Ping Pong PDI

Procesamiento Digital de Imágenes
FACULTAD DE INGENIERÍA
UNIVERSIDAD DE ANTIOQUIA
Departamento de Ingeniería de Sistemas
Semestre 2021-2

Miguel Angel Urueña Riobo
miguel.uruena@udea.edu.co

Descripción del Problema

1. Problema humano a resolver:
   - ¿Cómo podemos aprovechar los otros dispositivos de entrada y salida (camara) para generar interacción entre un juego y el entorno físico del usuario?

2. Problema técnico a resolver:
   - Si queremos generar identificar por medio de imágenes a quienes no cumplen la distancia social, ¿cuáles técnicas de procesado de imágenes podemos utilizar para reconocerlos?

3. Antecedentes en trabajos similares:

   - Soler, C. J. (2017). Table tennis ball tracking and bounce calculation using OpenCV.
   - Atamian, A. (2017). Ping Pong Trainer.
   - T Joy, D., Kaur, G., Chugh, A., & Bajaj, S. B. (2021). Computer Vision for Color Detection. International Journal of Innovative Research in Computer Science & Technology (IJIRCST) ISSN, 2347-5552.

Marco teórico de las técnicas a emplear

- Técnica 1: Umbralización

```
# Convert BGR to HSV
hsv = cv.cvtColor(frame, cv.COLOR_BGR2HSV)
#Define range of Blue color in HSV
lower_blue = np.array([100, 82, 42])
upper_blue = np.array([255, 248, 110])
#Define range of Red color in HSV
lower_red = np.array([169, 100, 100])
upper_red = np.array([189, 255, 255])
#Threshold the HSV image to get only blue colors
mask = cv.inRange(hsv, lower_blue, upper_blue)
#Threshold the HSV image to get only red colors
mask2 = cv.inRange(hsv, lower_red, upper_red)
```

Marco teórico de las técnicas a emplear

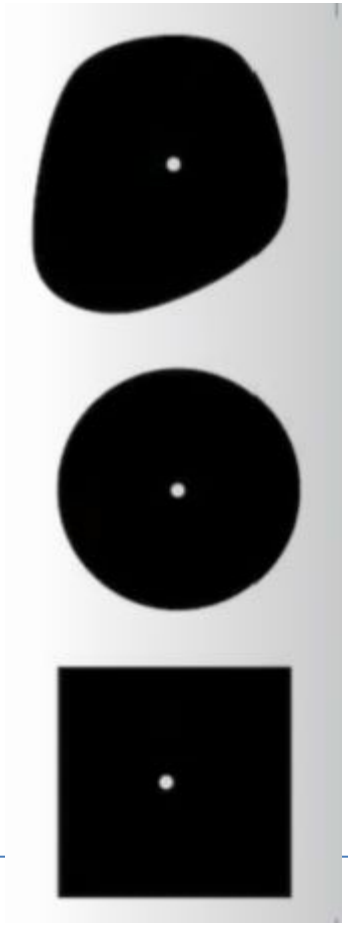
- Técnica 2: Erosión

```
#Aplicamos la erosión a la mascara resultante del rango Azul
filtro1 = cv.erode(mask, cv.getStructuringElement(cv.MORPH_RECT, (3, 3)), iterations=1)
filtro2 = cv.erode(filtro1, cv.getStructuringElement(cv.MORPH_RECT, (5, 5)), iterations=1)

#Aplicamos la erosión a la mascara resultante del rango Rojo
filtro3 = cv.erode(mask2, cv.getStructuringElement(cv.MORPH_RECT, (3, 3)), iterations=1)
filtro4 = cv.erode(filtro3, cv.getStructuringElement(cv.MORPH_RECT, (5, 5)), iterations=1)
```

Marco teórico de las técnicas a emplear

- Técnica 3: Calcular centros

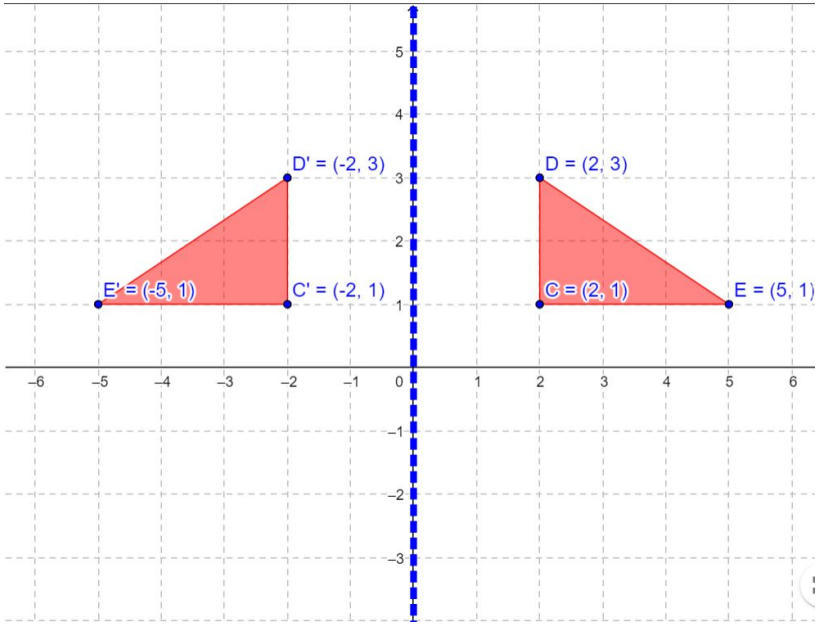
```
try:
    objct = cv.moments(filtro2)
    if objct['m00'] > 50000:
        cx = int(objct['m10'] / objct['m00'] + 1)
        cy = int(objct['m01'] / objct['m00'] + 1)
        cv.circle(frame, (cx, cy), 10, (0, 0, 255), 4)
        y = cy

    objct2 = cv.moments(filtro4)
    if objct2['m00'] > 50000:
        cx2 = int(objct2['m10'] / objct2['m00'] + 1)
        cy2 = int(objct2['m01'] / objct2['m00'] + 1)
        cv.circle(frame, (cx2, cy2), 10, (0, 0, 255), 4)
        y2 = cy2
except:
    print("error")
```



Marco teórico de las técnicas a emplear

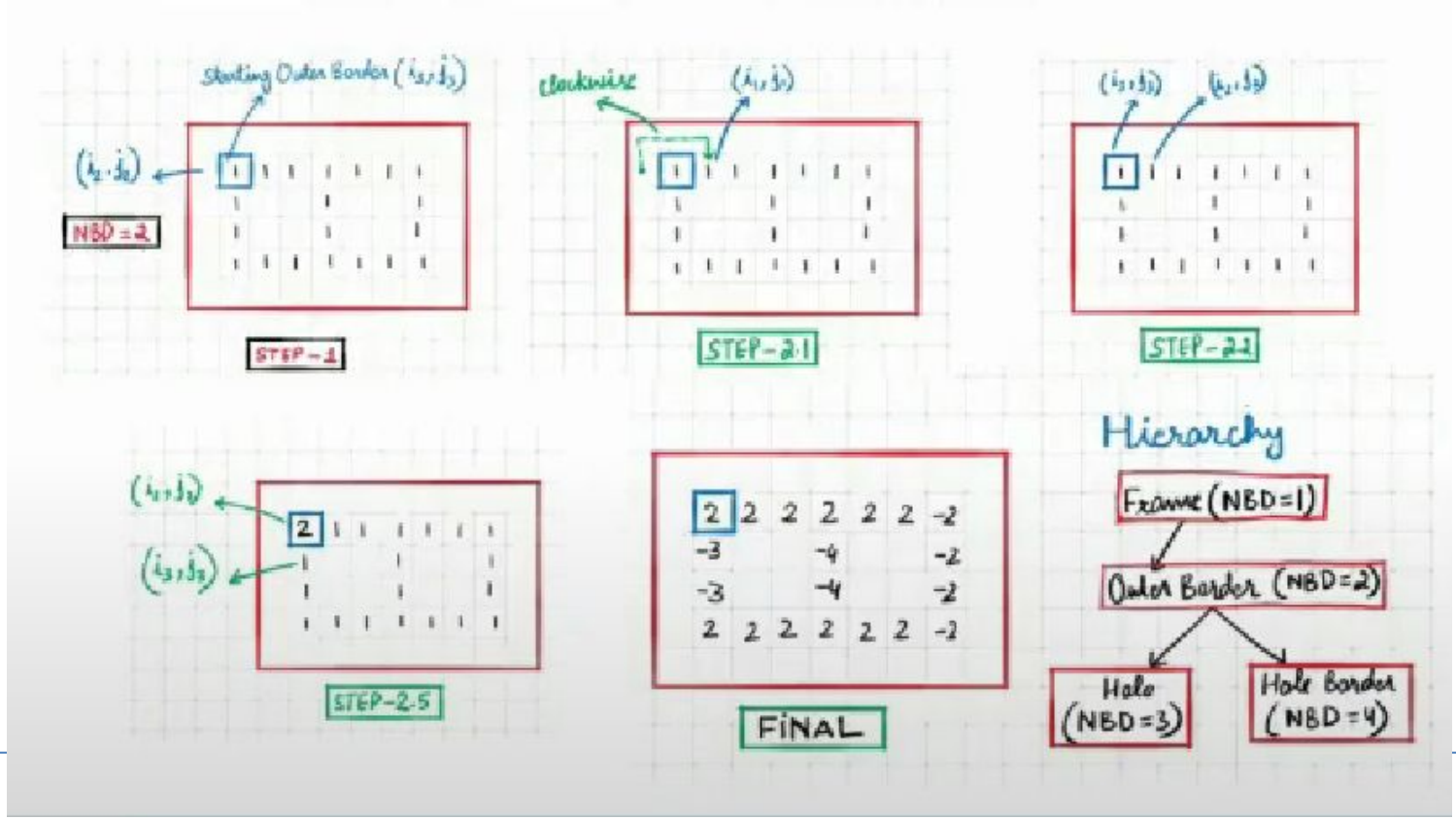
- Técnica 4: Reflexión

```
anchomitad=frame.shape[1]//2
frameEspejo=cv.flip(frame, anchomitad)
cv.imshow('VideoOriginal', frameEspejo)
```



Algoritmos de prueba demostrativos

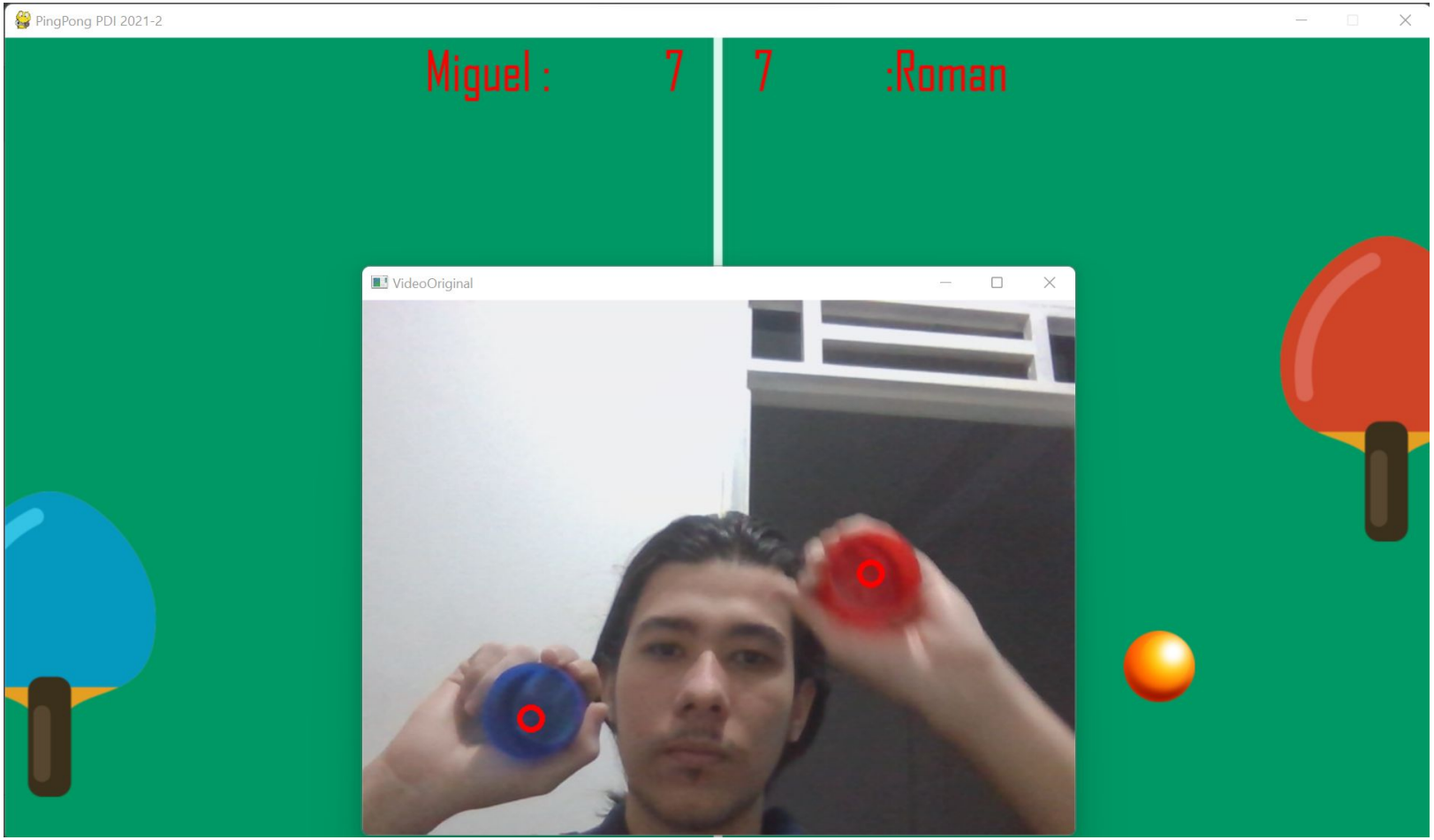
- OpenCV utiliza el algoritmo Suzuki85



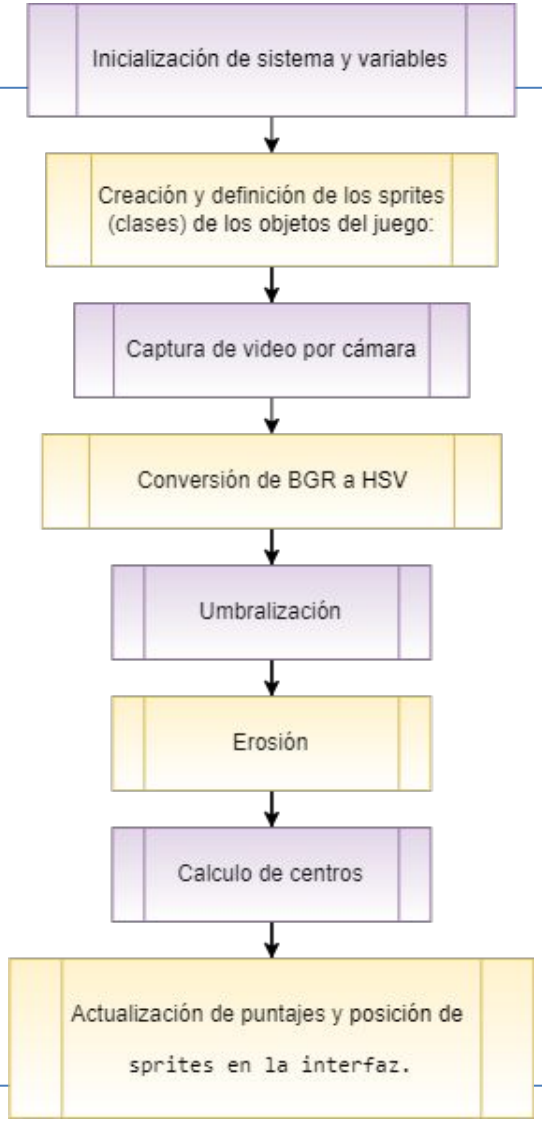
Propuesta de Solución

Conceptos del procesado de imágenes que se utilizaron: Umbralización, detección de contornos, erosión y reflexión.

Propuesta de Solución

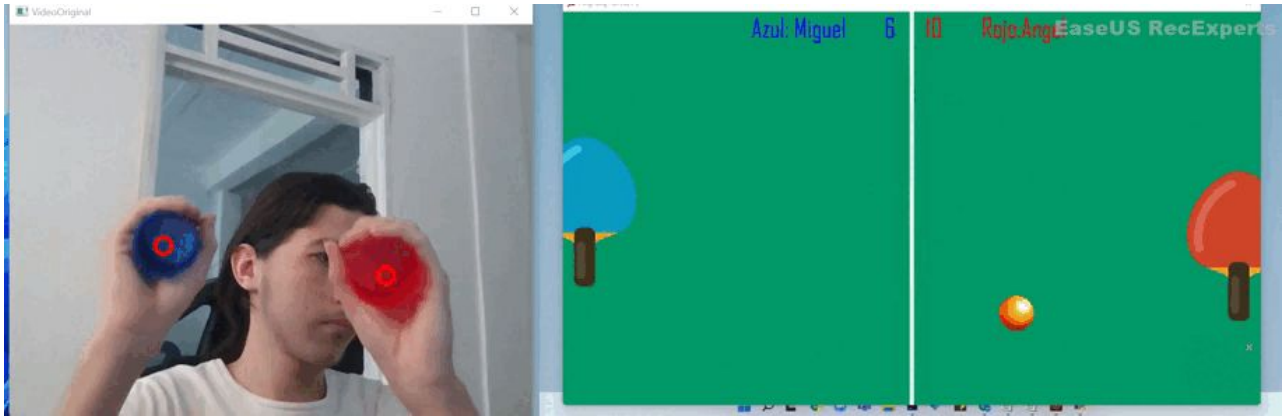


Estructura del Código



Resultados, Líneas Futuras

- Como resultado se observa que el algoritmo detecta los colores, formas y contornos de forma correcta, permitiendo de este modo la diferenciación de los movimientos de cada jugador.



- Líneas futuras:
  - ➢ Un algoritmo que logre detectar el objeto con mayor precisión.
  - ➢ Mejor filtrado y aislamiento de imágenes ante los cambios en las condiciones del entorno (iluminación).
  - ➢ Detección de mayor cantidad de colores, permitiendo mejor interactividad.

Bibliografía y webgrafía

- Barba-Guamán, L., Calderon-Cordova, C., & Quezada-Sarmiento, P. A. Detección de objetos en movimiento a través de la umbralización del color Detection of moving objects through color thresholding.

- Castro Casadiego, S., Sanchez Mojica, K. Y., Puerto, K., Niño, C., Medina Delgado, B., & Guevara-Ibarra, D. (2021). Efecto de los filtros morfológicos en los procesos de detección de objetos en movimiento. Mundo FESC, 11(21 (2021)), 87-95.

- Mordvintsev, A., & Abid, K. (2014). Opencv-python tutorials documentation. Obtenido de https://media. readthedocs. org/pdf/opencv-python-tutroals/latest/opencv-python-tutroals. pdf.

Repositorio del Proyecto:
https://github.com/cayu2312002/PingPong_OpenCV_PDI

Preguntas del público?

Aplausos?…