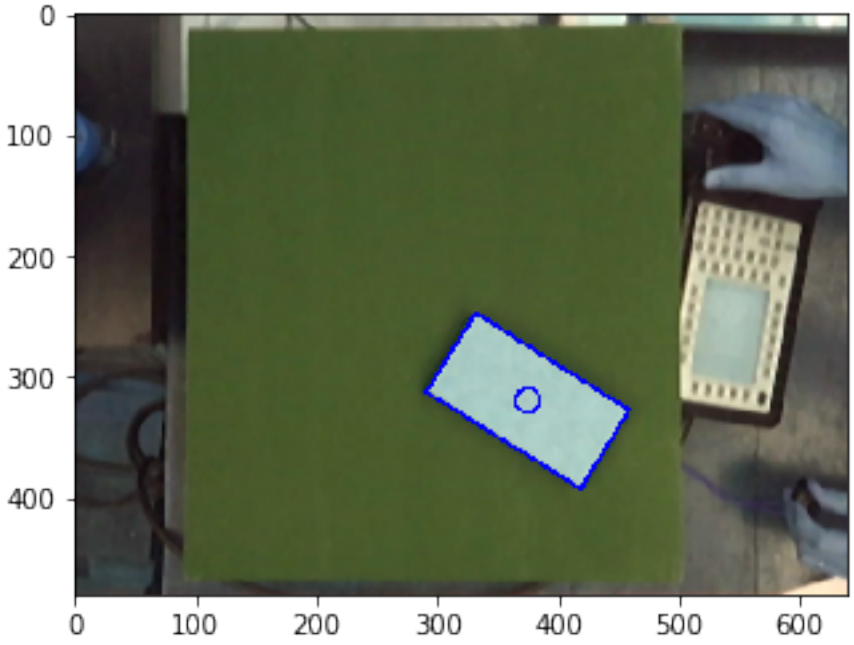



Imagen8
Encontré bloque!
Área: 11016.5 - Centro: (299.1384582519531, 157.4621124267578) - Dimensiones:
(75.22718811035156, 149.06796264648438) - Rotación: 131.87786865234375

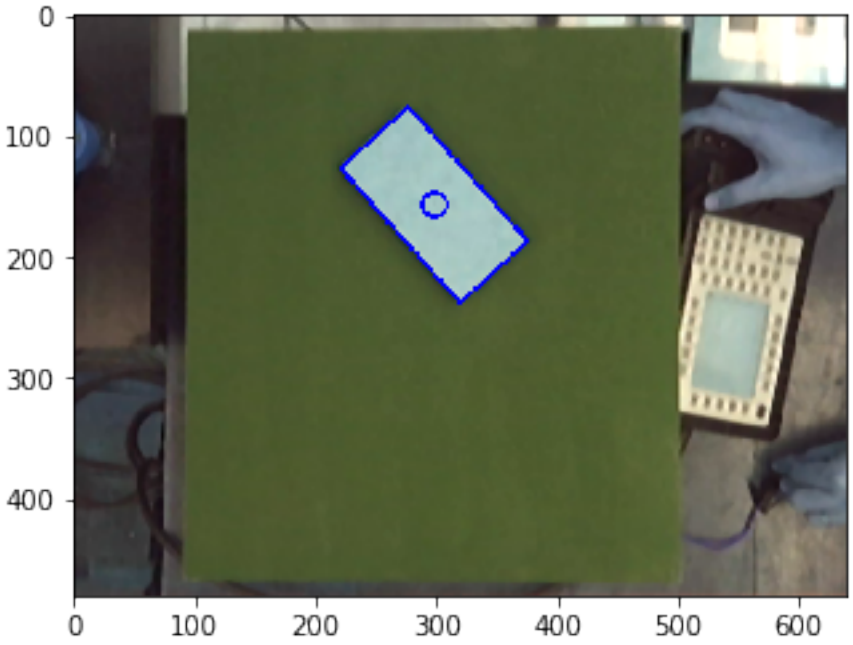
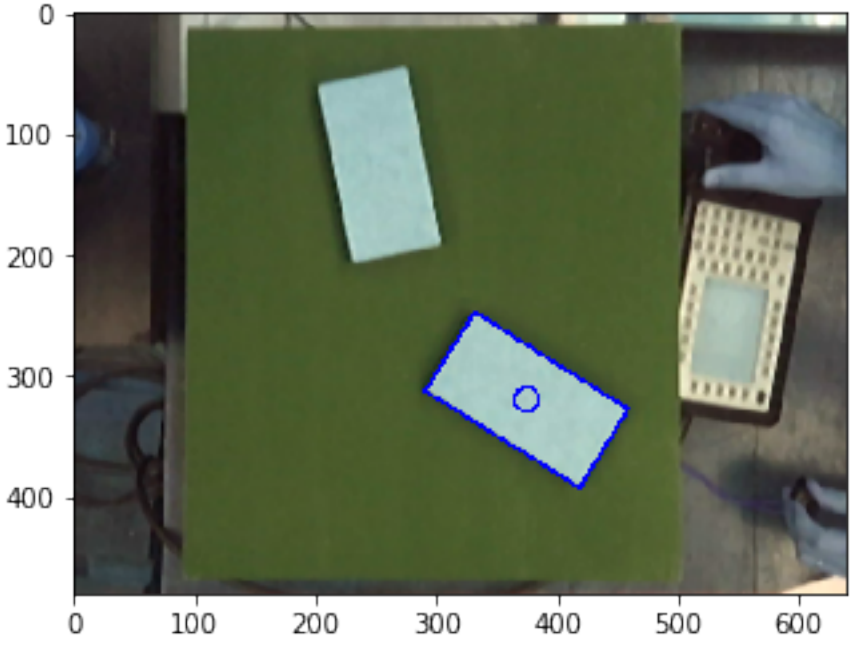



Imagen9
Encontré bloque!
Área: 11210.0 - Centro: (375.82025146484375, 319.8876953125) - Dimensiones: (76.3198471069336, 150.73171997070312) - Rotación: 147.9946174621582



Encontré bloque!
Área: 11148.0 - Centro: (254.31228637695312, 126.7426986694336) - Dimensiones: (75.33026123046875, 150.46812438964844) - Rotación: 101.09372329711914

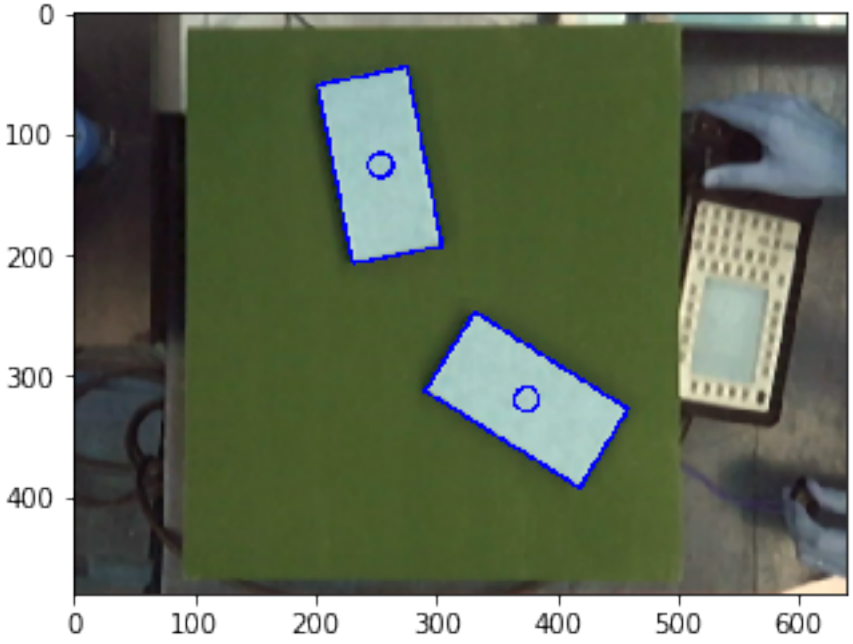
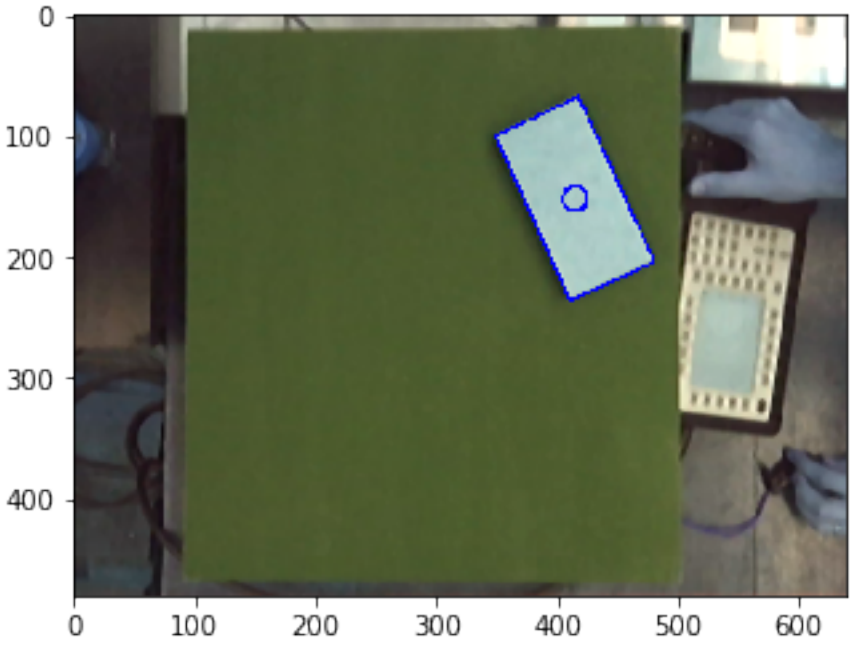



Imagen10
Encontré bloque!
Área: 11206.5 - Centro: (415.2177429199219, 152.42982482910156) - Dimensiones: (76.53124237060547, 149.95542907714844) - Rotación: 114.70243072509766




```
Imagen11
Encontré bloque!
Área: 11375.5 - Centro: (179.4128875732422, 362.5978698730469) - Dimensiones:
(78.24755096435547, 149.4651641845703) - Rotación: 113.31770896911621
```

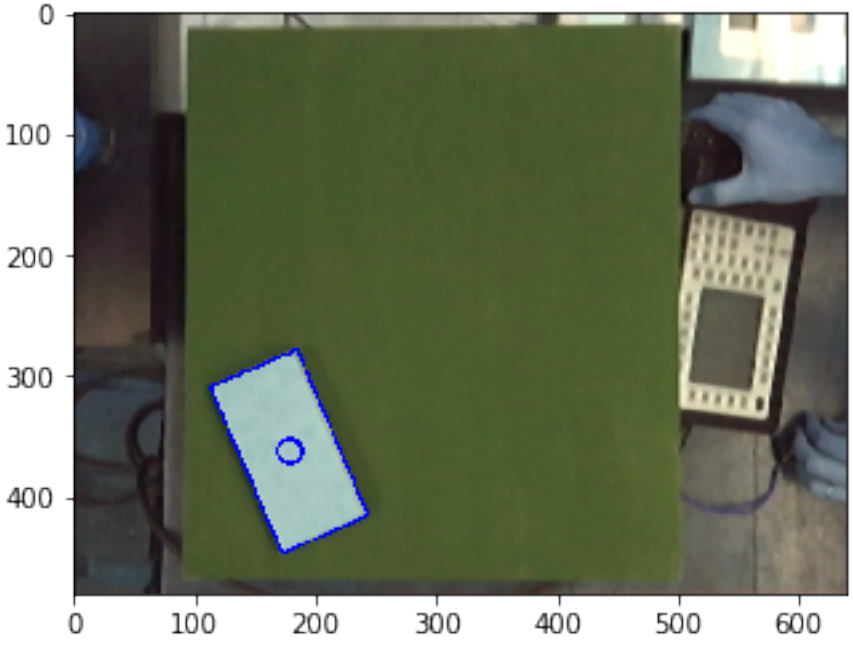


Imagen12
Encontré bloque!
Área: 11139.5 - Centro: (300.0484619140625, 356.8919677734375) - Dimensiones:
(150.10269165039062, 75.60794067382812) - Rotación: 43.68308639526367

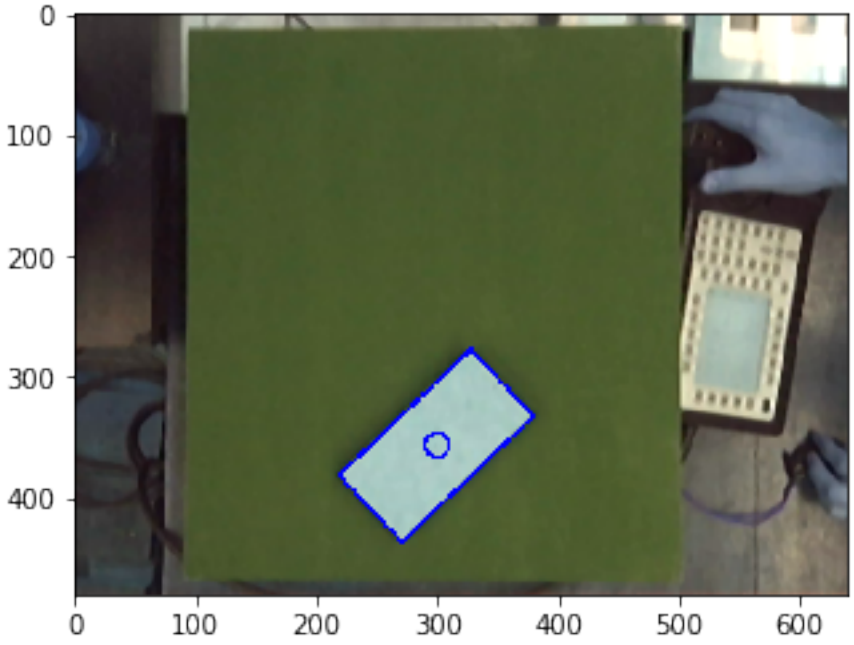



Imagen13
Encontré bloque!
Área: 11304.0 - Centro: (409.53369140625, 359.77203369140625) - Dimensiones:
(76.01253509521484, 152.02508544921875) - Rotación: 120.25644111633301

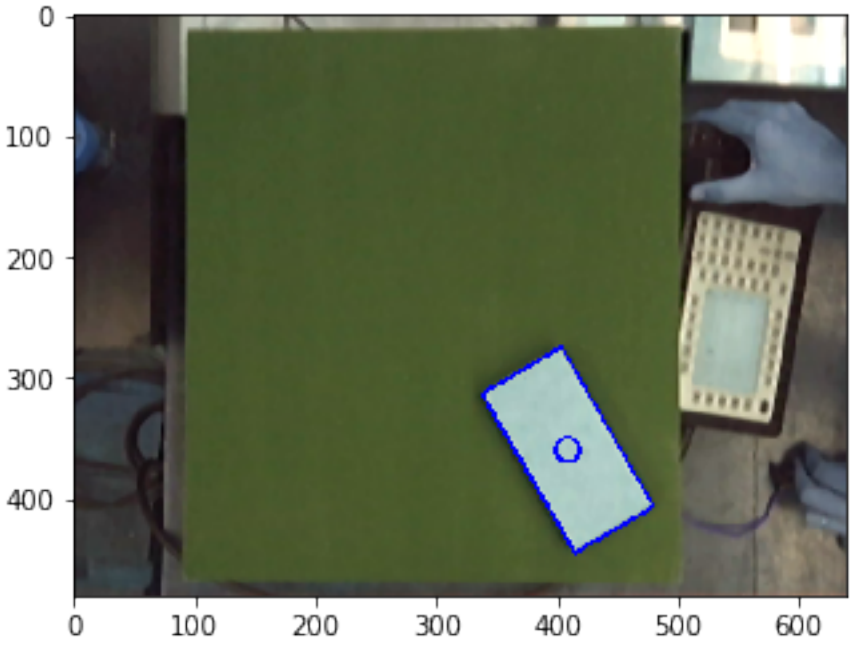



Imagen14
Encontré bloque!
Área: 11131.0 - Centro: (393.03448486328125, 272.4137878417969) - Dimensiones: (150.04183959960938, 75.7636947631836) - Rotación: 68.19859313964844

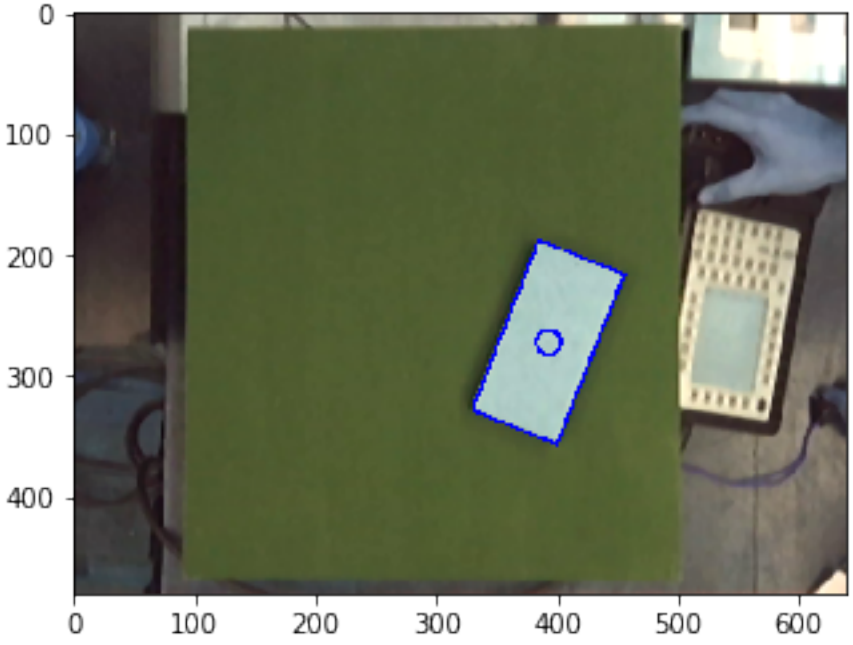


Imagen15
Encontré bloque!
Área: 11035.5 - Centro: (280.3117980957031, 267.0211181640625) - Dimensiones:
(75.49942779541016, 149.2349395751953) - Rotación: 110.43282890319824

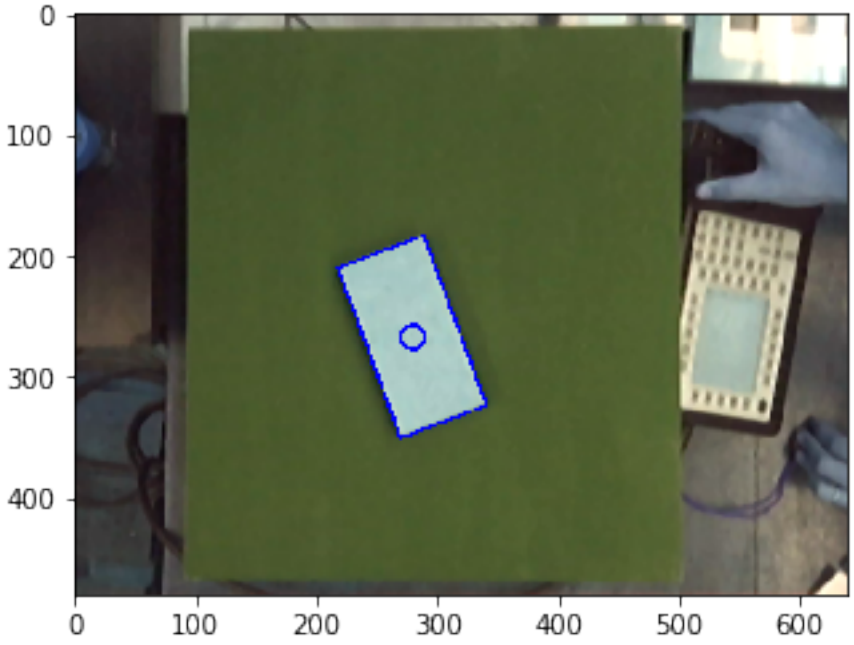
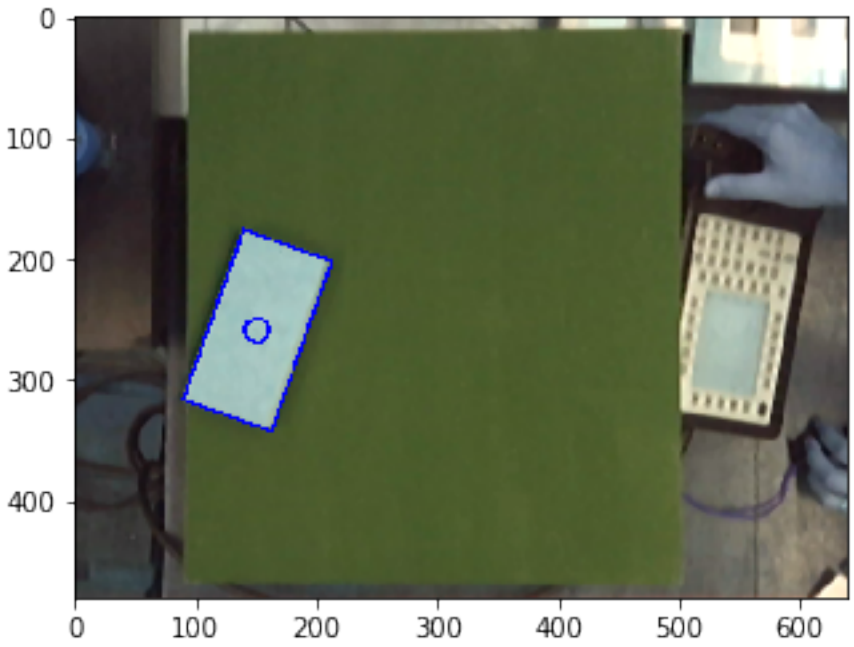



Imagen16
Encontré bloque!
Área: 11271.5 - Centro: (151.64498901367188, 259.4842834472656) - Dimensiones:
(148.94139099121094, 77.56439971923828) - Rotación: 70.41600799560547




```
Stored 'dimensiones' (list)
Stored 'centros' (list)
Stored 'vertices' (list)
Stored 'orientacion' (list)
```

[ ]:

[ ]:

[ ]: