Comptage de point automatisé au basketball

Hugo Berthod
Juin 2023

Sommaire

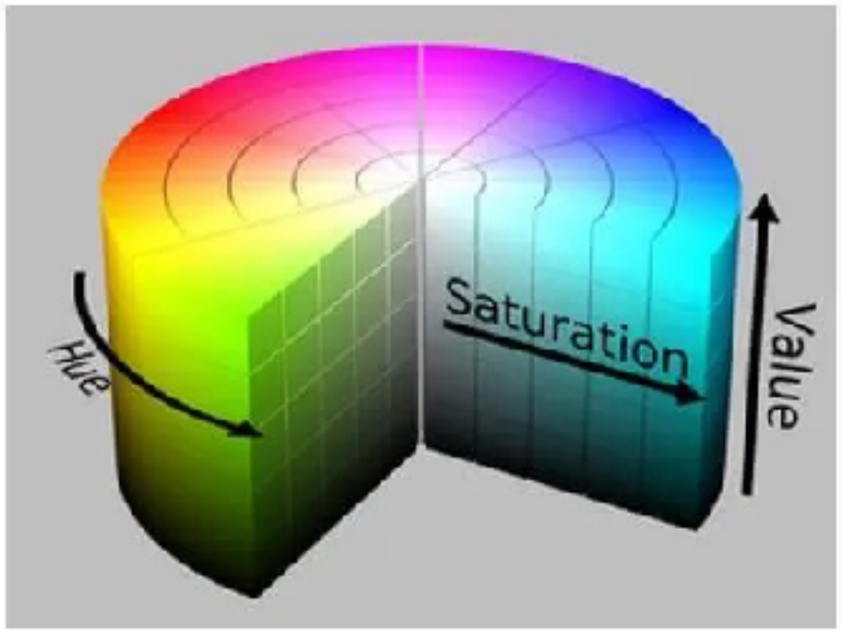
Présentation du problème

bebasket.fr

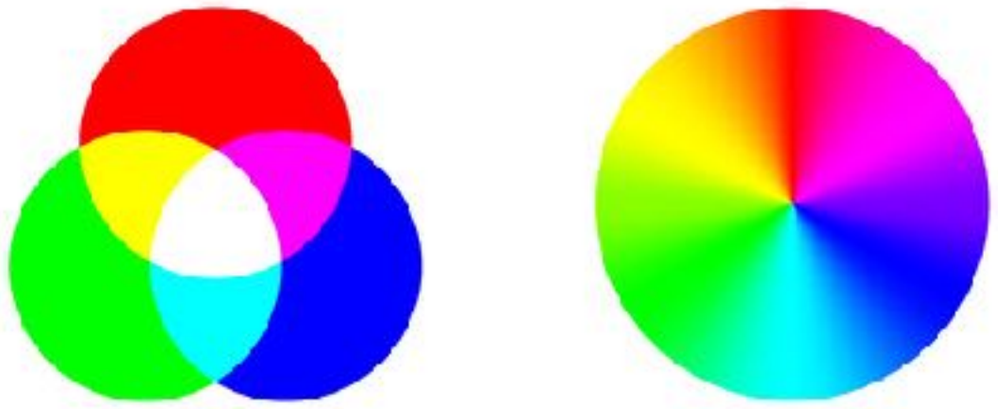
bodet-sport.com

Suivi de la balle

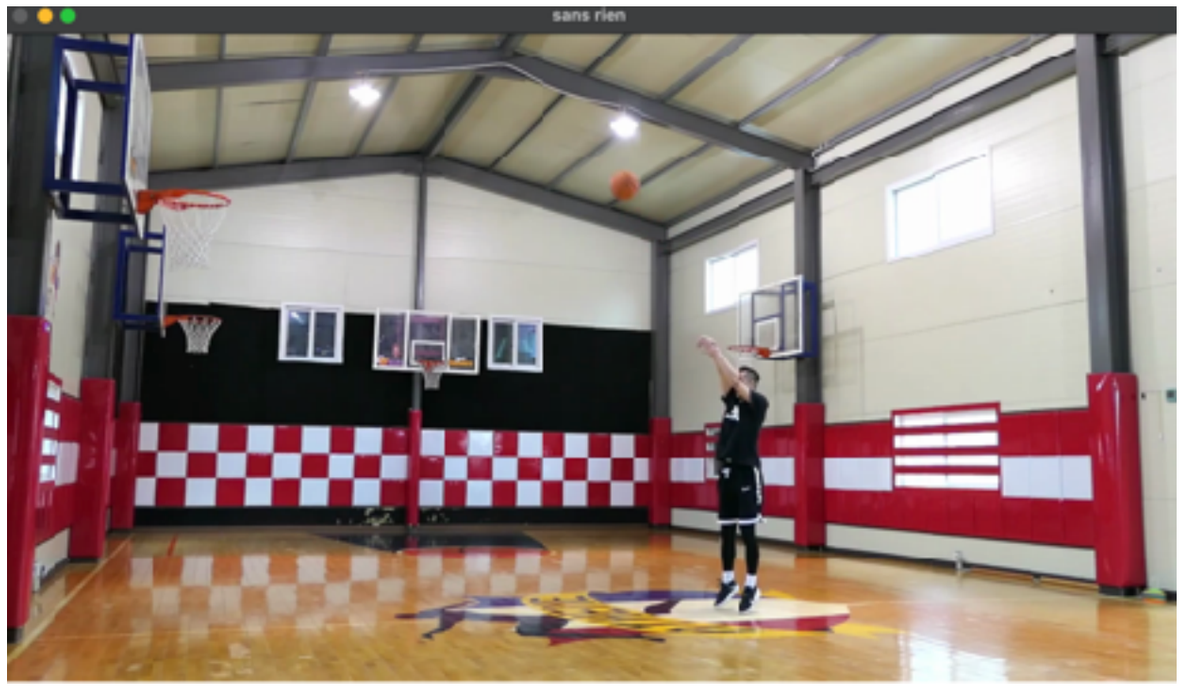
HSV : Hue Saturation Value
(Teinte saturation valeur)



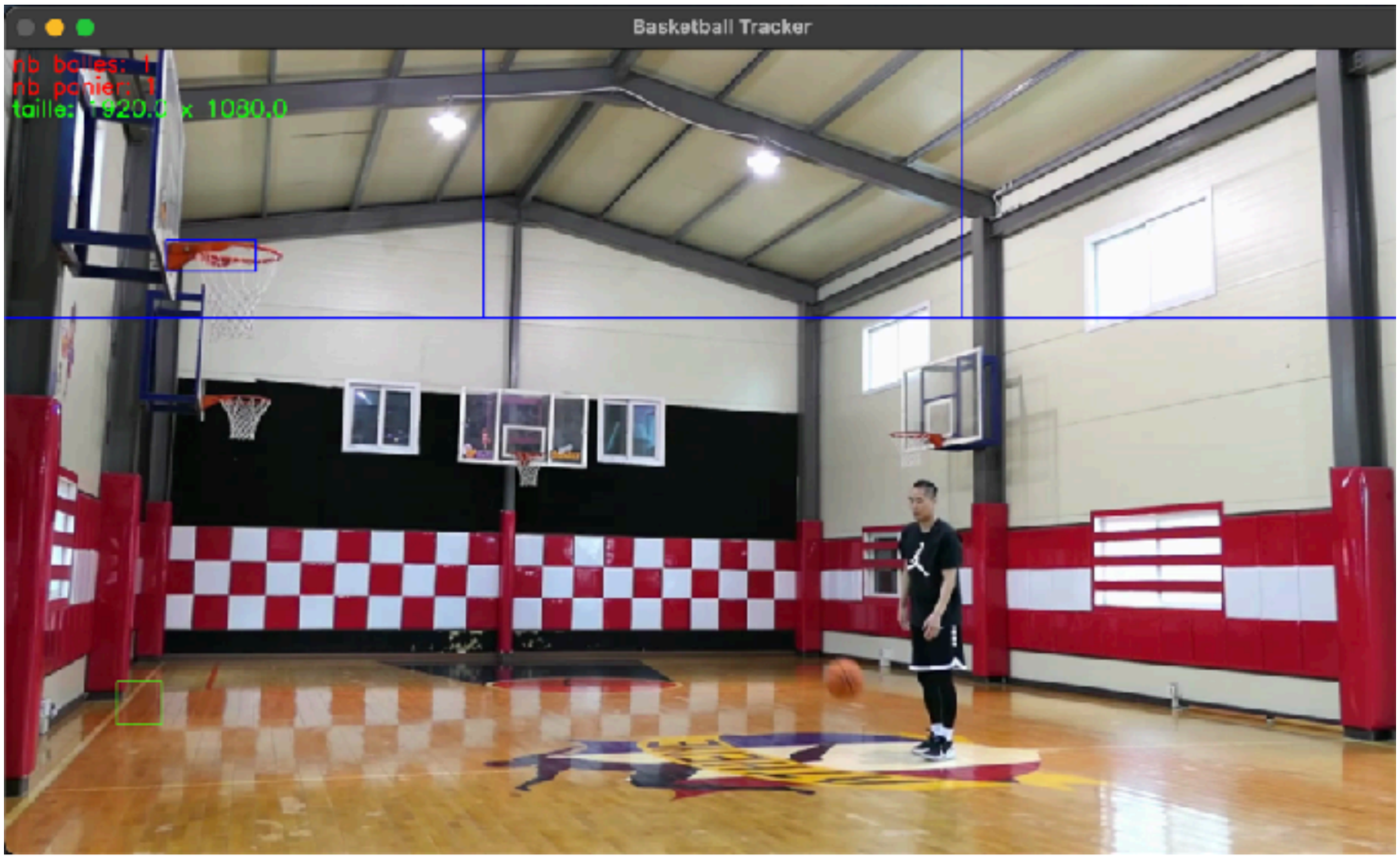
RVB : Rouge Vert Bleu



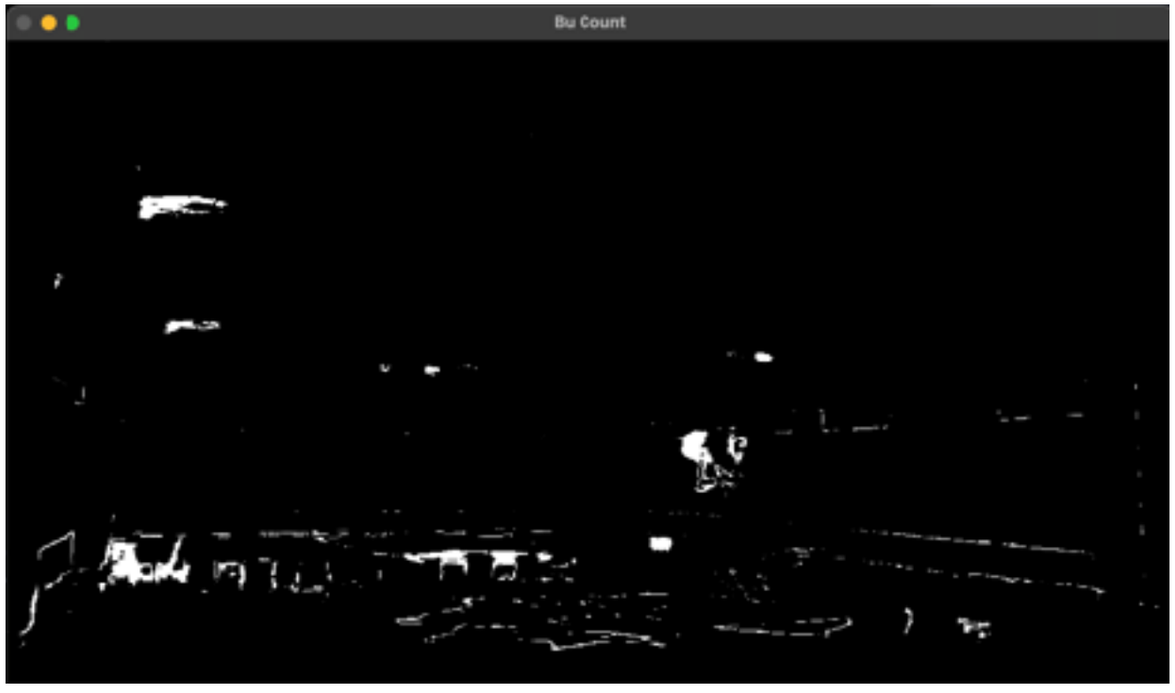
Suivi de la balle

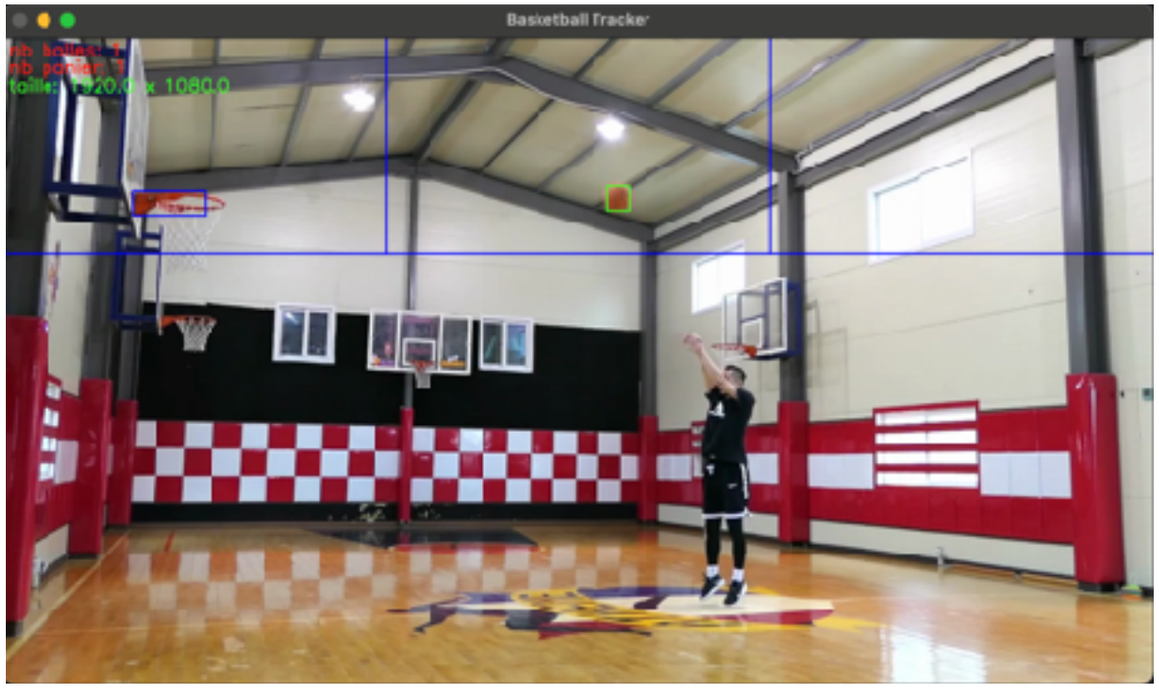


Suivi de la balle



Détection des paniers





Algorithme

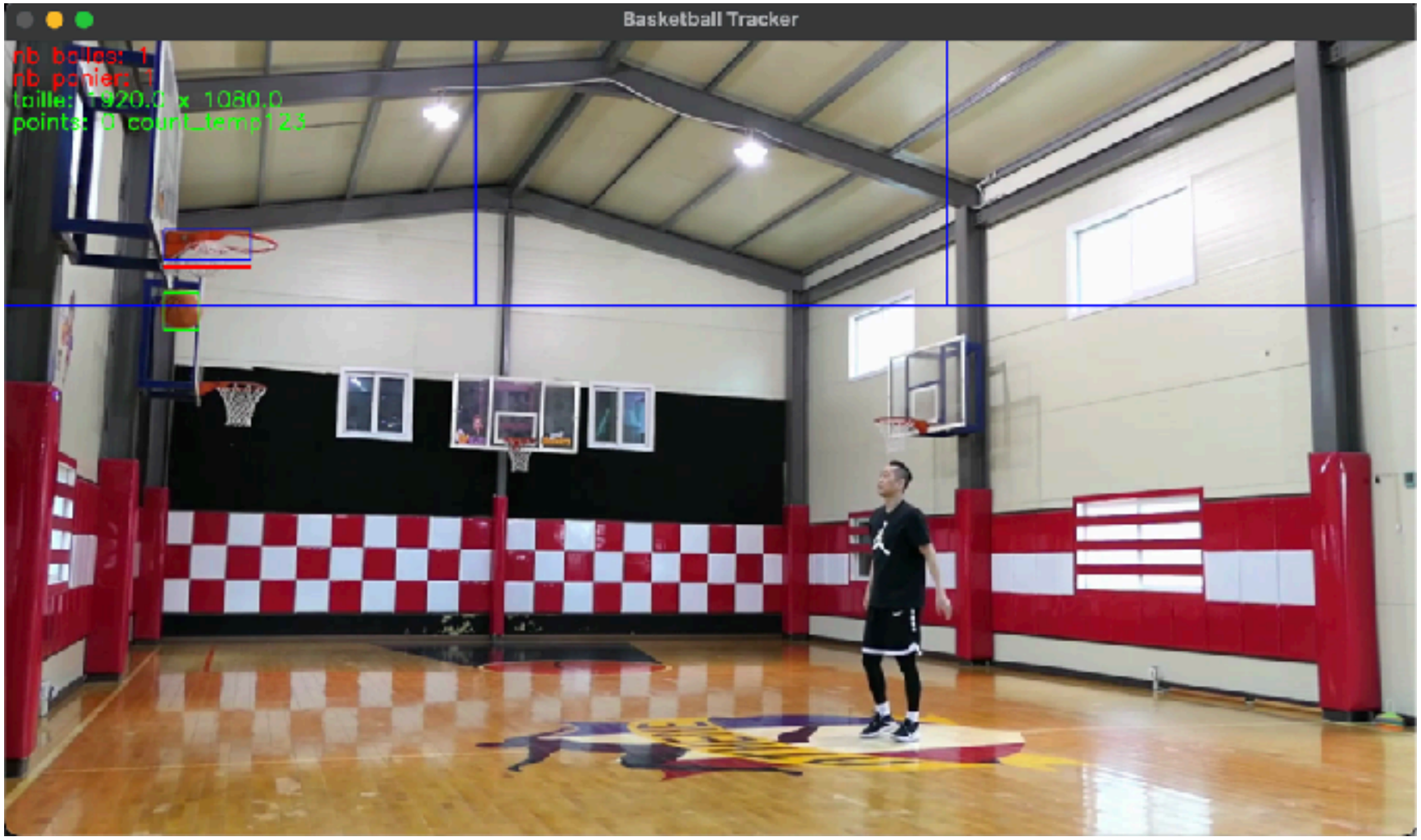
- Valeurs de teinte pour la balle
- Valeurs de teinte pour le panier
- Cap = récupération du flux vidéo
- Tant que la vidéo a encore des images :
    - HSV = conversion de cap de RGB à HSV
    - On applique le filtre avec les valeurs pour les balles et le paniers
    - On vérifie la taille des balles :
        - On dessine un carré autour
        - Si la balle est dans la zone rouge sous le panier
            - On ajoute 2 points au score
    - On vérifie la taille des paniers :
        - On vérifie la position des paniers
        - On dessine un rectangle autour
    - On affiche le nombre de balles et de paniers
    - On affiche l’image avec les contours

Algorithme

- Améliorations possibles :
  - Utilisation d’un filtre de Kalman
  - Détection de la balle grâce un filtre de mouvement
  - Utiliser le centre de la balle

- Chose encore à implémenter :
  - Affiner les valeurs pour la balle et le panier
  - Détecter quelle équipe marque

Réponse à la problématique



Annexe

```python
#importation des librairies
import cv2
import numpy as np

#Définition des valeurs maximale et minimale prise par le filtre HSV pour la balle
lower_ball = np.array([5, 120, 70])
upper_ball = np.array([10, 255, 255])

#Définition des valeurs maximale et minimale prise par le filtre HSV pour les paniers
lower_bu = np.array([0, 50, 50])
upper_bu = np.array([0, 100, 100])

# Initialisation des variables permettant de savoir le nombre de balles detecté
prev_ball_count = 0
ball_count = 0

prev_bu_count = 0
bu_count = 0

# initialisation du compteur pour les points
point_count = 0

#Initialisation des variables servant à compartimenter le flux vidéo
width = 0
height = 0

mid_w = 0
mid_h = 0

bu_pos_1 = np.array([]) # initialisation des tableaux pour la position
bu_pos_2 = np.array([])

count_temp = 0

def scored(x,y,x2,y2,x3,y3):
    if x <x3 and x3 <x2 and y< y2 and y2< y3 :
        return True
    else :
        return False



while True:
    # Capture frame from the video
    isclosed = 0
    point_count = 0

    # Initialisation du flux vidéo, pour pouvoir le réutiliser facilement
    cap = cv2.VideoCapture('/Users/hugo/Documents/Cours/Prepa/TIPE/TIPE_Basketball/script/IA_assistef/01.mp4')
    #Récuperation de la hauteur, de la largeur et de leurs moitiés
```

Annexe

```python
    #Récuperation de la hauteur, de la largeur et de leurs moitiés
    height = cap.get(cv2.CAP_PROP_FRAME_HEIGHT)
    width = cap.get(cv2.CAP_PROP_FRAME_WIDTH)

    mid_w = round((width/3))
    mid_h = round((height/3))

    kernel = np.ones((5,5), np.uint8)


    fst_ret,fst_frame = cap.read()

    fst_hsv = cv2.cvtColor(fst_frame, cv2.COLOR_BGR2HSV)

    bu_mask = cv2.inRange(fst_hsv, lower_bu, upper_ball)

    opening_bu = cv2.morphologyEx(bu_mask, cv2.MORPH_OPEN, kernel)
    closing_bu = cv2.morphologyEx(opening_bu, cv2.MORPH_CLOSE, kernel)

    contours_bu , hierarchy_bu = cv2.findContours(closing_bu,cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)

    print("hauteur : ", height, "largeur : ",width)
    print("1/2 hauteur : ", mid_h, "1/2 largeur : ",mid_w)
    while True:

        ret, frame = cap.read()
        rat , vide = cap.read()
        if not ret or not rat:
            isclosed
            break

        # Convertion des images de RGB à HSV
        hsv = cv2.cvtColor(frame, cv2.COLOR_BGR2HSV)

        # Création du masque pour isolé la balle et les paniers
        ball_mask = cv2.inRange(hsv, lower_ball, upper_ball)

        # Apply morphological transformations to the mask
        opening_ball = cv2.morphologyEx(ball_mask, cv2.MORPH_OPEN, kernel)
        closing_ball = cv2.morphologyEx(opening_ball, cv2.MORPH_CLOSE, kernel)

        # Find contours of the basketball
        contours_ball, hierarchy_ball = cv2.findContours(closing_ball, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
        contours_bu , hierarchy_bu = cv2.findContours(closing_bu,cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_SIMPLE)
        # Draw bounding boxes around the basketball and count them
        ball_count = 0
        bu_count = 0

        c_line = (255,0,0)
        tick_line = 2
        cv2.line(img=frame,pt1=(0,mid_h), pt2=(1919,mid_h), color=c_line, thickness= tick_line)
```

Annexe

```
        cv2.line(img=frame,pt1=(0,mid_h), pt2=(1919,mid_h), color=c_line, thickness= tick_line)
        cv2.line(img=frame, pt1=(mid_w,0), pt2=(mid_w,mid_h),color=c_line, thickness= tick_line)
        cv2.line(img=frame, pt1=(2*mid_w,0), pt2=(2*mid_w,mid_h),color=c_line, thickness= tick_line)
        #Boucle du traitement du resultat du filtre pour les paniers , affichage en bleur
        for countour in contours_bu :
            area_bu = cv2.contourArea(countour)
            xb,yb,wb,hb = cv2.boundingRect(countour)
            if area_bu > 1500 and  yb < mid_h and xb < mid_w :
                cv2.rectangle(frame, (xb,yb),(xb+wb,yb+hb), (255,0,0),2)
                y_area = yb + 50
                x_area = xb
                y_area_2 = y_area +2
                x_area_2 = xb + wb
                cv2.rectangle(frame, (x_area,y_area),(x_area_2,y_area_2),(0,0,255), 2 )
                bu_count += 1
            if area_bu > 1500 and  yb < mid_h and 2*mid_w<xb < width :
                cv2.rectangle(frame, (xb,yb),(xb+wb,yb+hb), (255,0,0),2)
                bu_count += 1

        if bu_count > 2 :
            print("Ya un problème qlq part : on détecte 2 paniers")
        #Boucle du traitement du resultat du filtre pour la balle, affichage en vert
        for element in contours_ball:
            area = cv2.contourArea(element)
            x, y, w, h = cv2.boundingRect(element)
            dif = abs(w -h)
            if area > 900 and dif < 10: #permet de s'assurer que les petites taches ne sont pas prises en compte et que le contour est proche d'un carré
                x, y, w, h = cv2.boundingRect(element)
                cv2.rectangle(frame, (x,y), (x+w,y+h), (0,255,0), 2)
                ball_count += 1
                if scored(x_area,y_area,x_area_2,y_area_2,x,y) and count_temp > 10 :
                    point_count += 2
                    count_temp = 0
        count_temp+=1

        print("nb de points marqué : ", point_count)
        if ball_count > 1 :
            print("on a un problème quelque part, on detecte 2 balles")

        # Display the tracking result on the screen
        cv2.putText(frame, "nb balles: " + str(ball_count), (10, 30), cv2.FONT_HERSHEY_SIMPLEX, 1, (0, 0, 255), 2)
        cv2.putText(frame, "nb panier: " + str(bu_count), (10,60), cv2.FONT_HERSHEY_SIMPLEX, 1, (0,0,255),2)

        cv2.putText(frame, "taille: "+str(width)+" x "+str(height), (10, 90), cv2.FONT_HERSHEY_SIMPLEX,1,(0,255,0),2)
        cv2.putText(frame, "points: "+str(point_count) + " count_temp"+str(count_temp), (10, 120), cv2.FONT_HERSHEY_SIMPLEX,1,(0,255,0),2)

        cv2.imshow("Ball Count", ball_mask)
        cv2.imshow("Bu Count", bu_mask)
        cv2.imshow("Basketball Tracker", frame)

        #cv2.imshow("sans rien",vide)
```

Annexe

```
        #cv2.imshow("sans rien",vide)
        # Exit the program if the 'q' key is pressed
        if cv2.waitKey(1) & 0xFF == ord('q'):
            isclosed= 1
            break

        # Update the previous count
        prev_ball_count = ball_count
        prev_bu_count = bu_count
    if isclosed :
        break

# Release the video capture and close all windows
cap.release()
cv2.destroyAllWindows()
```