

# Detect-STR: Avance de artículo (formato IEEE)

Paulina Cruz Álvarez, Perla Edith Lugo Hernández, Enrique Escalera Jiménez  
Universidad Autónoma del Estado de México  
Facultad de ingeniería  
Reconocimiento de patrones

**Resumen**—Este avance presenta en detalle la fase de preprocesamiento del proyecto Detect-STR, cuyo objetivo es identificar y contar fresas maduras e inmaduras a partir de fotografías. Se describe el software desarrollado (una versión en Python que replica y amplía un proyecto PID originalmente implementado en MATLAB), las decisiones de diseño, las transformaciones de imágenes aplicadas, y se acompaña con código y una libreta Jupyter reproducible.

**Index Terms**—detección de fruta, preprocesamiento de imágenes, OpenCV, SLIC, KMeans, segmentación, Python.

## I. INTRODUCCIÓN

La detección automática de fruta madura es una tarea de visión por computadora con aplicaciones en agricultura de precisión, automatización de cosecha y control de calidad. En Detect-STR se desarrolló un pipeline de preprocesamiento de imágenes para facilitar la etapa de segmentación y conteo. Esta fase es crítica: la calidad del preprocesado impacta directamente en la precisión de los detectores y clasificadores posteriores.

## II. CONTEXTO Y SOFTWARE

El software utilizado en esta fase es una reimplementación y extensión en Python de un proyecto previo desarrollado en MATLAB bajo el paradigma PID (Proporcional-Integral-Derivativo) aplicado originalmente a operaciones de imagen. La implementación Python mantiene la interfaz y la semántica original (métodos que modifican la imagen interna y devuelven resultados), y añade módulos para comandos específicos de fresas y un parser seguro para describir pipelines en formato de script.

Librerías principales: OpenCV, scikit-image, SciPy, scikit-learn, pandas y numpy. Utilidades principales incluidas como anexos: `proyecto.py`, `strawberry_commands.py`

## III. DISEÑO DEL SOFTWARE (CLASE PROYECTO Y PARSE)

La clase Proyecto replica la estructura funcional del código MATLAB: mantiene un estado interno y ofrece métodos atómicos que aplican transformaciones (ecualización, filtros, morfología, segmentación). Cada método actualiza el estado interno y retorna la imagen resultante o un diccionario con salidas. El ScriptParser permite describir pipelines en lenguaje simple y ejecutarlos

de manera segura usando el módulo `ast` para validar nodos permitidos.

## IV. ESTRATEGIA DE PREPROCESAMIENTO PARA DETECCIÓN DE FRESAS MADURAS E INMADURAS

El objetivo es transformar imágenes crudas en representaciones donde las fresas maduras e inmaduras sean distinguibles y contables. A continuación se resumen los pasos típicos del pipeline:

### IV-A. Lectura y normalización de imágenes

Leer con OpenCV, asegurar tipo `uint8` en rango `0..255` y, opcionalmente, normalizar dimensiones por redimensionado para balancear resolución y performance.

### IV-B. Conversión de espacio de color y máscaras HSV

Usar HSV para separar colores. El módulo contiene funciones `ToHSV` y `MaskHSV`. Rango típico (a calibrar): maduras (rojo) `H=0-10` o `160-179`; inmaduras `H` alrededor `35-85`.

### IV-C. Limpieza de máscara y filtrado de componentes

Operaciones morfológicas (`erode/dilate`) y filtrado por área (`CCFilter`) para eliminar componentes pequeños y reducir falsos positivos.

### IV-D. Segmentación y separación de objetos cercanos

Técnicas complementarias: KMeans (`imsegkmeans`), SLIC, y operaciones de watershed o separación basada en distancias para mejorar conteo en racimos.

### IV-E. Caracterización y extracción de features

Por cada componente conectado se extraen: área, perímetro, bounding box, centroid, color medio, aspect ratio, convexidad, etc. Estas features pueden alimentar clasificadores.

### IV-F. Preprocesamiento tabular y metadatos

El fichero `anexos/summary.csv` contiene metadatos por imagen. Se aplicaron imputación de faltantes, normalización/estandarización y creación de ratios como `ripe_ratio`.

#### IV-G. Aumentos y variabilidad

Aumentos recomendados: rotaciones, flips, cambios de brillo/contraste y recortes. Aplicar sólo en entrenamiento.

### V. TÉCNICAS DE PREPROCESAMIENTO

#### V-A. Manejo de valores faltantes

Se aplicó imputación por mediana en contadores y placeholder `unknown`.<sup>en</sup> categóricos. Divisiones por cero en ratios se resuelven definiendo el ratio como cero y registrándolo explícitamente.

#### V-B. Normalización

Normalización de píxeles (división por 255.0) y min-max en descriptores tabulares, guardando parámetros sobre el conjunto de entrenamiento.

#### V-C. Estandarización

Estandarización z-score para features numéricos que lo requieren; parámetros y versionados.

#### V-D. Codificación de categóricas

Etiquetado binario, one-hot para nominales y agrupación de niveles infrecuentes bajo `"other"`.

### VI. RESULTADOS PRELIMINARES

Se incluyen ejemplos de ejecución y visualizaciones en la libreta Jupyter. La detección es razonable con iluminación controlada; condiciones adversas requieren recalibración de rangos HSV.

### VII. CONCLUSIONES Y TRABAJO FUTURO

Se presentó la reimplementación en Python del pipeline MATLAB, con parser seguro y comandos especializados. Trabajo futuro: mejorar separación en racimos, entrenar clasificadores y extender dataset.

### APÉNDICE

Los módulos y ficheros auxiliares incluidos en el repositorio son: `proyecto.py`, `strawberry_commands.py`,

### REFERENCIAS

### REFERENCIAS

- [1] E. Escalera Jiménez, *RecPat\_2025\_Fresas*, GitHub repository, 2025.
- [2] G. Bradski and A. Kaehler, *Learning OpenCV: Computer Vision with the OpenCV Library*, O'Reilly Media, 2008.
- [3] S. van der Walt et al., "scikit-image: Image processing in Python," *PeerJ*, 2014.
- [4] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," *J. Mach. Learn. Res.*, 2011.
- [5] R. Achanta et al., "SLIC Superpixels Compared to State-of-the-Art," 2012.
- [6] L. Vincent and P. Soille, "Watersheds in digital spaces," *IEEE Trans. PAMI*, 1991.