

Apéndice F. Procedimientos complementarios de integración y sincronización del sistema

En este apéndice se describen los procedimientos complementarios implementados para integrar los módulos de adquisición, procesamiento, detección, clasificación y accionamiento del sistema automatizado de clasificación de frutos cítricos. Se presenta la arquitectura software desarrollada, la organización de los módulos que conforman la aplicación, el flujo de información entre ellos y los mecanismos utilizados para sincronizar la adquisición espectral con la inferencia y el accionamiento físico del prototipo.

A diferencia de la descripción presentada en la Sección 3.6, donde se expone el funcionamiento general del sistema integrado, este apéndice documenta los aspectos de implementación relacionados con la organización del software, la gestión de los datos durante el procesamiento, la planificación temporal de eventos, la comunicación entre módulos y los parámetros de configuración empleados durante la operación del sistema.

F.1 Arquitectura del software del sistema integrado

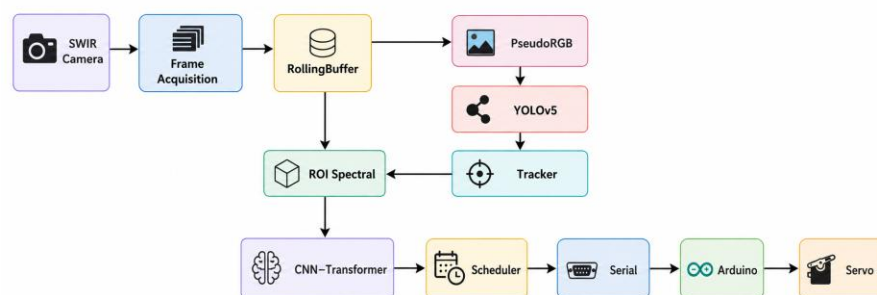
El software del sistema fue desarrollado utilizando Python como plataforma principal de procesamiento, debido a la disponibilidad de bibliotecas especializadas para adquisición de imágenes hiperespectrales, visión por computador, aprendizaje profundo, seguimiento de objetos e interfaces gráficas. La arquitectura fue diseñada bajo un enfoque modular, en el cual cada componente implementa una responsabilidad específica y se comunica con los demás mediante estructuras de datos compartidas y parámetros centralizados, facilitando el desarrollo, la depuración y el mantenimiento del sistema.

El flujo de procesamiento inicia con la adquisición continua de líneas espectrales provenientes de la cámara SWIR. Posteriormente, estas líneas son almacenadas temporalmente en

un RollingBuffer, encargado de reconstruir continuamente la escena correspondiente a la zona de captura. A partir de la información reconstruida, el sistema genera la representación pseudo-RGB utilizada por el detector YOLOv5, mientras que la información espectral completa permanece disponible para el proceso de clasificación.

Una vez detectadas las mandarinas presentes en la escena, el módulo de seguimiento asigna un identificador único a cada fruto y mantiene su correspondencia durante el desplazamiento sobre la banda transportadora. Cuando el centroide de una detección cruza la línea virtual de clasificación, el sistema determina el carril correspondiente, extrae la región espectral asociada y ejecuta la inferencia mediante el modelo CNN-Transformer. Finalmente, el resultado de clasificación es enviado al módulo de planificación temporal, responsable de programar el instante de accionamiento del mecanismo de separación y transmitir posteriormente la orden correspondiente al Arduino UNO mediante comunicación serial.

Figura F.1 *Arquitectura software del sistema integrado.*



F.1.1 Configuración centralizada del sistema

Con el propósito de garantizar un comportamiento uniforme entre todos los módulos de la aplicación, los parámetros de operación fueron centralizados en el archivo config.py. Este módulo reúne la configuración utilizada durante la adquisición, reconstrucción de imágenes, detección,

clasificación, seguimiento y sincronización, evitando la duplicación de parámetros dentro del código y facilitando su modificación cuando cambian las condiciones de operación del sistema.

Además de definir los parámetros de funcionamiento, este módulo incorpora verificaciones automáticas durante la inicialización de la aplicación, permitiendo detectar configuraciones inconsistentes antes de iniciar el proceso de adquisición e inferencia.

Tabla F.1. *Principales parámetros definidos en el módulo config.py*

Parámetro	Función dentro del sistema
Bandas pseudo-RGB	Bandas utilizadas para construir la representación pseudo-RGB empleada por YOLOv5.
Bandas espectrales adicionales	Bandas complementarias utilizadas por el modelo CNN–Transformer.
YOLO_CONF_THRESHOLD	Umbral mínimo de confianza para aceptar una detección.
YOLO_IOU_THRESHOLD	Umbral de IoU empleado durante la supresión de detecciones redundantes.
CENTER_CROSSING_DELAY_S	Retardo entre el cruce del centroide por la línea virtual y el accionamiento del mecanismo de separación.
Parámetros del tracker	Configuración utilizada para mantener la identidad de cada fruto durante el seguimiento.
Puerto serial y velocidad de transmisión	Configuración empleada para la comunicación con el Arduino UNO.

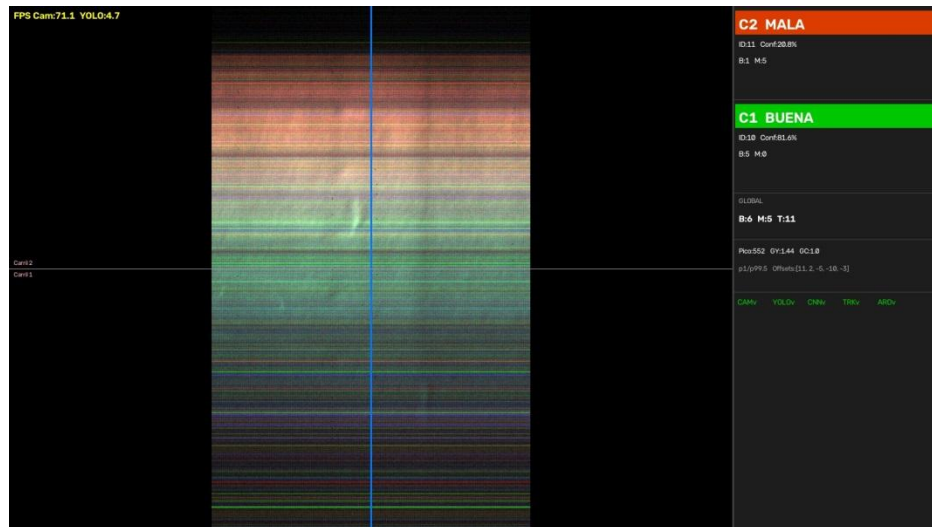
La centralización de estos parámetros permitió mantener un único punto de configuración para toda la aplicación, reduciendo inconsistencias entre módulos y facilitando la adaptación del sistema frente a cambios en la configuración experimental.

F.2 Adquisición continua y reconstrucción mediante RollingBuffer

La adquisición de información espectral se implementó utilizando la cámara SWIR con un montaje pushbroom, en el cual la cámara genera continuamente líneas espectrales en lugar de imágenes bidimensionales completas. Cada línea adquirida contiene la información espacial correspondiente a una fila de la escena junto con la respuesta espectral registrada para todas las longitudes de onda disponibles.

Con el fin de reconstruir continuamente la escena sin interrumpir el proceso de adquisición, se implementó una estructura de almacenamiento tipo RollingBuffer. Este módulo mantiene en memoria un número fijo de líneas espectrales correspondiente a la longitud de reconstrucción definida para la zona de captura. A medida que se recibe una nueva línea desde la cámara, la línea más antigua es descartada y la nueva adquisición se incorpora automáticamente al buffer, manteniendo una representación continuamente actualizada de la escena.

Figura F.2. *Funcionamiento del módulo RollingBuffer durante la reconstrucción continua de la escena.*



Una vez actualizado el buffer, el sistema selecciona las cinco bandas espectrales definidas durante el análisis de separabilidad (¡Error! No se encuentra el origen de la referencia.), reconstruyendo la escena correspondiente a dichas longitudes de onda. Posteriormente, tres de estas bandas son utilizadas para generar la representación pseudo-RGB destinada al detector YOLOv5, mientras que las cinco bandas permanecen disponibles para el proceso de clasificación.

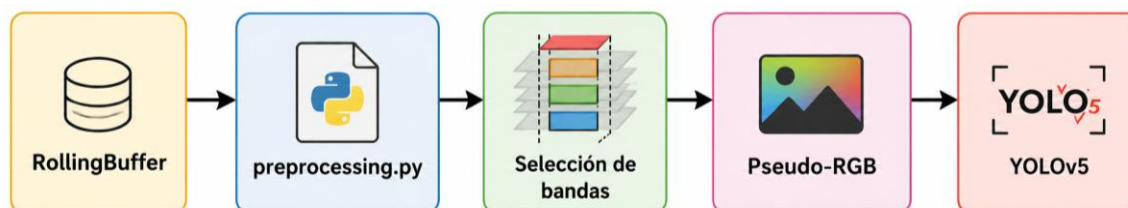
La utilización de un RollingBuffer permitió desacoplar el proceso de adquisición del resto de etapas del sistema, evitando reconstruir completamente la escena ante cada nueva adquisición y reduciendo el costo computacional asociado al procesamiento continuo. De esta manera, mientras el sistema ejecuta la detección, el seguimiento y la clasificación de los frutos presentes en la escena, el proceso de adquisición continúa funcionando de forma ininterrumpida, manteniendo permanentemente actualizada la información espectral disponible para las etapas posteriores del procesamiento.

F.3 Generación del pseudo-RGB y detección de frutos

Una vez reconstruida la escena espectral mediante el RollingBuffer, el sistema selecciona las cinco bandas espectrales definidas durante el proceso de análisis de separabilidad (¡Error! No se encuentra el origen de la referencia.). Antes de generar cualquier representación destinada a la detección o clasificación, estas bandas son procesadas mediante el módulo preprocessing.py, encargado de realizar de forma centralizada la normalización radiométrica utilizada por todo el sistema. La centralización de este procedimiento garantiza que tanto el detector YOLOv5 como el modelo de clasificación reciban información preprocesada bajo un mismo criterio, evitando inconsistencias entre los diferentes módulos de la aplicación.

Posteriormente, tres de las cinco bandas seleccionadas son combinadas para construir una representación pseudo-RGB, utilizada como entrada del modelo YOLOv5. A diferencia de una imagen RGB convencional, cada canal de esta representación corresponde a una longitud de onda específica del rango SWIR, permitiendo conservar el contraste espectral de la escena mientras se mantiene la compatibilidad con arquitecturas de detección originalmente desarrolladas para imágenes de tres canales.

Figura F.3. *Proceso de generación de la representación pseudo-RGB a partir de las cinco bandas espectrales seleccionadas.*



La imagen pseudo-RGB es utilizada como entrada del detector YOLOv5, encargado de localizar automáticamente las mandarinas presentes en la escena mediante regiones delimitadoras

(bounding boxes). Durante esta etapa únicamente se determina la posición espacial de cada fruto, sin realizar aún la clasificación de calidad. Las detecciones aceptadas corresponden únicamente a aquellas que satisfacen los umbrales de confianza e intersección sobre unión (IoU) definidos en el módulo `config.py`, garantizando un criterio uniforme durante toda la operación del sistema.

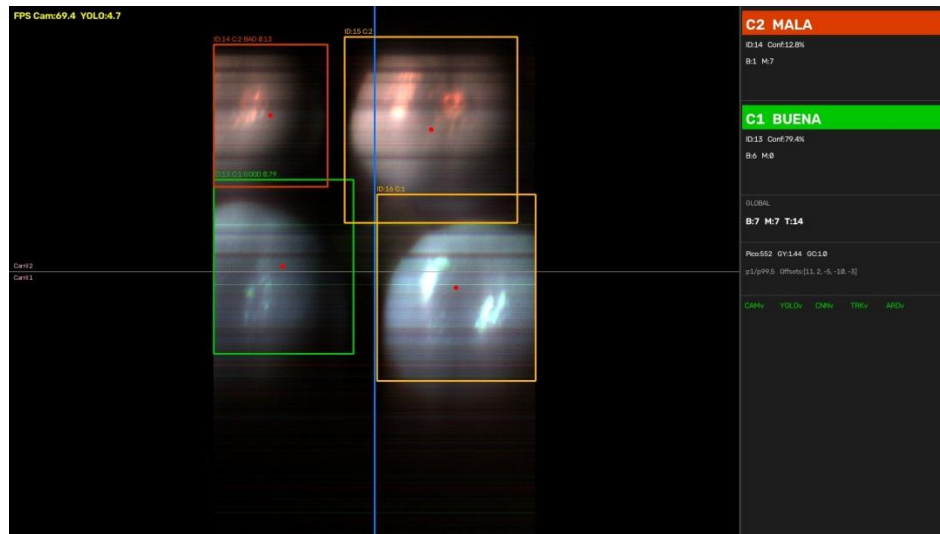
Con el propósito de mantener la coherencia entre las diferentes etapas del procesamiento, la representación pseudo-RGB utilizada para la inferencia corresponde exactamente a la misma imagen visualizada en la interfaz del sistema. De esta manera, las detecciones mostradas al operador reflejan directamente la información procesada por el detector, facilitando la supervisión del funcionamiento del sistema durante la operación en tiempo real.

F.4 Seguimiento, asignación de carriles y evento de clasificación

Una vez obtenidas las detecciones mediante YOLOv5, el sistema inicia el proceso de seguimiento (tracking), cuyo objetivo consiste en mantener la correspondencia entre cada mandarina detectada y su desplazamiento sobre la banda transportadora. Para ello, a cada detección se le asigna un identificador único (ID), el cual permanece asociado al fruto mientras este continúa siendo observado dentro de la región de captura.

El seguimiento permite conocer continuamente la posición del centroide de cada fruto, evitando que una misma mandarina sea procesada repetidamente a medida que avanza sobre la banda. Además de mantener la identidad de las detecciones, este módulo registra el historial de cada objeto y actualiza su posición en cada nueva reconstrucción de la escena, proporcionando la información necesaria para las etapas posteriores de clasificación y sincronización.

Figura F.4. *Proceso de seguimiento e identificación de frutos durante el desplazamiento sobre la banda transportadora.*



Con el fin de establecer el momento exacto en que debe ejecutarse la clasificación, se definió una línea virtual ubicada dentro de la región de captura. Cuando el centroide de un fruto cruza esta línea por primera vez, el sistema genera un evento de clasificación, garantizando que cada mandarina sea procesada una única vez durante su recorrido por la escena. Este criterio evita clasificaciones duplicadas y mantiene la correspondencia entre el fruto detectado y la decisión generada posteriormente por el modelo CNN–Transformer.

Durante este mismo evento se determina el carril virtual al que pertenece cada fruto. La asignación se realiza a partir de la posición horizontal del centroide con respecto a la división definida entre ambas zonas de operación de la banda transportadora. Esta información permanece asociada al identificador del fruto y posteriormente es utilizada por el módulo de planificación temporal para seleccionar el mecanismo de separación que deberá accionarse.

Finalmente, el evento de clasificación almacena la información necesaria para las siguientes etapas del procesamiento, incluyendo el identificador del fruto, el carril asignado, el

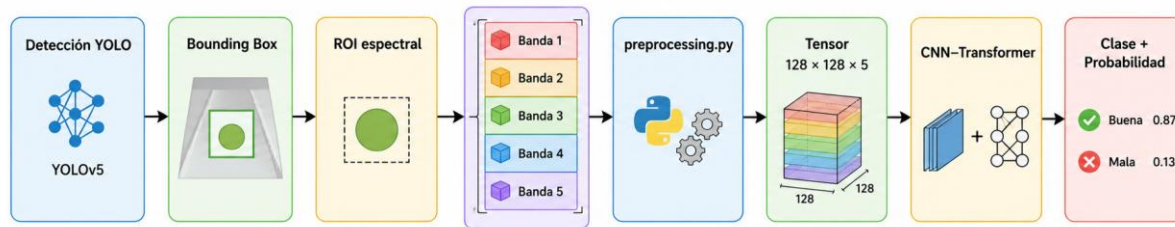
instante de cruce por la línea virtual y la referencia espacial utilizada para extraer posteriormente la región espectral correspondiente. De esta manera, el sistema conserva la trazabilidad completa de cada mandarina desde su detección inicial hasta el accionamiento del mecanismo de separación.

F.5 Extracción del ROI espectral y clasificación

Una vez generado el evento de clasificación, el sistema utiliza la información almacenada por el módulo de seguimiento para localizar la región espectral correspondiente al fruto dentro de la escena reconstruida. A partir de las coordenadas de la detección obtenidas por YOLOv5, se extrae una Región de Interés (Region of Interest, ROI) sobre las cinco bandas espectrales previamente seleccionadas, conservando la correspondencia espacial entre todos los canales utilizados por el modelo de clasificación.

Antes de ejecutar la inferencia, la información espectral extraída es procesada nuevamente mediante el módulo compartido `preprocessing.py`, responsable de aplicar la normalización utilizada por todo el sistema. Al emplear el mismo procedimiento implementado durante la generación del pseudo-RGB, se garantiza que tanto la etapa de detección como la de clasificación trabajen sobre datos procesados bajo un criterio uniforme, evitando diferencias derivadas de procesos de normalización independientes.

Figura F.5. Extracción de la región de interés espectral y construcción del tensor de entrada para el modelo de clasificación.



Posteriormente, la región espectral es redimensionada al tamaño de entrada requerido por el modelo (128×128 píxeles) y organizada como un tensor compuesto por las cinco bandas seleccionadas. Este tensor constituye la entrada del modelo CNN-Transformer, el cual realiza la inferencia y estima la probabilidad de pertenencia a cada una de las categorías definidas durante el entrenamiento.

Como resultado de la inferencia, el sistema determina si el fruto corresponde a una mandarina apta o no apta. Esta decisión queda asociada al identificador único asignado durante el seguimiento, preservando la trazabilidad del fruto a lo largo de todo el proceso. El resultado de clasificación es posteriormente transferido al módulo de planificación temporal, encargado de programar el instante de accionamiento del mecanismo de separación de acuerdo con el carril previamente asignado.

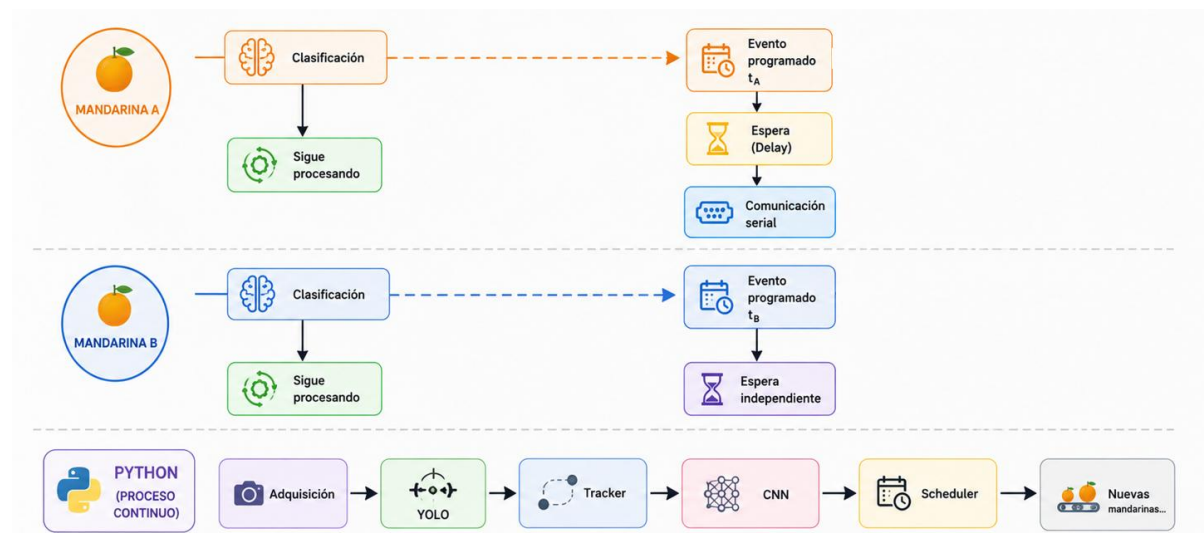
F.6 Planificación temporal no bloqueante

Una vez obtenida la clasificación de un fruto, el sistema no ejecuta inmediatamente el accionamiento del mecanismo de separación. En su lugar, la decisión generada es incorporada a un módulo de planificación temporal (scheduler), responsable de programar el instante en que deberá enviarse la orden correspondiente al sistema de control. Esta estrategia permite desacoplar

el proceso de inferencia del accionamiento físico, manteniendo el funcionamiento continuo del sistema mientras los frutos continúan desplazándose sobre la banda transportadora.

Para cada mandarina clasificada como apta, el sistema genera un evento independiente que almacena el identificador del fruto, el carril asignado durante el seguimiento y el instante programado para ejecutar el accionamiento. El tiempo de espera empleado entre el cruce del centroide por la línea virtual de clasificación y la activación del mecanismo de separación se define mediante el parámetro `CENTER_CROSSING_DELAY_S`, centralizado en el módulo `config.py`, permitiendo ajustar fácilmente la sincronización cuando cambian las condiciones de operación del sistema.

Figura F.6. Flujo de planificación temporal de eventos desde la clasificación hasta el accionamiento.



A diferencia de un esquema secuencial basado en retardos bloqueantes, el planificador implementado mantiene una lista de eventos pendientes y verifica continuamente si alguno ha alcanzado el instante programado para su ejecución. Mientras los eventos permanecen en espera, el sistema continúa realizando la adquisición de nuevas líneas espectrales, la reconstrucción de la

escena, la detección de nuevos frutos, el seguimiento y la clasificación, evitando interrupciones en el flujo principal de procesamiento.

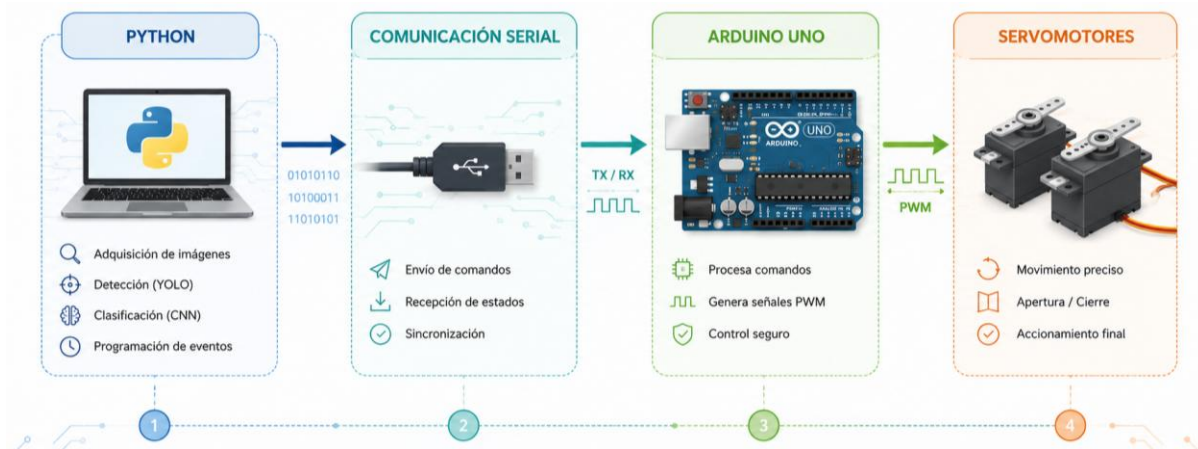
Esta estrategia permite gestionar simultáneamente múltiples frutos desplazándose por la banda transportadora, ya que cada evento es tratado de manera independiente. Como resultado, diferentes mandarinas pueden encontrarse en distintas etapas del procesamiento adquisición, detección, clasificación o espera de accionamiento sin interferir entre sí, garantizando la continuidad del sistema incluso cuando existen varios frutos presentes de forma simultánea.

Finalmente, cuando el tiempo programado para un evento es alcanzado, el planificador genera la orden correspondiente y la envía al módulo de comunicación serial, responsable de transmitirla al Arduino UNO para ejecutar el movimiento del servomotor asociado al carril previamente asignado.

F.7 Comunicación serial y accionamiento

Una vez que el módulo de planificación temporal determina que un evento debe ejecutarse, la aplicación principal envía la orden correspondiente al Arduino UNO mediante comunicación serial. El protocolo implementado transmite únicamente la información necesaria para identificar el carril y la acción asociada al fruto clasificado, permitiendo mantener una comunicación simple y eficiente entre el computador y el sistema de control de la Figura F7

Figura F.7. Flujo de comunicación entre la aplicación principal y el sistema de accionamiento

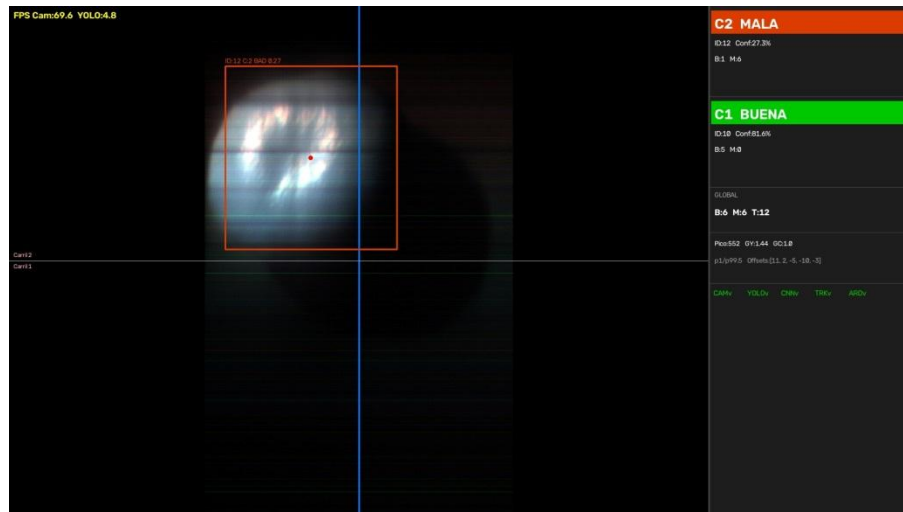


El Arduino interpreta la orden recibida y ejecuta el movimiento del servomotor correspondiente. Para garantizar un funcionamiento continuo, el firmware fue desarrollado mediante una máquina de estados basada en `millis()`, evitando retardos bloqueantes y permitiendo atender nuevas órdenes mientras se completa el accionamiento de las compuertas. De esta manera, la lógica de decisión permanece centralizada en la aplicación desarrollada en Python, mientras que el microcontrolador actúa exclusivamente como encargado del control de los actuadores.

F.8 Interfaz y registro de operación del sistema

Con el propósito de supervisar el funcionamiento del sistema durante las pruebas experimentales, se desarrolló una interfaz gráfica que integra en una única ventana la información generada durante las etapas de adquisición, detección, seguimiento, clasificación y accionamiento. Esta interfaz permite visualizar en tiempo real la representación pseudo-RGB reconstruida, las detecciones realizadas por YOLOv5, el identificador asignado a cada fruto, la clasificación obtenida y el estado general del proceso, facilitando la verificación del funcionamiento del sistema durante su operación.

Figura F.8. Interfaz desarrollada para la supervisión del sistema durante la operación en tiempo real.



Además de la visualización en tiempo real, la aplicación registra la información asociada a cada fruto procesado, incluyendo el identificador asignado, el carril correspondiente, el resultado de la clasificación y la ejecución del evento de accionamiento. Este registro permitió verificar la correspondencia entre las decisiones generadas por los modelos de inteligencia artificial y el comportamiento observado durante las pruebas experimentales, facilitando las labores de depuración y validación del sistema integrado.