Reporte de Laboratorio 2
Conteo de Frutos Pequeños en Tiempo Real Utilizando una Plataforma Giratoria

Miguel Angel Hernandez Ramirez*, Alexis Guadalupe Medina Ayala* Juan Ricardo Soriano Roel*,
Aris Uriel Torres Desilos*, Ricardo Emmanuel Uriegas Ibarra*
*Ingeniería en Tecnologías de la Información
Universidad Politécnica de Victoria

Resumen—

I. Introducción

La disponibilidad de metodologías prácticas y fiables de detección de frutos en campo resulta fundamental para realizar previsiones precisas de la cosecha[8].

II. Desarrollo Experimental

En este trabajo se presenta un sistema de detección en tiempo real de frutos pequeños que giran en una plataforma giratoria.

Para lograr esto se necesita conocimiento en las siguientes áreas:

- Procesamiento de Imágenes
- Visión por Computadora
- Redes Neuronales

La mayor parte de estos conocimientos los hemos aprendido en unidades previas de la materia, por lo que no se abordarán en este reporte.

II-A. Dataset

Antes de todo debemos definir qué es un dataset y por qué lo ocupamos en este trabajo. Un dataset es un conjunto de datos que se utilizan para entrenar un modelo de machine learning. En este caso, el dataset que se utilizó fue totalmente recolectado por nosotros mismos; de algunos frutos variados,[1] el cual consiste en imágenes de frutos chicos en diferentes entornos; esto porque se creía que el proyecto debía abarcar todo tipo de frutos. El dataset se compone de 1 sola clase, la cual es la de fruto. La cantidad de imágenes recolectada fue de 38 imágenes, pero realizando aumentación de datos se logró llegar a 114 imágenes recolectadas por el equipo.

Las técnicas de augmentation que se aplicaron a los datos recolectados fueron los siguientes:

- Flip: Horizontal, Vertical
- Rotación 90°: En sentido horario, antihorario, al revés
- Crop: 0 % mínimo, 20 % máximo
- Rotación: Entre -15° y +15°
- Shear: ±10° Horizontal, ±10° Vertical
- Hue: Entre -15° y +15°
- Saturación: Entre -25 % y +25 %
- Brillo: Entre -15 % y +15 %
- Exposición: Entre -10 % y +10 %
- Blur: Hasta 2.5px
- Ruido: Hasta 0.1 % de píxeles

II-B. Modelo

Debido a la baja cantidad de imágenes para entrenamiento que poseíamos, sumado a la eficiencia que debía de poseer el modelo para ser ejecutado en tiempo real, se opto por usar el modelo pre-entrenando para detección de objetos de la librería ultralytics[2] yolov8n[3].

II-C. Entrenamiento

Una vez ya se tenía la selección del modelo preentrenado a usar, se prosiguió con el entrenamiento del modelo dado el dataset mencionado anteriormente[1].

Antes del entrenamiento se dividió el dataset en 3 partes:

- 80 % para entrenamiento
- 10 % para validación
- 10 % para pruebas

Posteriormente se descargó el modelo yolov8n[3] y para el entrenamiento se realizaron 40 épocas con un tamaño de imagen de 640x640, todo lo demás se dejó en los valores por defecto del modelo (realmente se intentó con diferentes combinaciones de parámetros, pero estos fueron los que mejor resultados dieron a la hora de probarlo en tiempo real).

II-D. Uso del Modelo

Para usar el modelo en el código se agregó un archivo (UI.py) que, haciendo uso de una interfaz gráfica en Qt6, carga el modelo entrenado y realiza la detección de los frutos en tiempo real mediante la cámara de la computadora, con algunos parámetros que se pueden configurar:

- **Confianza:** Umbral de confianza para la detección de objetos.
- **Velocidad de Procesamiento:** Velocidad de procesamiento con la que se realiza la detección.
- **Reinicio de Conteo:** Boton para reiniciar el conteo de frutos.
- **Reinicio de**

Este codigo principalmente busca el modelo best1.onnx, debido a que en un principio se buscaba entrenar y guardar el modelo en este formato, pero sin embargo al final se opto por guardar el modelo en formato .pt debido a que se tenia problemas al cargar el modelo en formato onnxn en la interfaz gráfica en una de las computadoras.

III. Resultados

Al ejecutar UI.py aparecera la interfaz que se muestra en la figura 1. Al dar click al boton de Ïniciar Deteccion"podra observar lo que mira la camara y detecta el modelo en tiempo real2.

III-A. Resultados del modelo

En las figuras 3 a 6 se muestran ejemplos de la detección de distintos frutos diferentes a la fresa en tiempo real en una version de UI.py previa. Sin embargo en las figuras 7 y 8 se puede observar al modelo contabilizando los frutos detectados en una planta de fresa que esta dando vueltas sobre una base giratoria.

El modelo tiene una capacidad limitada debido a la cantidad de datos de entrenamiento usados, junto a la mezcla de imagenes fuera de la fresa debido a la confusion del comienzo del proyecto. Sin embargo el modelo es capaz de detectar frutos de colores llamativos como el rojo y el naranja.

III-A1. Precisión y Recall: Como se observa en las figuras 9 (Curva Precisión-Confianza) y 10 (Curva Recall-Confianza), el modelo presenta las siguientes características:

- **Precisión máxima:** Alcanza un valor perfecto (1.00) con un umbral de confianza de 0.583 (Figura 9), indicando que cuando el modelo está seguro de sus predicciones (58.3 % de confianza), todas las detecciones son correctas.
- **Recall base:** Muestra un 56 % de detección efectiva con un umbral de confianza 0 (Figura 10), lo cual quiere decir que sin filtrar por confianza, el modelo detecta poco más de la mitad de los frutos presentes.

III-A2. Limitaciones y desempeño: El modelo presenta una capacidad de generalización limitada debido a:

- Cantidad baja de datos para el entrenamiento
- Mezcla de imágenes que no son fresas debido a la confusion inicial del proyecto
- Sesgo hacia colores llamativos (rojo especialmente) visible en las falsas detecciones

Estas limitaciones se manifiestan en:

- Caída abrupta del recall con aumento de confianza (Figura 10)
- Discrepancia entre precisión máxima teórica (1.00) y desempeño práctico observado en las figuras 7-8
- Dificultad para detectar frutos en posiciones e iluminaciones peculiares

III-A3. Conclusiones del rendimiento: El análisis cuantitativo revela que el modelo:

- Funciona mejor con umbrales altos de confianza (¿0.5)
- Pierde capacidad de detección (recall) al exigir alta precisión
- Requiere aumentar la diversidad del dataset para mejorar cobertura
- Sería beneficiado por técnicas de aumento de datos para colores menos saturados

III-B. Debilidades del Modelo

El modelo es realmente capaz de detectar frutos coloridos, mas no es tan bueno detectando frutos de colores poco vistosos como el verde, de igual manera, uno de los problemas más grandes del proyecto es detectar un fruto que se encuentra en la parte trasera al estar girando. Como se puede ver en la figura 8 el modelo contabiliza nuevamente el fruto que ya había contado anteriormente, esto debido a que el pino tapa por completo la visibilidad del modelo, por lo que no puede identificar que pasó por la parte trasera.

Figura 1: Interfaz gráfica.

IV. Conclusión

En este trabajo se propuso una solución al problema de conteo de frutos en tiempo real, utilizando una plataforma giratoria. Se logró implementar un modelo de detección de objetos en tiempo real, que es capaz de detectar frutos en movimiento, sin embargo, el modelo presenta limitaciones en la detección de frutos de colores menos llamativos y en la detección de frutos en posiciones e iluminaciones peculiares, como lo son el fondo de una imagen.

Referencias

[1] *Proyecto Little Fruit*. https://universe.roboflow.com/littlefruitproject/little-fruit/dataset/3. Consultado el 19-03-2025.

[2] *Ultralytics*. https://pypi.org/project/ultralytics/. Consultado el 19-03-2025.

[3] *YOLOv8*. https://docs.ultralytics.com/es/models/yolov8/. Consultado el 19-03-2025.

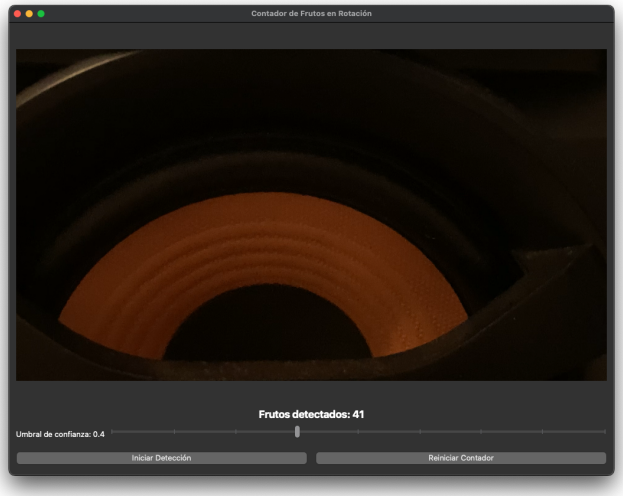

Figura 2: Interfaz gráfica ejecutando la detección de frutos en tiempo real.

[4] *Detección de Frutas con YOLOv4*. https://github.com/EricKiennemann/Fruit-Detection-with-Yolov4?tab=readme-ov-file. Consultado el 19-03-2025.

[5] *Roboflow*. https://roboflow.com. Consultado el 19-03-2025.

[6] *Albumentations*. https://pypi.org/project/albumentations/. Consultado el 19-03-2025.

[7] *PyTorch*. https://pypi.org/project/pytorch/. Consultado el 19-03-2025.

[8] *Detección de Frutos y Estimación de la Cosecha*. https://www.grap.udl.cat/es/recerca/linies-de-recerca/deteccio-de-fruits-i-estimacio-de-collita/. Consultado el 19-03-2025.

[9] *Proyecto Strawberry Detection*. https://universe.roboflow.com/strawberry-vheh2/strawberry-detection-qylms/fork. Consultado el 19-03-2025.

[10] Ian Goodfellow, Yoshua Bengio y Aaron Courville. *Deep Learning*. Cambridge, MA: MIT Press, 2016.

[11] Richard Szeliski. *Computer Vision: Algorithms and Applications*. London: Springer, 2010.

[12] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. New York: Springer, 2006.

[13] Kevin P. Murphy. *Machine Learning: A Probabilistic Perspective*. Cambridge, MA: MIT Press, 2012.

[14] Gary Bradski y Adrian Kaehler. *Learning OpenCV: Computer Vision with the OpenCV Library*. Sebastopol, CA: O'Reilly Media, 2008.

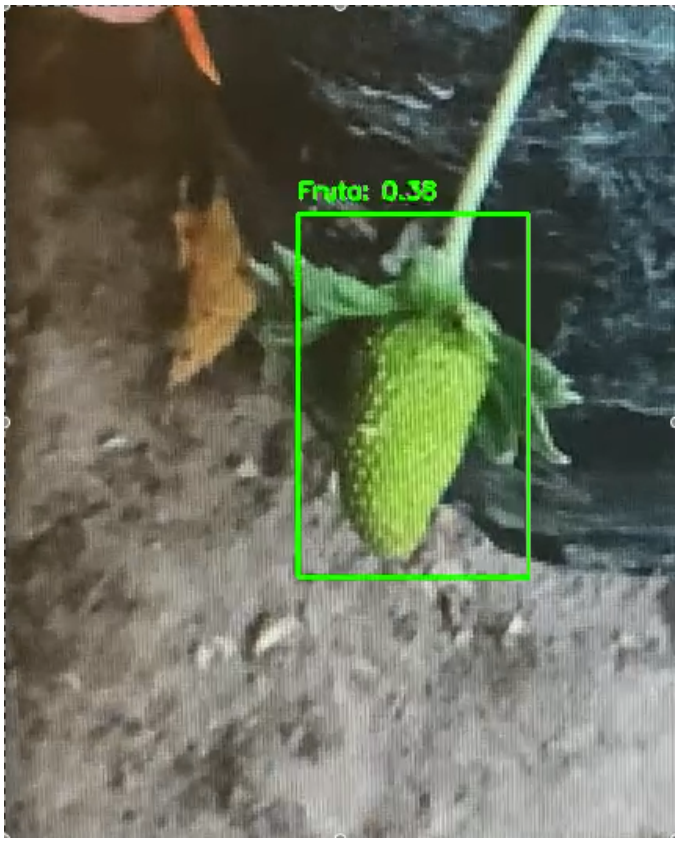

Figura 3: Imagen ejemplificando la detección de frutos en tiempo real.

Figura 4: Imagen ejemplificando la detección de frutos en tiempo real.

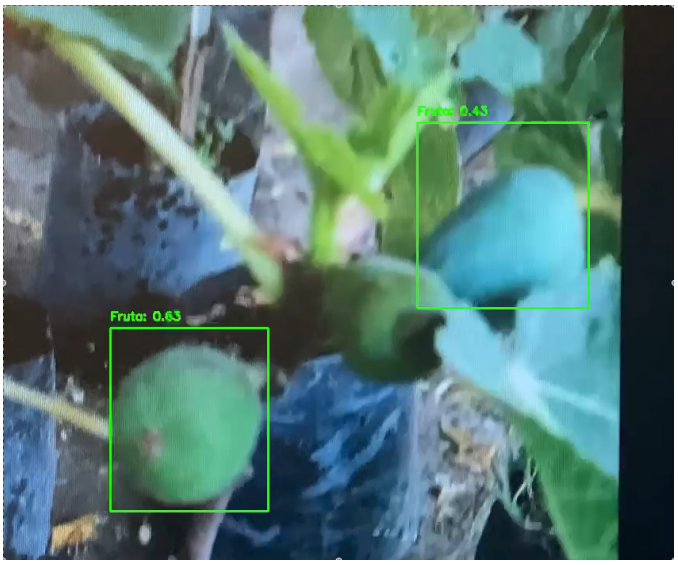

Figura 5: Imagen ejemplificando la detección de frutos en tiempo real.

Figura 6: Imagen ejemplificando la detección de frutos en tiempo real.

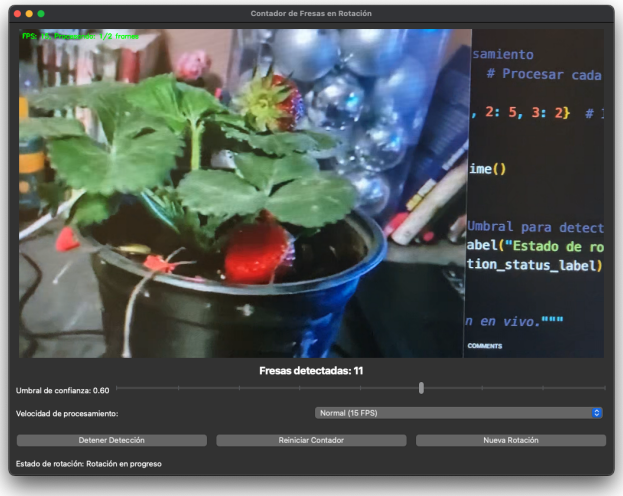

Figura 7: Imagen ejemplificando la detección de frutos en tiempo real en una planta de fresa.

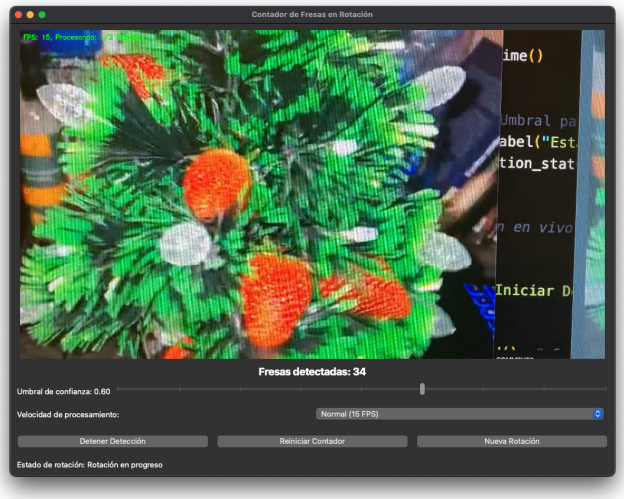

Figura 8: Imagen ejemplificando la detección de frutos en tiempo real en una planta de fresa.

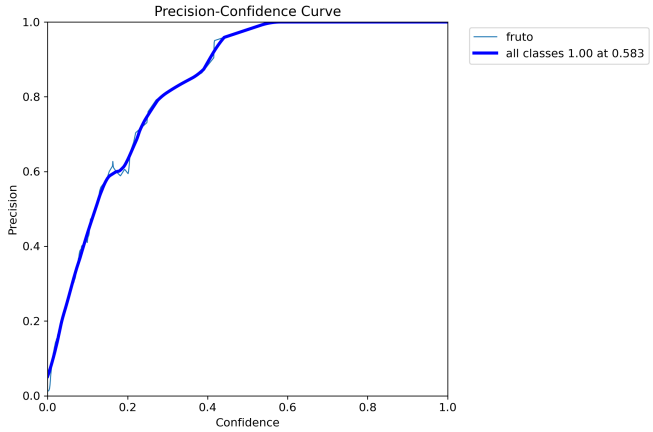

Figura 9: Curva de Precisión.

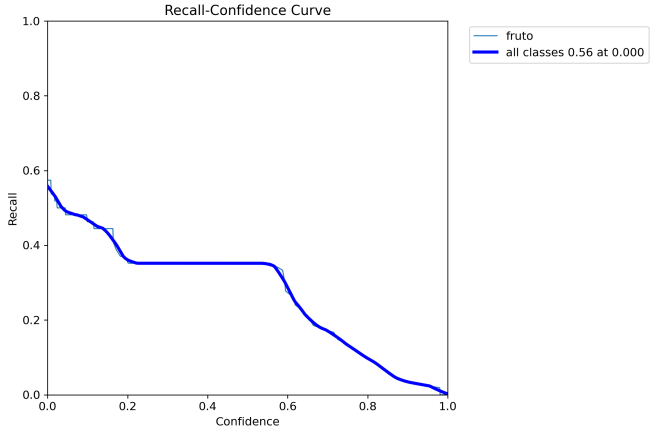

Figura 10: Curva de Recall.

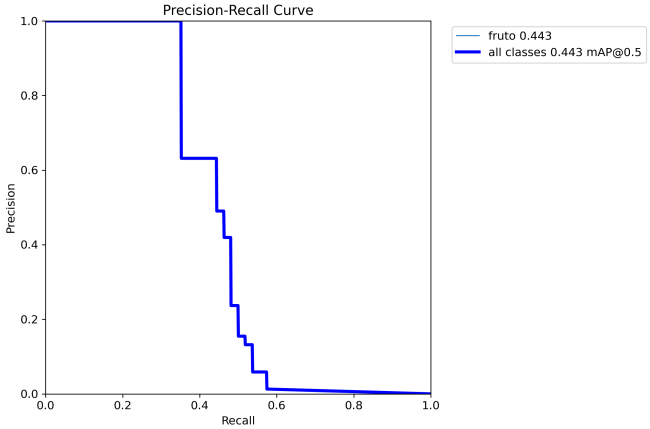

Figura 11: Curva de Precisión y Recall.