


```
_, imagen = captura.read()
hsv = cv2.cvtColor(imagen, cv2.COLOR_BGR2HSV)
verde_bajos = np.array([0, 115, 121], dtype=np.uint8)
verde_altos = np.array([115, 255, 255], dtype=np.uint8)
mask = cv2.inRange(hsv, verde_bajos, verde_altos)
img = cv2.morphologyEx(mask, cv2.MORPH_CLOSE, kernel)
circles = cv2.HoughCircles(img, cv2.HOUGH_GRADIENT, 1, 200,
                           param1=30, param2=45, minRadius=0, maxRadius=0)
```

Imagen 4

Analiza cada frame donde está la pelota para actualizar la posición y detectar si se ha hecho un salto.

```
if circles is not None:
    circles = np.round(circles[0, :]).astype("int")
    for (x, y, r) in circles:
        cv2.circle(imagen, (x, y), r, (0, 255, 0), 4)
        cv2.rectangle(imagen, (x - 5, y - 5), (x + 5, y + 5), (0, 128, 255), -1)
        b.update(x, y)
```

Imagen 5

Finalmente va mostrando en pantalla la cantidad de saltos que va realizando la pelota

```
font = cv2.FONT_HERSHEY_SIMPLEX
cv2.putText(imagen, str(b.jumps), (10, 400), font,
            4, (255, 255, 255), 2, cv2.LINE_AA)
cv2.imshow('img', img)
cv2.imshow('Camara', imagen)
```

Imagen 6

2. Library.py

Este archivo contiene la clase ball, la cual se encarga de identificar la pelota mediante métodos como getx y gety para tener la coordenada de la pelota y el método update para ir actualizando la posición y poder realizar bien el conteo de los saltos en el momento indicado.

En la siguiente imagen se observa la codificación de este archivo.

```
library.py
import cv2
import numpy as np
import sys

class ball():
    def __init__(self, x, y):
        self.x = x
        self.y = y
        self.dir = "down"
        self.jumps = 0

    def getX(self):
        return self.x

    def getY(self):
        return self.y

    def getDir(self):
        return self.dir

    def getJumps(self):
        return self.jumps

    def setPos(self, x, y):
        self.x = x
        self.y = y

    def update(self, x, y):
        if(self.y < y and self.dir == "down"):
            self.jumps += 1
            self.dir = "up"
        if(self.y > y):
            self.dir = "down"
        self.setPos(x, y)
```

Imagen 7

III. CONCLUSIONES

- Se aprendió mucho a la hora de trabajar con la librería OpenCV ya que tiene una infraestructura de visión por computador muy completa.
- Fue complicado al principio hacer que el software identificara correctamente la pelota de color verde pero después de una ardua investigación se logró solucionar el error.

RECOMENDACIONES

La visión artificial o visión computador es un campo muy amplio y muy interesante de la inteligencia artificial el cual se debe seguir investigando para desarrollar tecnología de última generación.

REFERENCIAS

<http://opencv.org/>

<http://www.numpy.org/>

<http://sourceforge.net/projects/opencvlibrary/>

<https://pypi.python.org/pypi/opencv-python>