Katherine Michelle Ortiz Burgos
Código: 506221041
Juan Esteban Acosta Rodriguez
Código: 506221068
Profesor: Oscar Alexander Mendez Aguirre
Fundación Universitaria Konrad Lorenz

Prototipo Vision : Detección de Persona Puntero En La Nariz

I. Resumen

Este prototipo utiliza las bibliotecas Pygame, OpenCV y MediaPipe para crear una aplicación que muestra la entrada de una cámara en una ventana de Pygame y superpone un círculo en tiempo real en la posición de la nariz detectada en la cara del usuario. El proceso implica la captura de frames de la cámara, la detección facial con MediaPipe para encontrar las coordenadas de la nariz, y luego la representación de esta información en una ventana de Pygame. Además, el código maneja eventos de usuario, como el redimensionamiento de la ventana y el cierre de la aplicación, mientras sigue el movimiento de la nariz con líneas dibujadas en la pantalla.

II. Introducción

El prototipo de seguimiento de la nariz en tiempo real ofrece una integración avanzada entre el frontend y el backend, proporcionando una experiencia completa de seguimiento facial a través de una interfaz web. El frontend permite la visualización en tiempo real del video capturado por la cámara y muestra la posición de la nariz detectada. Por otro lado, el backend se encarga de la detección de la cara y el seguimiento de la posición de la nariz en el flujo de video en tiempo real, utilizando las bibliotecas OpenCV, MediaPipe y Pygame para procesar y analizar los datos del video, detectar la cara del usuario y calcular la posición de la nariz, enviando estos datos al frontend para su visualización.

El código implementado utiliza OpenCV para capturar y procesar el flujo de video en tiempo real desde la cámara, MediaPipe para la detección de puntos faciales específicos como la nariz, y Pygame para la visualización y creación de una interfaz interactiva. Se inicializa Pygame para crear la ventana de visualización y se configura la cámara con OpenCV. En el ciclo principal del programa, se capturan fotogramas de la cámara, se procesan para detectar la cara y la posición de la nariz, y se actualiza la interfaz de Pygame con esta información. Si se detecta la nariz, se dibuja un círculo en la posición correspondiente en el video.

El procesamiento del video en tiempo real se realiza capturando cada fotograma de la cámara y verificando que haya un fotograma disponible. Luego, el fotograma se invierte horizontalmente para que la imagen no esté espejada, se escala a las dimensiones deseadas y se convierte a un formato adecuado para Pygame. Se utiliza MediaPipe para procesar el fotograma y detectar la cara, obteniendo las coordenadas de la punta de la nariz. Si se detecta la nariz, se dibuja un círculo en la posición correspondiente y se actualiza la interfaz de Pygame. Si no se detecta la nariz, se utiliza la última posición válida detectada.

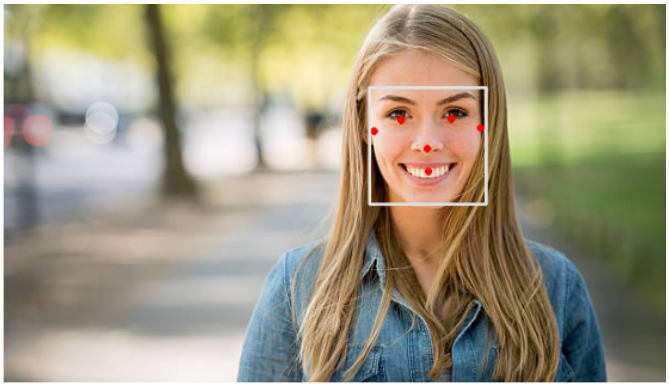

Figura 1.Detección De Rostro.

III. Diseño y desarrollo de la API backend

El diseño y desarrollo de la API backend para el proyecto de seguimiento de la nariz en tiempo real se centra en proporcionar los servicios necesarios para procesar el flujo de video, detectar la cara y la nariz del usuario, y enviar esta información al frontend para su visualización. La API se implementa utilizando las bibliotecas OpenCV, MediaPipe y Pygame, cada una de las cuales desempeña un papel crucial en diferentes etapas del procesamiento.

1. Captura y Procesamiento del Video: El backend inicia capturando el flujo de video en tiempo real desde la cámara del usuario utilizando OpenCV. Esto se realiza a través de cv2.VideoCapture(0), que abre la cámara. En un bucle continuo, se capturan los fotogramas de la cámara y se procesan. Cada fotograma se invierte horizontalmente para evitar que la imagen esté espejada, y luego se escala a las dimensiones deseadas. Posteriormente, se convierte el formato de color de BGR a RGB para que sea compatible con Pygame y MediaPipe.
2. Detección Facial y Seguimiento de la Nariz: Para detectar la cara y la nariz, se utiliza MediaPipe, específicamente el

modelo FaceMesh. Este modelo permite identificar varios puntos de referencia en el rostro, incluyendo la punta de la nariz. Cada fotograma procesado se pasa a través del modelo de MediaPipe, y si se detecta una cara, se extraen las coordenadas de la nariz. Estas coordenadas se utilizan para dibujar un círculo en la posición de la nariz y para actualizar las posiciones anteriores almacenadas.

3. Visualización en Pygame: Finalmente, el backend utiliza Pygame para mostrar los resultados procesados. Cada fotograma procesado se convierte a una superficie que Pygame puede usar para renderizar en la ventana de visualización. Se dibujan elementos adicionales como una cruz en el centro de la ventana y un mensaje de error si no se detecta la nariz. Las posiciones de la nariz se almacenan y se utilizan para dibujar líneas que siguen el movimiento de la nariz, proporcionando una visualización clara y dinámica del seguimiento facial.

IV. Creación de la interfaz gráfica y conexión con el backend

La creación de la interfaz gráfica para el proyecto de seguimiento de la nariz en tiempo real se realiza utilizando la biblioteca Pygame, que facilita la visualización y la interacción del usuario con la aplicación. Esta interfaz gráfica no solo muestra el video en tiempo real capturado por la cámara, sino que también presenta elementos visuales como un círculo que indica la posición de la nariz y otras anotaciones para mejorar la experiencia del usuario. La conexión con el backend es esencial para procesar los datos en tiempo real y actualizar la interfaz de manera dinámica.

1. Configuración de la Interfaz Gráfica:Para comenzar, se inicializa Pygame y se configura la ventana principal de la aplicación con dimensiones específicas y la posibilidad de redimensionarla. Además, se define una fuente para mostrar mensajes en pantalla, y se prepara una lista para almacenar las posiciones de la nariz detectadas, lo cual es útil para dibujar trayectorias.
2. Captura y Procesamiento del Video: Dentro del bucle principal de la aplicación, se manejan los eventos de Pygame, como el cierre de la ventana y el redimensionamiento. Posteriormente, se captura un fotograma de la cámara utilizando OpenCV, se invierte horizontalmente para evitar la imagen espejada, y se escala a las dimensiones adecuadas. Este fotograma se convierte al formato RGB para ser compatible con Pygame y MediaPipe.
3. Conexión con el Backend y Visualización: La conexión con el backend se realiza mediante el uso de MediaPipe para detectar la cara y la nariz en cada fotograma procesado. Las coordenadas de la nariz se utilizan para dibujar un círculo en la posición detectada. Si no se detecta la nariz, se mantiene la última posición válida. La interfaz gráfica se actualiza continuamente con estos datos procesados, mostrando también mensajes de error si la nariz no se puede detectar.
4. nteracción del Usuario y Actualización Dinámica: Además de la detección de la nariz, la interfaz gráfica incluye elementos visuales adicionales como cruces que indican el centro de la pantalla y mensajes que informan al usuario cuando el cursor de la nariz se encuentra fuera de la ventana visible. Las posiciones de la nariz se almacenan y se utilizan para dibujar líneas que representan el movimiento de la nariz, creando una visualización dinámica y continua de la trayectoria.

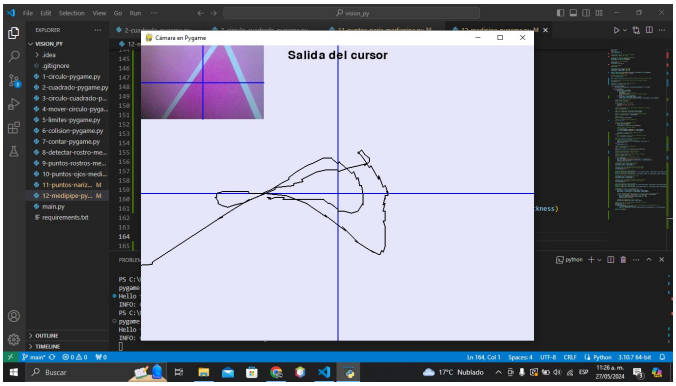

Figura 2. Prototipo Vision.

V. Conclusiones

1. La implementación del prototipo demuestra cómo se pueden integrar de manera eficiente diferentes tecnologías y bibliotecas, como Pygame, OpenCV y MediaPipe, para crear una aplicación interactiva de seguimiento facial en tiempo real. Cada biblioteca cumple una función específica, desde la captura y procesamiento de video hasta la detección de rasgos faciales y la representación gráfica, mostrando cómo se puede construir una solución completa mediante la colaboración de múltiples herramientas.
2. Utilizando MediaPipe para la detección de puntos de referencia faciales, el proyecto destaca las capacidades de detección y seguimiento en tiempo real. La precisión y rapidez de MediaPipe permiten identificar y rastrear la posición de la nariz de manera continua, ofreciendo una experiencia fluida y responsiva.
3. La creación de una interfaz gráfica con Pygame que se adapta a los eventos del usuario, como el redimensionamiento de la ventana y la detección de la salida del cursor, asegura una experiencia de usuario intuitiva y adaptativa. La visualización dinámica de la trayectoria de la nariz y los mensajes informativos mejoran la interactividad y usabilidad de la aplicación, demostrando la importancia de una interfaz bien diseñada para aplicaciones de seguimiento en tiempo real.

VI. Referencias

1. Vision_py. (s/f). GitLab. Recuperado el 28 de mayo de 2024, de `https://gitlab.com/konrad_lorenz/interfaces-2024-1/vision_py`