

**UNIVERSIDAD NACIONAL DE SAN AGUSTÍN**  
**FACULTAD DE INGENIERÍA DE PRODUCCIÓN Y SERVICIOS**  
**ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMAS**



---

**Manual de Usuario — SignPeru**

---

**Trabajo presentado por:**

- Juan Sergio Zeballos Perez
- Wilson Josue Turpo Huanca
- Choquehuanca Berna William Herderson

**Curso:**  
TECNOLOGÍA DE OBJETOS

**Fecha de entrega:**  
Hasta el lunes 7-12-2025  
a las 23:59 horas

**AREQUIPA-PERÚ**

**2025**

<b>Manual de Usuario — SignPeru.....</b>	<b>3</b>
1. Resumen / propósito.....	3
2. Requisitos.....	3
Hardware mínimo.....	3
Software.....	3
Dependencias principales (ejemplos).....	3
3. Estructura principal del proyecto.....	4
4. Preparación inicial (instalación y base de datos).....	5
5. Arrancar la aplicación.....	5
a) Modo GUI (recomendado).....	5
b) Modo prueba: lanzar un juego directamente.....	6
6. Uso de la GUI principal (MainWindow).....	6
7. Uso — controles de los juegos.....	7
Juego: Ahorcado (AH).....	7
Juego: Letras que Caen (LC).....	7
Juego: Ladrillos / Arkanoid (LADRILLOS).....	7
8. Guardado de puntuaciones.....	8
9. Ejecutar pruebas rápidas.....	8
10. Solución de problemas comunes.....	8
A. La cámara no muestra frames / está ocupada.....	8
B. Mensajes largos de TensorFlow (oneDNN, XNNPACK).....	9
C. Error: “handler error for hand_detected: got an unexpected keyword argument 'landmarks'”.....	9
D. “The truth value of an array with more than one element is ambiguous...”.....	9
E. Pygame no abre ventana (headless server).....	9
F. Velocidad/pérdida de frames.....	9
11. Preguntas frecuentes (FAQ).....	9
12. Notas para desarrolladores / mantenimiento.....	10
13. Comandos útiles resumen.....	10

# Manual de Usuario — SignPeru

*(Sistema de Reconocimiento de la Lengua de Señas Peruana + Juegos Serios)*

## 1. Resumen / propósito

SignPeru es un prototipo que reconoce señas (LSP) en tiempo real usando una webcam y permite practicar mediante juegos interactivos: Ahorcado (AH), Letras que Caen (LC) y Arkanoid / Ladrillos (LADRILLOS). La captura y el procesamiento corren en hilos independientes y se comunican con los juegos mediante un **EventBus**.

## 2. Requisitos

### Hardware mínimo

- CPU dual-core
- 8 GB RAM
- Webcam (preferiblemente 720p, 30 FPS)
- Altavoces (opcional, para sonidos del juego)

### Software

- Sistema operativo: Windows / Linux / macOS (se probó en Windows)
- Python 3.10+ (en nuestro entorno usamos 3.12)
- Entorno virtual **venv** recomendado

### Dependencias principales (ejemplos)

- opencv-python
- mediapipe
- tensorflow / tf-lite-runtime (según tu modelo)
- numpy
- pillow
- pygame
- customtkinter (CTk)
- imageio

Instálalas con:

None

```
python -m venv venv
```

```
source venv/bin/activate    # Windows: venv\Scripts\activate
pip install -r requerimientos.txt
```

(si tu repo incluye `requerimientos.txt`, úsalo; si no, instala las librerías listadas arriba)

### 3. Estructura principal del proyecto

(ubicada en `sr1sp-game/src/signperu/`)

```
None
app.py
config.py
core/
  capture.py
  processing.py
  detector.py
  events.py
gui/
  main_window.py
  frames/...
games/
  game_base.py
  juego_AH.py
  juego_LC.py
  juego_ladrillos.py
  clase_ah.py
  clase_lc.py
  clase_ladrillos.py
persistence/
  db_manager.py
  models.sql
data/
  signperu.db
...
```

### 4. Preparación inicial (instalación y base de datos)

1. Clona o sitúa el proyecto en tu máquina.

Crea y activa un entorno virtual:

None

```
python -m venv venv  
source venv/bin/activate    # Windows: venv\Scripts\activate
```

2. Instala dependencias:

None

```
pip install -r requerimientos.txt
```

3. o, si no hay `requerimientos.txt`, instala manualmente ej:

None

```
pip install opencv-python mediapipe numpy pillow pygame customtkinter  
imageio tensorflow
```

Inicializa la base de datos (si incluyes script `init_db.py`):

None

```
python src/signperu/init_db.py
```

4. Si no, verifica que `src/signperu/data/signperu.db` exista.

## 5. Arrancar la aplicación

### a) Modo GUI (recomendado)

Inicia la ventana principal (MainWindow) que controla la cámara y lanza los juegos:

None

```
python -m signperu.app
```

La GUI principal te mostrará:

- Botones para **Iniciar cámara / Detener cámara**
- Preview (miniatura) de la cámara
- Botones para lanzar los juegos (AH, LC, LADRILLOS)

- Consola simple para logs

## b) Modo prueba: lanzar un juego directamente

Para pruebas rápidas puedes arrancar un juego directamente (el script inicia los hilos de captura y procesamiento):

```
None
python -m signperu.app --game AH
# o
python -m signperu.app --game LC
# o
python -m signperu.app --game LADRILLOS
```

## 6. Uso de la GUI principal (MainWindow)

1. Al iniciar `app.py` sin argumentos se abre la **ventana principal**.
2. **Iniciar cámara**: pulsa el botón “Iniciar cámara”.
  - Esto arranca `CaptureThread` y `ProcessingThread` y comienza a publicarse `frame_captured` y `hand_detected` en el `EventBus`.
  - El preview mostrará la miniatura de la cámara.
3. **Detectar letras**: cuando el detector reconozca una letra, la GUI actualizará “Última detección”.
4. **Lanzar juego**:
  - Pulsar el botón del juego deseado. Si la cámara no estaba activa, la GUI la iniciará automáticamente antes de arrancar el juego.
  - La ventana principal se oculta (`withdraw`) y el juego se ejecuta en modo bloqueante. Cuando el juego termine, la ventana principal reaparecerá.
5. **Detener cámara**: corta los hilos de captura/procesamiento y para el preview.

## 7. Uso — controles de los juegos

### Juego: Ahorcado (AH)

- Interfaz: (Pygame o Tkinter dependiendo de la implementación).
- Control principal: el juego recibe eventos `hand_detected` con una letra.
  - Ejemplo: si detecta la letra correcta, la palabra se revela; si no, se pierde una vida.
- Teclas: normalmente `ESC` para salir, otras teclas según la implementación específica en `juego_AH.py`.

### **Juego: Letras que Caen (LC)**

- Visual: panel cámara a la izquierda + área de juego a la derecha (como implementaste en `juego_lc.py`).
- Objetivo: hacer desaparecer la letra que cae mostrando la seña correcta.
- Controles:
  - Señales reconocidas (A–Z) comparadas con letras en pantalla.
  - `ESC` para salir del juego.
- Notas: la caja grande debajo del panel de cámara muestra la letra detectada.

### **Juego: Ladrillos / Arkanoid (LADRILLOS)**

- Lógica separada en `clase_ladrillos.py`; UI en `juego_ladrillos.py`.
- Visual: cámara incrustada dentro del canvas principal (no Toplevel).
- Control:
  - Se mapea la seña `'A'` → mover paleta izquierda
  - Se mapea la seña `'B'` → mover paleta derecha
  - También puedes extender `ClaseLadrillos.process_detection()` para más letras/acciones.
- Cierre: el juego vuelve al menú principal al finalizar (ganar/perder) o al cerrar la ventana.

## **8. Guardado de puntuaciones**

- Los juegos llaman a `save_score(score, game_name=...)` (implementado en `game_base.py`) al finalizar; las puntuaciones quedan en la BD `data/signperu.db`.
- Puedes consultar la BD con cualquier cliente SQLite.

## 9. Ejecutar pruebas rápidas

- `test/quick_test_debug.py`: script sencillo para testear la cámara y dibujo de frames.

None

```
python src/signperu/test/quick_test_debug.py
```

- `tests/test_detector.py`: pruebas unitarias para el detector (requiere dependencias y posiblemente modelos).

## 10. Solución de problemas comunes

### A. La cámara no muestra frames / está ocupada

- Asegúrate de que no haya otra aplicación usando la cámara (Zoom, Teams, navegador).
- Verifica `config.CAMERA_SRC` (si usas cámara USB secundaria, pon `1`, `2`, ...).
- En Windows a veces `cv2.VideoCapture(0, cv2.CAP_DSHOW)` ayuda (ya incluido en `capture.py`).

### B. Mensajes largos de TensorFlow (oneDNN, XNNPACK)

- Mensajes informativos de TensorFlow (INFO/WARNING) son normales. No detienen la ejecución.
- Si quieres silenciarlos, ajusta variables de entorno (ej. `TF_CPP_MIN_LOG_LEVEL`) — pero no es obligatorio.

### C. Error: “handler error for hand\_detected: got an unexpected keyword argument 'landmarks'”



- Significa que un `handler` (tu juego) esperaba un tipo de firma distinto. Las versiones robustas de `juego_lc` y `juego_ladrillos` aceptan `*args`, `**kwargs`.
- Solución: usa la implementación que procesa `*args/**kwargs` (en tus archivos refactorizados ya está corregido).

#### D. “The truth value of an array with more than one element is ambiguous...”

- Proviene de evaluar `numpy.ndarray` como booleano. Evita condiciones como `if arr;` usa `if arr is not None` o `if arr.size>0`.
- Tus clases ya usan comprobaciones explícitas; revisa handlers personalizados si aparecen.

#### E. Pygame no abre ventana (headless server)

- Pygame necesita un display. En servidores sin cabeza usa Xvfb o evita ejecutar juegos que requieren ventana.

#### F. Velocidad/pérdida de frames

- Reduce FPS en `config` (por ejemplo 12) o disminuye tamaño de frame en `capture.py`.
- Aumenta tamaño de cola o ajusta `frame_queue` en `CaptureThread`.

## 11. Preguntas frecuentes (FAQ)

### ¿Puedo usar otra cámara?

Sí: cambia `config.CAMERA_SRC` o modifica parámetro `src` en `app.py/MainWindow.start_capture()`.

### ¿Cómo añadir un nuevo juego?

- Implementa una clase que herede `GameBase` y defina `start()`, `stop()` e `on_hand_detected()`.
- Registra la clase en `app.py / MainWindow` para que aparezca el botón.

### ¿Cómo mejorar la precisión del detector?

- Recolecta más datos, entrena modelos más robustos (CNN, LSTM) y ajusta preprocesamiento (normalización, data augmentation).

## 12. Notas para desarrolladores / mantenimiento

- **Hilos y cierre ordenado:** siempre detener `CaptureThread` y `ProcessingThread` al cerrar la app para liberar la cámara. `MainWindow._on_close()` ya hace este trabajo.
- **EventBus:** patrón pub/sub central — los juegos se suscriben a `frame_captured` y `hand_detected`.
- **Separación lógica / UI:** usa `clase_*` para la lógica y `juego_*` para UI. Esto facilita tests unitarios de la lógica.
- **Logs:** agrega logs en `utils/logger.py` y usa la `console` en `MainWindow` para depuración.

## 13. Comandos útiles resumen

Crear entorno e instalar:

```
None
python -m venv venv
source venv/bin/activate
pip install -r requerimientos.txt
```

Inicializar BD:

```
None
python src/signperu/init_db.py
```

Ejecutar GUI/launcher:

```
None
python -m signperu.app
```

Ejecutar juego directo (modo test):

None

```
python -m signperu.app --game LC
```