

Detección de Postura con MediaPipe utilizando Hilos, Mutex y Semáforos

Yossed Mauricio Riaño Páez , Jeferson Jair Hernandez Garzon

2025

1. Introducción

El propósito de esta práctica fue implementar un sistema capaz de detectar en tiempo real si una persona se encuentra en posición de pie o sentada. Para esto se empleó el modelo de reconocimiento de pose de la librería **MediaPipe**, junto con técnicas de **programación concurrente** como hilos, semáforos y mutex, con el fin de gestionar adecuadamente el flujo de datos provenientes de la cámara web.

La detección de postura se basa en el análisis de puntos clave del cuerpo (landmarks), principalmente hombros y caderas, para determinar la relación proporcional del torso y clasificar la postura.

2. Instalación de Dependencias

Antes de ejecutar el sistema, fue necesario instalar las siguientes librerías:

- **MediaPipe**: permite la detección de puntos corporales.
- **OpenCV**: captura y procesamiento de imágenes.
- **Streamlit**: interfaz gráfica en navegador.
- **Threading y Queue**: manejo de hilos y buffer de frames.

2.1. Instalación de MediaPipe

```
Requirement already satisfied: aio>1.5 in c:\Users\jeferson\AppData\Local\packages\pythonsoftwarefoundation.python.3.11_qbz5k2frapq\localcache\local-packages\python311\
site-packages (from python-datutil>2.7>matplotlib>mediapipe) (1.12.0)
INFO: pip is looking at multiple versions of opencv-contrib-python to determine which version is compatible with other requirements. This could take a while.
Collecting opencv-contrib-python (from mediapipe)
  Downloading opencv-contrib-python-4.11.0.80-cp37-abi3-win_amd64.whl.metadata (20 kb)
Downloaded mediapipe-0.10.21-cp311-cp311-win_amd64.whl (51.0 MB)
 51.0/51.0 MB 11.5 MB/s 0:00:04
Downloaded numpy-1.26.4-cp311-cp311-win_amd64.whl (15.8 MB)
 15.8/15.8 MB 11.5 MB/s 0:00:01
Downloaded protobuf-4.25.8-cp310-abi3-win_amd64.whl (413 kb)
Downloaded attrs-25.4.0-py3-none-any.whl (67 kb)
Downloaded flatbuffers-25.9.23-py2.py3-none-any.whl (30 kb)
Downloaded sounddevice-0.5.3-py3-none-win_amd64.whl (364 kb)
Downloaded cffi-2.0.0-cp311-cp311-win_amd64.whl (182 kb)
Downloaded absl_py-2.3.1-py3-none-any.whl (135 kb)
Downloaded jax-0.7.1-py3-none-any.whl (2.8 MB)
 2.8/2.8 MB 11.0 MB/s 0:00:00
Downloaded jaxlib-0.7.1-cp311-cp311-win_amd64.whl (61.2 MB)
 61.2/61.2 MB 11.0 MB/s eta 0:00:01
```

Figura 1: Instalación de MediaPipe en consola.

Se observa la instalación de MediaPipe mediante pip, necesaria para la detección de landmarks.

2.2. Instalación y prueba de Streamlit

```
Downloading rpds_py-0.28.0-cp311-cp311-win_amd64.whl.metadata (4.2 kB)
Requirement already satisfied: six>1.5 in c:\users\jeferson\appdata\local\packages\pythonsoftwarefoundation.python.3.11_qbz5n2kfra8p0\localcache\local-packages\python311\
site-packages (from python-dateutil>2.8.2>pandas>3.9.1.4.0->streamlit) (1.17.0)
Downloading streamlit-1.31.0-py3-none-any.whl (10.2 MB)
10.2/10.2 MB 9.6 MB/s 0:00:01
Downloading altair-5.5.0-py3-none-any.whl (731 kB)
731.2/731.2 kB 10.0 MB/s 0:00:00
Downloading blinker-1.9.0-py3-none-any.whl (8.5 kB)
Downloading cachetools-6.2.1-py3-none-any.whl (11 kB)
Downloading click-8.3.0-py3-none-any.whl (107 kB)
Downloading gitpython-3.1.45-py3-none-any.whl (208 kB)
Downloading gitdb-4.0.12-py3-none-any.whl (62 kB)
Downloading pandas-2.3.3-cp311-cp311-win_amd64.whl (11.3 MB)
11.3/11.3 MB 9.6 MB/s 0:00:01
Downloading pyarrow-21.0.0-cp311-cp311-win_amd64.whl (26.2 MB)
26.2/26.2 MB 9.3 MB/s 0:00:02
Downloading pydeck-0.9.1-py2.py3-none-any.whl (6.9 MB)
6.9/6.9 MB 7.7 MB/s 0:00:00
Downloading requests-2.32.5-py3-none-any.whl (64 kB)
```

Figura 2: Instalación y prueba de Streamlit.

Se verifica que Streamlit se haya instalado correctamente.

2.3. Prueba de cámara en Streamlit



Figura 3: Prueba de visualización de cámara en Streamlit.

Se confirma que la cámara funciona y los frames están listos para ser procesados.

3. Funcionamiento del Sistema

El sistema se divide en dos hilos principales:

1. **Hilo productor:** captura los frames de la cámara y los deposita en un buffer.
2. **Hilo consumidor:** toma los frames del buffer, procesa la pose con MediaPipe y determina si la persona está de pie o sentada.

Para evitar conflictos entre los hilos, se implementaron:

- **Mutex:** protege secciones críticas al acceder al buffer.
- **Semáforos:** controlan disponibilidad de espacio e ítems en el buffer.

4. Ejemplos de detección de postura

4.1. Persona de pie

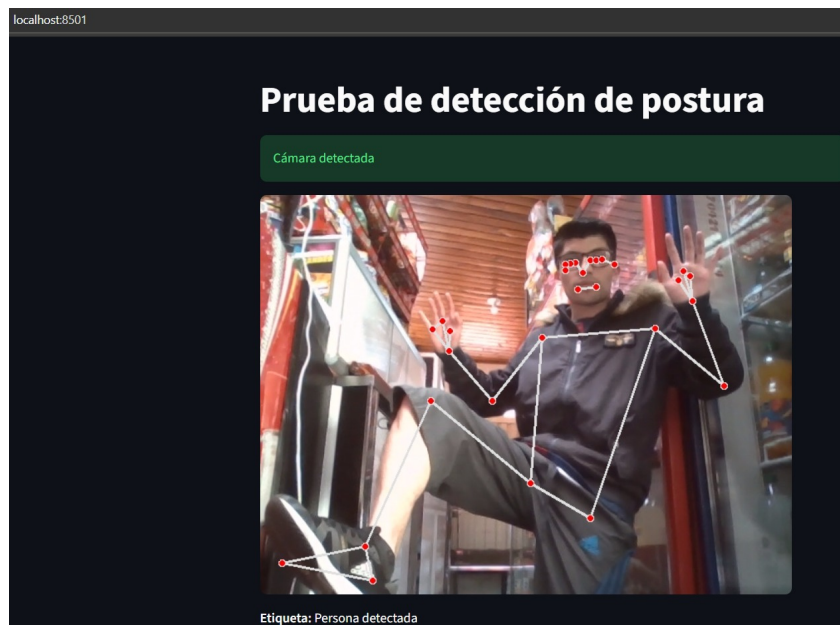


Figura 4: Detección de postura: Persona de pie.

El sistema identifica correctamente una postura de pie mediante los landmarks de hombros y caderas.

4.2. Persona sentada

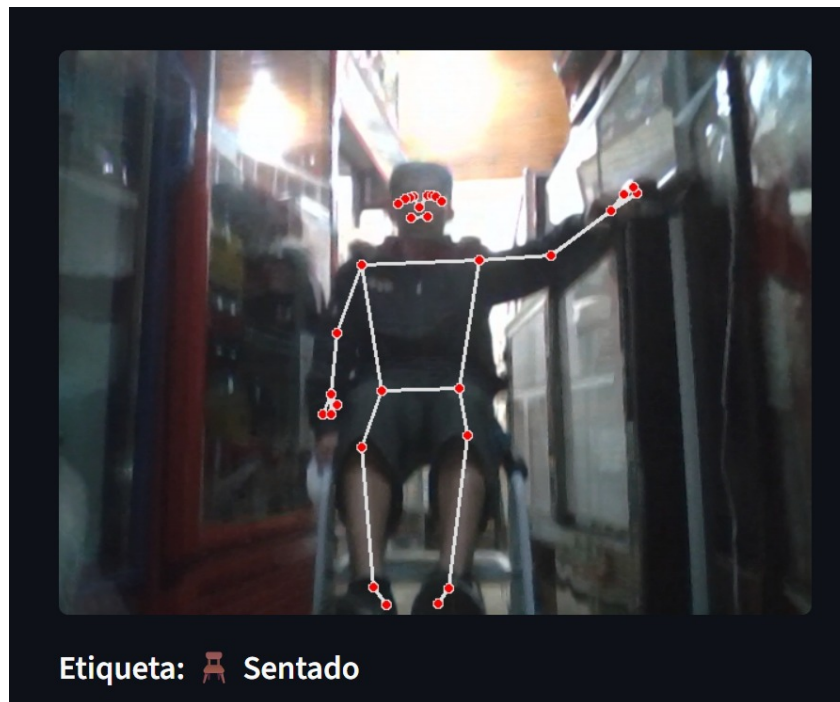


Figura 5: Detección de postura: Persona sentada.

El sistema reconoce la postura de sentado comparando la proporción torso-hombros-caderas.

5. Código Fuente

```
import streamlit as st
import cv2
import mediapipe as mp
import threading
import time
import queue
```

```
st.set_page_config(page_title="Detección de Postura | Hilos y Semáforos", layout="wide")
st.title("Detección de postura | De pie / Sentado")
```

```
mutex_postura = threading.Lock()
semaforo_espacios = threading.Semaphore(4)
semaforo_items = threading.Semaphore(0)
buffer_frames = queue.Queue(maxsize=4)
ejecucion_activa = threading.Event()
```

```
etiqueta_postura = "Esperando detección..."
ultimo_frame = None
```

```

mp_pose = mp.solutions.pose
mp_dibujo = mp.solutions.drawing_utils

def hilo_productor():
    global buffer_frames
    camara = cv2.VideoCapture(0)
    if not camara.isOpened():
        st.error("No se pudo abrir la cámara")
        ejecucion_activa.clear()
        return
    while ejecucion_activa.is_set():
        ret, frame = camara.read()
        if not ret:
            continue
        frame = cv2.flip(frame, 1)
        semaforo_espacios.acquire()
        with mutex_postura:
            if not buffer_frames.full():
                buffer_frames.put(frame)
        semaforo_items.release()
        time.sleep(0.03)
    camara.release()

def hilo_consumidor():
    global ultimo_frame, etiqueta_postura
    pose = mp_pose.Pose(static_image_mode=False, model_complexity=1,
                        enable_segmentation=False, min_detection_confidence=0.5,
                        min_tracking_confidence=0.5)
    while ejecucion_activa.is_set():
        semaforo_items.acquire()
        with mutex_postura:
            if not buffer_frames.empty():
                frame = buffer_frames.get()
        semaforo_espacios.release()
        imagen_rgb = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
        resultado = pose.process(imagen_rgb)
        texto_etiqueta = "Sin detección"
        if resultado.pose_landmarks:
            mp_dibujo.draw_landmarks(frame, resultado.pose_landmarks, mp_pose.POSE_CO
            puntos = resultado.pose_landmarks.landmark
            hombros_y = (puntos[11].y + puntos[12].y) / 2
            caderas_y = (puntos[23].y + puntos[24].y) / 2
            altura_torso = abs(caderas_y - hombros_y)
            texto_etiqueta = "Persona de pie" if altura_torso > 0.30 else "Persona se
        with mutex_postura:
            ultimo_frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
            etiqueta_postura = texto_etiqueta
        time.sleep(0.03)

```

```

area_video = st.empty()
texto_resultado = st.empty()
col1, col2 = st.columns(2)

if col1.button("Iniciar detección"):
    if not ejecucion_activa.is_set():
        ejecucion_activa.set()
        hilo_1 = threading.Thread(target=hilo_productor, daemon=True)
        hilo_2 = threading.Thread(target=hilo_consumidor, daemon=True)
        hilo_1.start()
        hilo_2.start()
        st.success("Detección iniciada")

if col2.button("Detener"):
    ejecucion_activa.clear()
    st.warning("Detección detenida")

while True:
    if ultimo_frame is not None:
        area_video.image(ultimo_frame, channels="RGB")
        texto_resultado.markdown(f"### {etiqueta_postura}")
    else:
        texto_resultado.markdown("Esperando cámara...")
    time.sleep(0.05)

```

6. Dockerfile

```

# Imagen base de Python con Debian (más completa que slim)
FROM python:3.10-bullseye

# Evitar preguntas interactivas en apt
ENV DEBIAN_FRONTEND=noninteractive

# Actualizar repositorios e instalar librerías del sistema necesarias
RUN apt-get update && apt-get install -y \
    libgl1-mesa-glx \
    libglib2.0-0 \
    && apt-get clean && rm -rf /var/lib/apt/lists/*

# Establecer el directorio de trabajo
WORKDIR /app

# Copiar todos los archivos del proyecto al contenedor
COPY . /app

# Instalar dependencias de Python

```

```
RUN pip install --no-cache-dir streamlit opencv-python mediapipe
```

```
# Exponer el puerto que usa Streamlit  
EXPOSE 8501
```

```
# Comando para ejecutar la app  
CMD ["streamlit", "run", "quiz.py"]
```

7. Conclusiones

- MediaPipe ofrece una forma eficiente y precisa de detectar poses humanas en tiempo real.
- El uso de hilos mejora el rendimiento del sistema evitando bloqueos y retrasos en la captura de video.
- Los semáforos y mutex permiten coordinar correctamente el acceso al buffer, evitando condiciones de carrera.
- Streamlit proporciona una forma rápida y funcional de crear interfaces para visualización en tiempo real.