

Universidad de Guadalajara



Algoritmo 17

Muñoz Nuñez Ian Emmanuel

Visión Robótica

"Algoritmo Canny"

```
# Se importa opencv
import cv2

# Se declara la captura del video
captura = cv2.VideoCapture(0, cv2.CAP_DSHOW)

# Se obtiene el ancho de la captura del video
width = int(captura.get(cv2.CAP_PROP_FRAME_WIDTH))
# Se obtiene la altura de la captura del video
height = int(captura.get(cv2.CAP_PROP_FRAME_HEIGHT))

# Se declara el codec que se usara para grabar el video
fourcc = cv2.VideoWriter_fourcc(*'mp4v')

# Se define el archivo de salida del video original
output_original = cv2.VideoWriter('original.mp4', fourcc, 10, (width, height))
# Se define el archivo de salida del video
output = cv2.VideoWriter('resultado.mp4', fourcc, 10, (width, height), 0)

# Se inicializa el iterador
i = 0
# Comienza el ciclo y termina hasta que haya completado 50 iteraciones
while i <= 50:
    # Con la variable 'leido' se sabe si hay mas frames o no. Se asignan
    # los frames a la variable 'video'
    leido, video = captura.read()
    # Si no hay mas frames en el video...
    if not leido:
        # ...El ciclo termina
        break

    # Si se presiona la tecla 'esc'...
    if cv2.waitKey(1) == 27:
        # ...Se termina el ciclo
        break

    # Se cambia el formato del video a escala de grises
    grises = cv2.cvtColor(video, cv2.COLOR_BGR2GRAY)

    # Se aplica el algoritmo Canny al video en escala de grises
    video_blur = cv2.GaussianBlur(grises, (3, 3), 0)
    bordes = cv2.Canny(image=grises, threshold1=10, threshold2=200)

    # Se muestra el video original
    cv2.imshow('Video', video)
    # Se muestra el video con el algoritmo de Canny
    cv2.imshow('Bordes', bordes)

    # Se escribe el video original
```

```
        output_original.write(video)
        # Se escribe el video resultante
        output.write(bordes)

        # El iterador incrementa 1
        i += 1

# Se cierra la captura del video
captura.release()

# Fin del programa
```

Para el código del algoritmo primero se importa la librería **opencv**, luego se declara la captura del vídeo y se obtienen las dimensiones y el codec, después se declaran las variables con las que se escribirán los vídeos. Se comienza el ciclo para tomar cada uno de los frames del vídeo, este ciclo termina hasta completar 50 iteraciones, se cambia el formato de cada frame de BGR a escala de grises. Luego se aplica el algoritmo Canny y se muestran los vídeos (el original y el resultado). Al final se escriben estos vídeos. Por último se cierra la captura del vídeo.

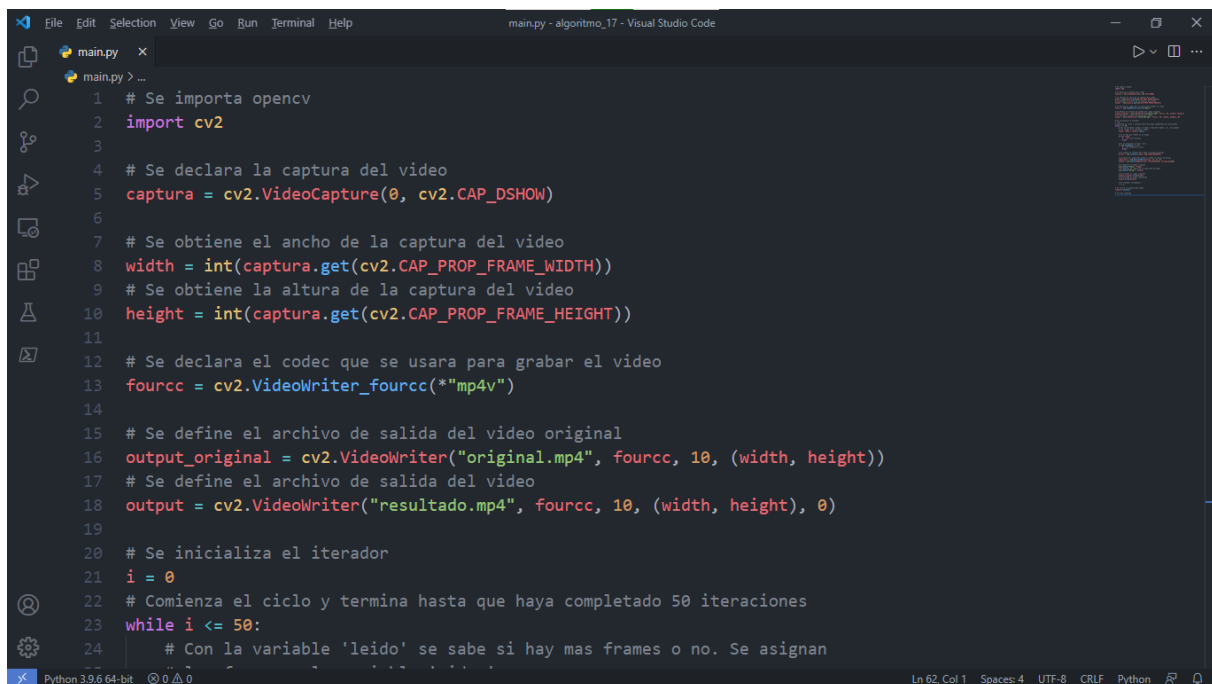
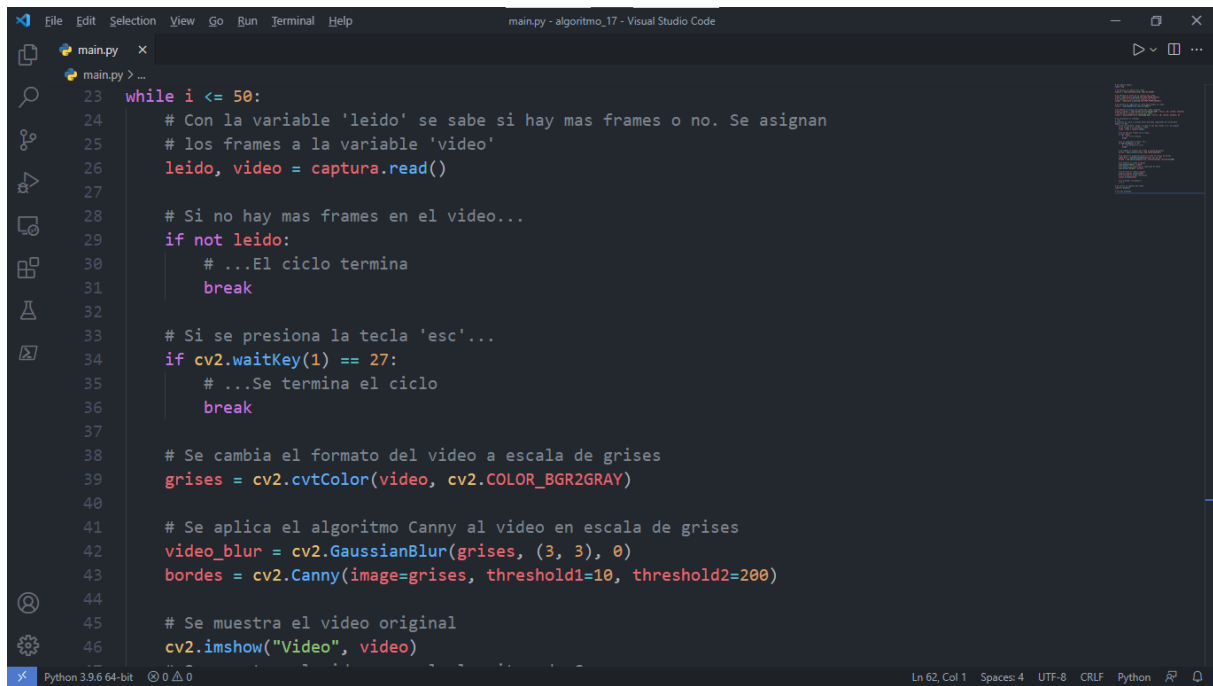
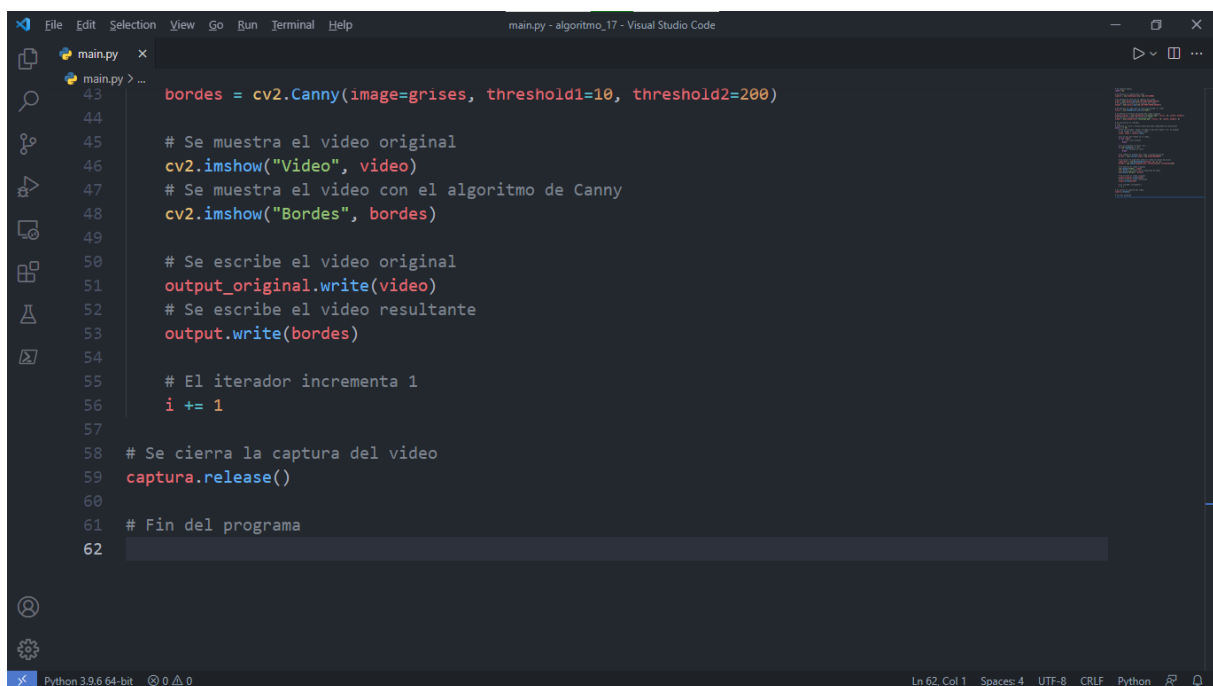


Figura 1: Captura 1 del código



```
23 while i <= 50:
24     # Con la variable 'leido' se sabe si hay mas frames o no. Se asignan
25     # los frames a la variable 'video'
26     leido, video = captura.read()
27
28     # Si no hay mas frames en el video...
29     if not leido:
30         # ...El ciclo termina
31         break
32
33     # Si se presiona la tecla 'esc'...
34     if cv2.waitKey(1) == 27:
35         # ...Se termina el ciclo
36         break
37
38     # Se cambia el formato del video a escala de grises
39     grises = cv2.cvtColor(video, cv2.COLOR_BGR2GRAY)
40
41     # Se aplica el algoritmo Canny al video en escala de grises
42     video_blur = cv2.GaussianBlur(grises, (3, 3), 0)
43     bordes = cv2.Canny(image=grises, threshold1=10, threshold2=200)
44
45     # Se muestra el video original
46     cv2.imshow("Video", video)
```

Figura 2: Captura 2 del código



```
43     bordes = cv2.Canny(image=grises, threshold1=10, threshold2=200)
44
45     # Se muestra el video original
46     cv2.imshow("Video", video)
47     # Se muestra el video con el algoritmo de Canny
48     cv2.imshow("Bordes", bordes)
49
50     # Se escribe el video original
51     output_original.write(video)
52     # Se escribe el video resultante
53     output.write(bordes)
54
55     # El iterador incrementa 1
56     i += 1
57
58     # Se cierra la captura del video
59     captura.release()
60
61     # Fin del programa
62
```

Figura 3: Captura 3 del código

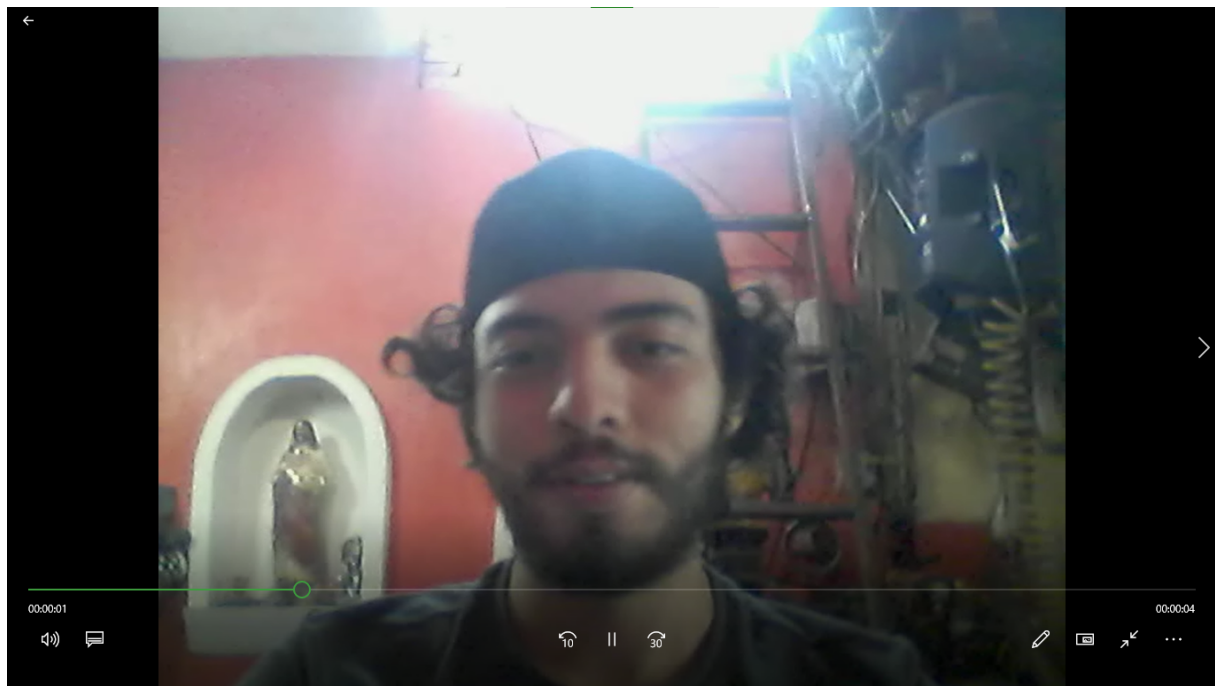


Figura 4: Captura del vídeo original



Figura 5: Captura del vídeo obtenido