UNIVERSIDAD DISTRITAL
FRANCISCO JOSÉ DE CALDAS

Revista Vínculos

https://revistas.udistrital.edu.co/index.php/vinculos/

Recibido: 4 de febrero 2026 / Aceptado: 18 de mayo 2026

ANOTACIÓN YOLO DE OBJETOS PEQUEÑOS EN ORTOFOTOS.

YOLO ANNOTATION OF SMALL OBJECTS IN ORTHOPHOTOS.

Damian-Montiel Uriel[1] , Rocío N Ramos-Bernal[2], Alarcón Paredes- Antonio[3], Vázquez-Jiménez Rene[4] , Alonso-Silverio Gustavo[5]

Resumen:

Este trabajo presenta una herramienta manual para la anotación en formato de anotación YOLO de objetos pequeños en ortofotos de alta resolución. La solución se implementó en Python con arquitectura modular, teselado dinámico para navegación fluida y persistencia inmediata de etiquetas (.txt homónimo por imagen). Su desempeño se validó en equipos con Windows 10/11 de gama media, y mediante una prueba práctica con 10 estudiantes que anotaron tapas de registro y marcas viales en campañas de dos jornadas. Los resultados evidencian un tiempo de inicio de la aplicación inferior a 3 segundos desde que el usuario ejecuta el archivo hasta que la interfaz gráfica está lista para trabajar, junto con estabilidad sin cierres inesperados y una experiencia de uso positiva (zoom fluido, líneas guía, deshacer), favoreciendo la consistencia del conjunto de entrenamiento. La herramienta reduce la fricción operativa respecto a anotadores genéricos al centrarse en el flujo geoespacial con ortofotos grandes y producir salidas compatibles con YOLO, lo cual agiliza la construcción de datasets para tareas de detección en infraestructura urbana.

Palabras claves: Ortofotos, herramienta de anotación, Etiquetas YOLO, YOLO, UAV.

Abstract

This work presents a manual tool for YOLO-format annotation of small objects in highresolution orthophotos. The solution was implemented in Python with a modular architecture, dynamic tiling for smooth navigation, and immediate label persistence (per-image homonymous .txt). Its performance was validated on mid-range Windows 10/11 workstations and through a practical test with 10 students who annotated manhole covers and road markings over two field-day sessions. Results show an application startup time of less than 3 seconds from execution to a fully operational graphical interface, as well as stable operation without unexpected crashes and a positive user experience (fluid zoom, crosshair guides, undo), contributing to consistent training datasets. The tool reduces operational friction compared to generic annotators by focusing on geospatial workflows with large orthophotos and producing YOLO-compatible outputs, thereby streamlining the creation of datasets for object detection in urban infrastructure applications.

Keywords: Orthophotos, annotation tool, YOLO labels, YOLO, UAV

1. Introducción

La generación de conjuntos de datos de alta calidad para visión por computadora sigue dependiendo, en gran medida, de procesos de anotación manual que son costosos en tiempo y propensos a errores, aun cuando existe una amplia oferta de herramientas de etiquetado [1]. En infraestructura urbana, trabajos recientes muestran que la identificación de tapas de registro de drenaje para entrenar detectores sigue requiriendo etiquetado manual previo, tanto en escenas de calle como en ortofotos capturadas con vuelos no tripulados (UAV por sus siglas en ingles)[2]–[4].

YOLO (You Only Look Once) es una familia de detectores de una sola etapa (one-stage) que predicen directamente, en un único paso, las cajas delimitadoras y las probabilidades de clase, lo que posibilita inferencia eficiente y la adopción amplia de su formato de anotación por simplicidad y compatibilidad en la comunidad [12], [13]. En consecuencia, muchos flujos de trabajo de detección urbana se apoyan en conjuntos de datos anotados en formato de anotación YOLO.

Pese a la variedad de opciones, persisten limitaciones técnicas y prácticas al trabajar con ortofotos de alta resolución y flujos centrados en un dominio específico (p. ej., tapas de drenaje). Herramientas ligeras como Labellmg ofrecen un flujo simple de cuadros delimitadores y compatibilidad con formatos comunes (YOLO, PASCAL VOC), pero carecen de funcionalidades especializadas para metadatos geoespaciales o navegación fluida sobre imágenes muy grandes [5], [6]. Por su parte, VGG Image Annotator (VIA) prioriza la portabilidad y la anotación manual con exportación a CSV/JSON, pero no contempla de origen la manipulación eficiente de ortofotos multigigapíxel [7].

[1] Maestría en Ingeniería para la Innovación y Desarrollo Tecnológico, Facultad de Ingeniería, Universidad Autónoma de Guerrero, México. Correo electrónico: 15488831@uagro.mx
[2] Maestría en Ingeniería para la Innovación y Desarrollo Tecnológico, Facultad de Ingeniería, Universidad Autónoma de Guerrero, México. Research Group on Technologies for Landscape Analysis and Diagnosis (TADAT), Department of Chemical and Environmental Technology, ESCET, Universidad Rey Juan Carlos, Madrid, España. Correo electrónico: rnramos@uagro.mx
[3] Maestría en Ingeniería para la Innovación y Desarrollo Tecnológico, Facultad de Ingeniería, Universidad Autónoma de Guerrero, México. Correo electrónico: aalarcon@uagro.mx

ISSN 1794-211 X • e-ISSN 2322-939X • Vol 20, N° 2 (julio-diciembre 2023). pp. 91-98. Universidad Distrital Francisco José de Caldas-Facultad Tecnológica.

[4] Maestría en Ingeniería para la Innovación y Desarrollo Tecnológico, Facultad de Ingeniería, Universidad Autónoma de Guerrero, México. Research Group on Technologies for Landscape Analysis and Diagnosis (TADAT), Department of Chemical and Environmental Technology, ESCET, Universidad Rey Juan Carlos, Madrid, España. Correo electrónico: rvazquez@uagro.mx

[5] Maestría en Ingeniería para la Innovación y Desarrollo Tecnológico, Facultad de Ingeniería, Universidad Autónoma de Guerrero, México. Correo electrónico: gsilverio@uagro.mx

Las plataformas colaborativas de mayor alcance, como CVAT, amplían el tipo de tareas (detección, segmentación, puntos clave) y el trabajo en equipo, pero su despliegue suele requerir instalación y mantenimiento mediante Docker, lo que introduce una barrera operativa para equipos sin soporte de TI o sin servidores dedicados [8]. En el extremo comercial, soluciones como Labelbox o SuperAnnotate incluyen ecosistemas para administración de datos y etiquetado a escala, pero operan con planes de pago y funciones avanzadas sujetas a suscripciones o niveles tarifarios, lo que puede ser restrictivo para grupos académicos con presupuestos limitados [9], [10]. Inclusive anotadores webs gratuitos como MakeSense.ai reportan limitaciones prácticas al manejar tareas grandes o al guardar el progreso, afectando la continuidad del trabajo [11].

La propuesta se desarrolló a partir de tres líneas complementarias: (i) el diseño e implementación de una arquitectura modular con teselado dinámico y persistencia inmediata de etiquetas YOLO; (ii) el empaquetado como ejecutable autónomo para evitar dependencias externas; y (iii) una validación funcional que incluyó pruebas controladas en hardware de gama media y un ensayo con 10 estudiantes, quienes etiquetaron objetos urbanos sobre ortofotos. Estas acciones permitieron evaluar la utilidad de la herramienta en escenarios reales de etiquetado manual con requerimientos específicos de rendimiento y continuidad.

De esta manera se aborda ese vacío con una herramienta manual, de uso local y gratuito, orientada a la anotación en formato de anotación YOLO directamente sobre ortofotos. En concreto, resuelve problemas prácticos que hemos observado en anotadores genéricos:

(i) Manejo estable de ortofotos pesadas sin cierres inesperados gracias a un teselado dinámico que soloprocesa la porción visible, manteniendo desplazamiento y zoom fluidos;

(ii) Prevención de pérdida de avance mediante persistencia inmediata de etiquetas en archivoshomónimos por imagen;

(iii) Reducción de fricción operativa al prescindir de servidores, despliegues complejos o costes desuscripción;

Con este diseño centrado en el caso de objetos pequeños, el proceso de anotación mantiene la fluidez en ortofotos de gran tamaño y la integridad del trabajo, mitigando las limitaciones reportadas en herramientas de propósito general [5]–[11].

2. Metodología

2.1. Materiales e insumos

Hardware utilizado

Para el desarrollo y la validación se emplearon estaciones de trabajo de gama media con las especificaciones mínimas descritas en la Tabla 1.

Componente	Especificación
CPU	Intel Core i5-8250U (1.6 GHz, 4 c/8 h) o equivalente
GPU	Integrada Intel UHD 620 (sin aceleración CUDA)
RAM	16 GB DDR4
Almacenamiento	SSD SATA de 256 GB
Monitor	1920 × 1080 px, 60 Hz

Tabla 1. Componentes del hardware usado

Estas características garantizan que la aplicación pueda ejecutarse sin cuellos de botella de memoria ni latencias perceptibles en el renderizado de teselas de alta resolución.

Software empleado

- **Sistema operativo:** Windows 10 (22H2) y Windows 11 (23H2).
- **Intérprete:** Python 3.12.
- **Bibliotecas:**

o Tkinter (interfaces gráficas, incluido en la distribución estándar). o Pillow 10.3.0 (conversión de matrices a objetos gráficos). o OpenCV-Python 4.10 (manipulación de imágenes y redimensionado). o NumPy 1.26 (gestión de arreglos).

o PyInstaller 6.5 (empaquetado).

- **Control de versiones:** Git 2.45; repositorio privado en GitHub.
- **Editor / IDE:** VS Code 1.90 con extensión Python.

2.2. Análisis de requisitos

Para el desarrollo de esta herramienta, se han identificado los siguientes requisitos fundamentales: la capacidad de cargar y visualizar ortofotos georreferenciadas, permitiendo al usuario un desplazamiento fluido por la imagen. Asimismo, se requiere la implementación de una interfaz intuitiva para la anotación de objetos y la generación de archivos de

ISSN 1794-211 X • e-ISSN 2322-939X • Vol 20, N° 2 (julio-diciembre 2023). pp. 91-98. Universidad Distrital Francisco José de Caldas-Facultad Tecnológica.

etiquetas en formato de anotación YOLO. Es crucial que la herramienta ofrezca facilidad de uso mediante un ejecutable independiente. Además, debe ser capaz de manejar diversos tamaños de ortofotos y permitir la creación y eliminación eficiente de anotaciones.

Finalmente, es esencial que proporcione mecanismos de gestión de clases de objetos y la asignación de etiquetas consistentes, asegurando la calidad de los datos para el entrenamiento.

Con el fin de garantizar la interoperabilidad con detectores ampliamente utilizados, se adopta el formato de anotación YOLO para detección por cajas delimitadoras. En este esquema, cada objeto de una imagen se describe en una línea del archivo de etiquetas homónimo (mismo nombre que la imagen y extensión.txt), con cinco campos separados por espacios:

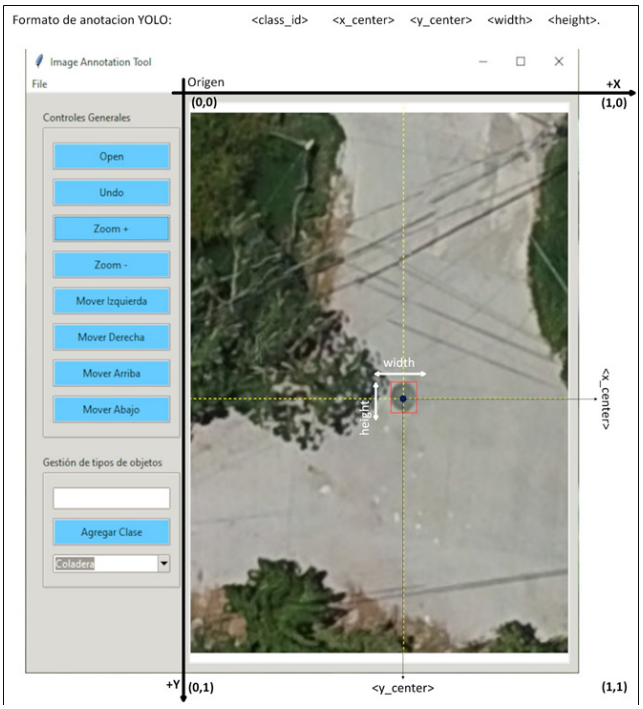

Figura 1. Esquema del formato de anotación YOLO para cajas delimitadoras.

Los cuatro valores geométricos son coordenadas normalizadas en [0, 1] respecto al ancho (W) y alto (H) de la imagen: x_center y y_center indican el centro de la caja, mientras que width y height son su ancho y alto relativos como se muestra en la Figura 1. El identificador class_id es un entero que inicia en 0 y corresponde al orden de clases definido en la sesión. Esta codificación, independiente de la resolución, facilita el entrenamiento y la reutilización del dataset en distintos modelos de la familia YOLO [12], [13].

Conversión de píxeles a valores normalizados.

Si la caja en píxeles está dada por $(x_{min}, y_{min}, x_{max}, y_{max})$ y la imagen tiene dimensiones (W,H)(W, H)(W,H):

- $x_{center} = \frac{(x_{min} + y_{max})/2}{W}$ (1)
- $y_{center} = \frac{(y_{min} + x_{min})/2}{H}$ (2)
- $width = \frac{x_{max} - x_{min}}{W}$ (3)
- $height = \frac{y_{max} - y_{min}}{H}$ (4)

2.3. Arquitectura general

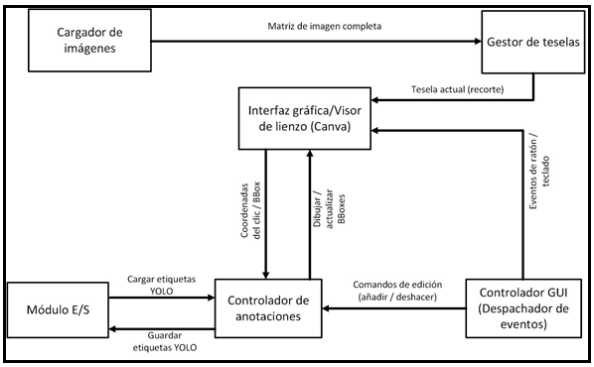

Figura 2. Visión General de la Arquitectura de la Herramienta de anotación de etiquetas en formato de anotación YOLO.

La arquitectura de la herramienta se ilustra en la Figura 2 y adopta un enfoque modular y desacoplado, facilitando su extensibilidad (por ejemplo, al agregar nuevas funcionalidades como exportar a otros formatos) y simplificando las tareas de mantenimiento, tales como corregir errores específicos, actualizar bibliotecas o adaptar módulos para nuevos sistemas operativos. El flujo principal inicia con el cargador de imágenes, responsable de abrir el archivo original y exponerlo como matriz NumPy; este módulo delega en el gestor de teselas el cálculo dinámico del recorte visible (crop), denominado tesela, que corresponde exactamente al área visible de la ortofoto en pantalla, ajustándose según el zoom y el desplazamiento del usuario. Este método evita penalizar el consumo de memoria al no cargar la imagen completa constantemente.

La tesela resultante se envía al visor de lienzo, donde se renderiza

ISSN 1794-211 X • e-lSSN 2322-939X • Vol 20, N° 2 (julio-diciembre 2023). pp. 91-98. Universidad Distrital Francisco José de Caldas-Facultad Tecnológica.

mediante Pillow + Tkinter, superponiendo las líneas guía y las cajas de anotación. La lógica de etiquetado reside en el controlador de anotaciones, que traduce eventos del usuario (clic, arrastre, deshacer) en coordenadas absolutas y normalizadas; cada modificación se persiste al instante mediante el módulo de entrada/salida, encargado de serializar (convertir datos en memoria a un formato de texto plano que pueda guardarse en disco) y deserializar (leer datos guardados del disco y reconstruirlos en memoria para volver a ser utilizados) los archivos en formato de anotación YOLO. Finalmente, el controlador de la GUI (Despachador de Eventos) actúa como orquestador: intercepta acciones del ratón y teclado, actualiza el estado global y notifica a los demás módulos cuando es necesario redibujar o guardar datos.

2.4. Interfaz gráfica

La interfaz gráfica se diseña con el objetivo de maximizar la usabilidad y eficiencia en el proceso de anotación. Se implementó una interfaz intuitiva basada en Tkinter, que permite a los usuarios interactuar fácilmente con la herramienta, permitiendo la visualización de las ortofotos georreferenciadas con fluidez, así como la creación, modificación y eliminación de las etiquetas en formato de anotación YOLO. Se incluyen controles para la gestión de clases de objetos, la asignación de etiquetas y la configuración de parámetros de visualización, posibilitando ajustar el zoom y desplazarse por la imagen para una adaptación precisa de la visualización, permitiendo un análisis detallado.

2.5. Gestión de memoria y rendimiento

Para garantizar una experiencia fluida incluso con ortofotos de varios cientos de megapíxeles, la aplicación adopta un esquema de teselado dinámico. Al abrir un archivo, la imagen completa sólo se mantiene una vez en memoria como arreglo NumPy, evitando copias redundantes. Cada operación de acercamiento o deslizamiento invoca un método de zoom, que calcula el rectángulo de recorte centrado en las coordenadas actuales y extrae de la matriz original sólo los píxeles necesarios. Esa tesela se redimensiona al tamaño del canvas empleado y se visualiza. El objeto gráfico resultante se reutiliza mientras permanezca inalterada la vista, de manera que el recolector de basura puede liberar enseguida la tesela anterior y mantener bajo el consumo de RAM.

El renderizado se realiza sobre un único Widget Canvas, en el que la capa de la imagen se coloca en un nivel inferior para que las anotaciones vectoriales de dibujen por encima sin requerir composiciones costosas. Las líneas guías y los cuadros delimitadores se actualizan mediante coordenadas relativas, lo que evita recalcular la tesela mientras el usuario arrastra el ratón. Cada pulsación de las flechas de dirección desplaza el centro de vista una distancia fija y vuelve a ejecutar el recorte; este proceso implica únicamente operaciones de slicing de NumPy y un escalado ambos lo suficientemente rápido para mantener la interacción sin retardos perceptibles.

Con este enfoque la herramienta conserva en memoria, peor de los casos, la imagen original y la tesela visible, mientras que el almacenamiento interno de los archivos de etiquetas resulta irrelevante desde el punto de vista del uso de recursos. La combinación de acceso directo a la matriz de píxeles y recortes oportunos permite navegar, anotar y hacer zoom sin que el consumo de memoria crezca con la profundidad de la interacción.

2.6. Flujo de anotación

La interfaz está diseñada de acuerdo a las siguientes características:
A la izquierda se despliega la barra lateral con dos secciones:

- **Controles generales:** apertura de imagen, deshacer, zoom y desplazamientos cardinales.
- **Gestión de tipos de objetos:** creación dinámica de etiquetas y selector desplegable para fijar la clase antes de dibujar.

Para anotar un objeto, el usuario sigue los siguientes pasos:

1. Seleccionar la clase en el combobox inferior (p. ej. TapaRegistro).
2. Posicionar el cursor sobre el objeto; las guías cruzadas indican el punto de partida.
3. Mantener presionado el botón izquierdo y arrastrar hasta cubrir el área de interés. Aparece un rectángulo rojo semitransparente que se actualiza en tiempo real.
4. Soltar el botón: Automáticamente se convierte las coordenadas del canvas a la imagen original, normaliza los valores y persiste la línea correspondiente en el archivo .txt.
5. Si la caja no queda conforme, **Ctrl + Z** o el botón Undo, eliminan la última fila del archivo y redibuja el lienzo.

La actualización es inmediata porque el Controlador de Anotaciones comunica cada cambio al Visor de Lienzo y, en paralelo, al Módulo de E/S, cerrando así el ciclo mostrado.

ISSN 1794-211 X • e-lSSN 2322-939X • Vol 20, N° 2 (julio-diciembre 2023). pp. 91-98. Universidad Distrital Francisco José de Caldas-Facultad Tecnológica.

2.7. Persistencia de los datos

Esta herramienta emplea un mecanismo de persistencia inmediata que garantiza que cada acción del usuario refleje de forma permanente en el disco y pueda recuperarse en sesiones posteriores sin pérdida de información. Para ello, cada imagen se vincula con un archivo de etiquetas homónimo (nombre.txt) ubicando en la misma carpeta, lo que simplifica la organización del conjunto de datos y evita depender de bases de datos externas. Las etiquetas se guardan en formato de anotación YOLO estándar (class_id x_center y_center width heigth) con coordenadas normalizadas entre 0 y 1 de modo que los archivos resultan compatibles de forma directa con los flujos de entrenamiento de YOLO.

Al abrir una imagen, la función existe una función que busca automáticamente el archivo con extensión .txt correspondiente: si existe, transforma cada línea a coordenadas absolutas y reconstruye los cuadros delimitadores sobre el lienzo; si no existe, la lista de anotaciones se inicia vacía. Cada vez que el usuario confirma un nuevo rectángulo o deshace el último, se recorre la lista interna, y normaliza las coordenadas para sobre escribir el archivo. Como solo se escribe en texto plano, esta operación apenas tarda milisegundos y mantiene el estado en disco sincronizado con el estado visual.

Gracias a este guardado transaccional, el usuario puede interrumpir el trabajo en cualquier momento: al reabrir la imagen, las anotaciones reaparecen exactamente como se dejaron, sin pasos manuales de importación. Los nombres descriptivos de las clases se conservan en memoria, mientas que el archivo se almacenan únicamente los índices; para evitar incoherencias, el diseño obliga a definir las clases antes de anotar y a mantener estable el orden de ñ añosa durante la sesión. Como cada imagen genera un único archivo de unos kilobytes incluso con cientos de cajas, el sistema escala linealmente con el número de imágenes sin penalizar los tiempos de E/S ni requerir repositorios centralizados. En conjunto este enfoque ofrece una solución robusta y sencilla que resiste cierres inesperados y mantiene compatibilidad total con el ecosistema YOLO.

2.8. Empaquetado con PyInstaller para facilitar la distribución y uso de la herramienta en distintos entornos operativos se optó por generar un ejecutable autónomo utilizando la biblioteca PyInstaller. Esta decisión permite a los usuarios finales puedan ejecutar la aplicación directamente, sin necesidad de instalar Python ni configurar entornos virtuales o dependencias externas.

El comando utilizado para la generación del ejecutable fue: pyinstaller --onefile --noconsole anotador.py

La opción –onefile permite empaquetar todos los archivos necesarios (Incluyendo el intérprete de Python y las bibliotecas como Tkinter, Pillow, Numpy y OpenCV) En un solo archivo .exe. La opción –noconsole se empleó para evitar que se abra una ventana de terminal al iniciar la interfaz gráfica, mejorando la experiencia visual en sistemas Windows.

El ejecutable final tiene un tamaño aproximado de 35 MB Y fue probado exitosamente en sistemas Windows 10 y 11.

Esta estrategia elimina barreras de entrada y permite que la herramienta pueda ser utilizada en equipos de escritorio sin conexión a internet, como es común en contextos de trabajo de campo, laboratorios o instituciones educativas. Además, garantiza portabilidad y consistencia entre instaladores reducción de errores relacionados con incompatibilidad entre versiones de bibliotecas.

2.9. Implementación y pruebas

En la fase de implementación se compilo la versión definitiva de la aplicación mediante PyInstaller y se distribuyó como ejecutable único de 35 MB. El archivo se instaló en equipos con Windows 10 y 11 sin requerir configuraciones adicionales.

Para la validación práctica se reclutaron 10 alumnos de la carrera de Ingeniería en Topografía y Geomática que realizaban sus prácticas profesionales. Cada estudiante recibió un lote de ortofotos y una capacitación en la que se les explicó el funcionamiento de la herramienta.

Durante 2 jornadas consecutivas etiquetaron tapas de registro de drenaje y marcas en el suelo.

3. Resultados

3.1. Funcionamiento general de la herramienta

La herramienta desarrollada se ejecutó correctamente en estaciones de trabajo con características de gama media, cumpliendo con los requisitos mínimos de rendimiento y compatibilidad establecidos durante la fase de

ISSN 1794-211 X • e-lSSN 2322-939X • Vol 20, N° 2 (julio-diciembre 2023). pp. 91-98. Universidad Distrital Francisco José de Caldas-Facultad Tecnológica.

diseño. Gracias al uso de PyInstaller, se generó un ejecutable autónomo de aproximadamente 35 MB, el cual fue instalado sin dificultades en equipos con sistemas operativos Windows 10 y Windows 11. El tiempo de arranque fue inferior a tres segundos en unidades SSD, sin presentar advertencias por dependencias faltantes ni errores en la ejecución inicial. Este comportamiento confirma la portabilidad de la herramienta y su capacidad para operar sin necesidad de conexión a internet ni configuraciones adicionales.

3.2. Validación práctica con usuarios

Para evaluar el desempeño práctico de la herramienta, se realizó una prueba de campo con la participación de diez estudiantes de la carrera de Ingeniería en Topografía y Geomática, quienes se encontraban en periodo de prácticas profesionales. A cada participante se le asignó un conjunto de ortofotos de alta resolución, y recibió una breve capacitación sobre el uso del sistema. Durante dos jornadas consecutivas, los estudiantes realizaron la anotación manual de objetos de interés, principalmente tapas de registro de drenaje y marcas pintadas sobre el pavimento.

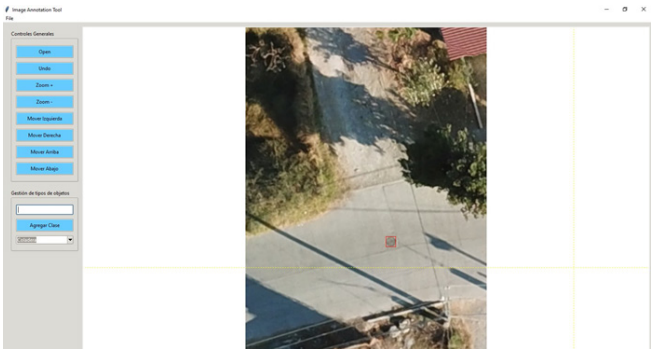

Figura 3. Interfaz gráfica de la herramienta donde se puede notar que se ha etiquetado un objeto.

La interacción con la herramienta se desarrolló sin incidentes y los estudiantes pudieron usar la interfaz que se muestra en la Figura 3, en la cual no se reportaron errores durante la instalación ni cierres inesperados durante las sesiones de trabajo. Tampoco se presentaron pérdidas de datos al reanudar tareas previamente iniciadas. Esta validación en un entorno real permite afirmar que la solución es robusta y adecuada para ser adoptada en campañas de mayor escala, así como en proyectos colaborativos de etiquetado de datos.

3.3. Eficiencia en el proceso de anotación

Aunque no se realizaron mediciones cuantitativas del tiempo de anotación por imagen, los participantes coincidieron en señalar que la herramienta ofrecía una experiencia fluida y eficiente. Funcionalidades como el zoom con recorte dinámico, la visualización en tiempo real del rectángulo de selección, las líneas guía que siguen al cursor y la opción de deshacer anotaciones contribuyeron significativamente a reducir la fatiga visual y los errores durante el proceso de etiquetado. Esta percepción sugiere que la herramienta favorece un flujo de trabajo ágil, permitiendo completar tareas de anotación en menos tiempo en comparación con métodos manuales tradicionales.

La Tabla 2 resume la cantidad de etiquetas generadas en cada ortofoto durante las pruebas prácticas. Estos datos permiten dimensionar el volumen de anotaciones realizadas y muestran la capacidad de la herramienta para manejar imágenes de gran tamaño sin afectar la continuidad del trabajo. La distribución de etiquetas evidencia que, pese a las diferencias en la complejidad visual de cada ortofoto, el flujo de anotación se mantuvo consistente, favorecido por el teselado dinámico y la persistencia inmediata de datos, lo cual redujo el riesgo de pérdidas de progreso y aseguró la integridad del conjunto de entrenamiento.

Ortofotos	Etiquetas
V1	102
V2	69
V3	55
V4	87
V5	79
V6	68
V7	75

ISSN 1794-211 X • e-ISSN 2322-939X • Vol 20, N° 2 (julio-diciembre 2023). pp. 91-98. Universidad Distrital Francisco José de Caldas-Facultad Tecnológica.

V8	90
V9	63
V10	76

Tabla 2. Cantidad de tapas de registro de drenaje etiquetadas en cada ortofoto.

3.4. Rendimiento y consumo de recursos

Gracias a la implementación de un esquema de teselado dinámico, la herramienta mantuvo un uso de memoria eficiente incluso al manipular ortofotos de varios cientos de megapíxeles. La imagen completa se almacena una sola vez en memoria como matriz NumPy, y únicamente se renderiza la sección visible (tesela activa) sobre el canvas de la interfaz gráfica. Esta tesela se redimensiona y reutiliza mientras el usuario no modifique el encuadre, minimizando el uso de recursos. Las operaciones de acercamiento y desplazamiento se ejecutaron sin retardos perceptibles, permitiendo una navegación fluida y precisa. Esta optimización garantiza que la herramienta pueda ser utilizada en contextos donde no se cuenta con equipos de alto rendimiento.

3.5. Persistencia y recuperación de etiquetas

El sistema de guardado inmediato permitió asegurar la persistencia de las anotaciones sin requerir intervenciones manuales. Cada vez que el usuario completó una caja delimitadora, la información se registró automáticamente en un archivo .txt homónimo ubicado en la misma carpeta que la imagen correspondiente. Este archivo conserva las coordenadas normalizadas en el formato estándar YOLO, permitiendo su uso directo en flujos de entrenamiento. Al reabrir una imagen previamente anotada, las etiquetas fueron restauradas con precisión sobre el lienzo, conservando su posición, tamaño y clase. Este comportamiento demostró ser resistente a interrupciones inesperadas y facilitó la gestión de conjuntos de datos extensos sin necesidad de bases de datos centralizadas.

3.6. Compatibilidad con flujos de entrenamiento YOLO

Los archivos generados por la herramienta fueron compatibles de forma inmediata con entornos de entrenamiento basados en YOLO. Al mantener el formato de anotación estándar y la estructura por pares de archivos imagen–etiqueta, se garantizó la integración directa con scripts de entrenamiento. Esta compatibilidad valida el propósito central del desarrollo: facilitar la creación de conjuntos de datos para modelos de detección de objetos, en particular para aplicaciones en análisis geoespacial, infraestructura urbana y monitoreo territorial. En trabajos posteriores se emplearán las anotaciones producidas con esta herramienta para entrenar modelos YOLO y evaluar su precisión en la detección de objetos en ortofotos multitemporales.

4. Conclusiones

La herramienta desarrollada constituye una solución efectiva y portable para la anotación manual de objetos pequeños en ortofotos de alta resolución, generando etiquetas compatibles con el formato YOLO. Su arquitectura basada en teselado dinámico permite mantener un uso de memoria eficiente y una navegación fluida incluso en equipos de gama media, sin comprometer la estabilidad ni la precisión del etiquetado.

Las pruebas prácticas con usuarios confirmaron su robustez, facilidad de uso y confiabilidad en contextos reales, lo que refuerza su utilidad tanto en entornos académicos como en aplicaciones profesionales. La compatibilidad directa con flujos de entrenamiento YOLO permite una integración inmediata en proyectos de visión por computadora sin necesidad de herramientas adicionales.

La gestión eficiente de recursos se logra mediante la carga única de la ortofoto como matriz y la actualización de una única tesela visible, evitando cuellos de botella comunes en otros anotadores. Este enfoque técnico resuelve problemas frecuentes como cierres inesperados, pérdida de progreso o ralentizaciones durante la anotación.

A partir de estos resultados, se recomienda explorar líneas futuras que incluyan: (i) integración de métricas cuantitativas de eficiencia (tiempo por imagen, errores por sesión), (ii) incorporación de módulos de

ISSN 1794-211 X • e-lSSN 2322-939X • Vol 20, N° 2 (julio-diciembre 2023). pp. 91-98. Universidad Distrital Francisco José de Caldas-Facultad Tecnológica.

anotación semiautomática y aprendizaje activo, y (iii) extensión del sistema hacia exportaciones georreferenciadas (shapefiles, GeoJSON) para vinculación con entornos GIS.

Trabajos futuros

Como trabajo futuro, se plantea utilizar las etiquetas generadas con esta herramienta para entrenar modelos YOLO (como YOLOv11) especializados en la detección automática de objetos urbanos tales como tapas de registro, señales viales, alcantarillas o luminarias. Estos modelos se aplicarán sobre ortofotos multitemporales generadas con UAVs para identificar cambios en la posición de estos elementos, lo que permitirá monitorear de manera automatizada desplazamientos del terreno y posibles indicios de deslizamientos activos.

Asimismo, se considera integrar esta herramienta en un flujo de trabajo más amplio que incluya:

- Evaluación de métricas de desempeño del modelo (precisión, recall, F1-score) en escenarios reales.
- Comparación de ortofotos multitemporales para detectar desplazamientos con respaldo en modelos digitales de elevación (DEM).
- Exportación automática de detecciones como archivos shapefile georreferenciados.
- Incorporación futura de métodos de anotación semiautomática y aprendizaje activo para reducir la carga manual en nuevas campañas de etiquetado.

Estos avances permitirán consolidar un sistema integral de monitoreo geoespacial, aprovechando la inteligencia artificial para la interpretación de datos visuales urbanos en entornos vulnerables a fenómenos geodinámicos.

5. Referencias

[1] M. Aljabri, M. AlAmir, M. AlGhamdi, M. Abdel-Mottaleb, and F. Collado-Mesa, "Towards a better understanding of annotation tools for medical imaging: A survey," Multimedia Tools and Applications, vol. 81, no. 18, pp. 25877–25911, 2022, doi: 10.1007/s11042-022-12100-1.

[2] J. Weng, Z. Tang, and Y. Liu, "Application of improved YOLOv8 image model in urban manhole cover defect management and detection: Case study," Sensors, vol. 25, no. 13, Art. 4144, 2025, doi: 10.3390/s25134144.

[3] Y. Zhang, F. Li, X. Wang, et al., "Real-time detection of road manhole covers with a deep learning approach," Scientific Reports, vol. 13, Art. 18791, 2023, doi: 10.1038/s41598-023-43173-z.

[4] H. Li, Q. Zhao, S. Chen, et al., "Manhole cover classification based on super-resolution and UAV aerial imagery with YOLOv8," Applied Sciences, vol. 14, no. 7, Art. 2769, 2024, doi: 10.3390/app14072769.

[5] V7 Labs, "A Friendly Guide to LabelImg," 2022. [Online]. Available: https://www.v7labs.com/blog/labelimg-guide

[6] HumanSignal, "labelImg," GitHub repository, 2025. [Online]. Available: https://github.com/HumanSignal/labelImg (original release 2015).

[7] VGG (Visual Geometry Group, Oxford), "VGG Image Annotator (VIA)," 2019–2020. [Online].
Available: https://www.robots.ox.ac.uk/~vgg/software/via/via_demo.html