AUTOMATED BUILDING OBJECT SEGMENTATION FROM UAV ORTHOGRAPHIC IMAGERY USING DEEP LEARNING AND GIS TECHNIQUES

Van Truong Vu[1,*], Thi Thu Trang Truong[2]
[1]*Institute of Techniques for Special Engineering, Le Quy Don Technical University*
[2]*Faculty of Foreign Languages, Le Quy Don Technical University*

Abstract

This paper presents an automated model for segmenting building objects in orthographic images captured by unmanned aerial vehicles (UAVs) using deep learning and GIS techniques. The proposed method consists of two phases: Phase 1 employs a two-stage deep learning method (named YOLO-SAM) that uses YOLOv11 for coarse building segmentation, followed by the SAM2 (Segment Anything Model) algorithm to refine and enhance the results. Phase 2 further processes this shapefile using a series of geometry-based post-processing techniques, including the Douglas-Peucker algorithm for boundary simplification and curve smoothing to optimize the overall shape. The approach achieves high segmentation accuracy (~mAP of 0.75) and provides a scalable, practical solution for mapping in Vietnam. Unlike previous methods, this approach stands out by combining YOLOv11 and SAM2 for accurate, detail-rich segmentation, and by integrating this with GIS-based post-processing in a unified, efficient pipeline.

Keywords: *Building segmentation; UAV; deep learning; GIS.*

1. Introduction

Building detection from aerial imagery has become a critical task in urban planning, disaster management, and infrastructure development. The availability of high-resolution orthographic images from unmanned aerial vehicles (UAVs) has spurred significant research over the past five years, particularly with the integration of deep learning and geographic information systems (GIS). Commercial tools like ArcGIS Pro, developed by Esri, have emerged as powerful platforms supporting machine learning (ML) and deep learning (DL) workflows for geospatial tasks, including building segmentation. This section surveys these capabilities, evaluates prior studies globally and in Vietnam, and highlights the novel contributions of the proposed method.

In recent years, there has been a growing trend of applying deep learning techniques to building footprint extraction from remote sensing imagery due to their ability to capture complex spatial patterns. Table 1 summarizes several representative studies employing

[*] Corresponding author, email: truongvv@lqdtu.edu.vn
DOI: 10.56651/lqdtu.jst.v8.n01.969.sce

deep learning models, demonstrating advancements in both accuracy and computational efficiency. These include architectures like U-Net, ResNet, and DeepLabV3+ that focus on semantic segmentation tasks. Some studies explore hybrid methods that integrate CNNs with attention mechanisms or use high-resolution datasets to refine outputs. In addition to segmentation-based models, object detection approaches have gained popularity for their real-time capabilities. Among them, YOLO-based methods have shown significant potential. For instance, Chang *et al.* proposed YOLOv5ds-RC to enhance building contour detection, achieving accuracy of 0.8849. These examples highlight YOLO's adaptability and effectiveness when customized for building extraction tasks.

Tab. 1. Summary of recent deep learning-based methods for building footprint extraction from remote sensing imagery

No.	Reference	Method/Model	Main Contribution	Dataset	Results
1	Shi *et al.* (2019) [1]	BFGAN (Conditional GAN + WGAN Loss)	Improved building boundary extraction	Satellite images	Outperformed U-Net in accuracy
2	Huang and Jin (2022) [2]	Mobile-UNet + GAN	Mapping wildland-urban interface buildings	3 counties, California	F1-score: 0.62-0.75
3	Dimitrovski *et al.* (2024) [3]	U-Net ensemble (ViT, ConvFormer, EfficientNet)	Geometric averaging to enhance segmentation	Multiple remote sensing datasets	Top-tier performance
4	Alsabhan and Alotaiby (2022) [4]	U-Net + ResNet50	High segmentation accuracy for urban areas	Massachusetts Building dataset	IoU: 82.2%, F1: 0.9, Accuracy: 90%
5	Deng *et al.* (2023) [5]	Hybrid Task Cascade (HTC)	Multi-level instance segmentation with box-mask fusion	3 datasets	Higher accuracy than traditional instance segmentation
6	Li and Zhao (2022) [6]	DeepLabV3+ with attention mechanism	Clearer boundaries and semantic segmentation	High-res satellite images	Significant segmentation improvement
7	Guo *et al.* (2024) [7]	NPSFF-Net (Pseudo-Siamese + Feature Fusion)	Strong semantic extraction, generalization	Aerial, Massachusetts, Satellite	IoU: 89.45%, 71.88%, 68.72%
8	Chang *et al.* (2025) [8]	YOLOv5ds-RC	Semantic segmentation + edge refinement	Gaofen-2	Accuracy: 0.8849, AP@0.5: 0.75863, mAP@0.5-0.95: 0.47388

In general, the trend of related research in recent years has been to applying deep learning architectures like GANs, U-Net, Mobile-UNet, and variations of YOLO to solve this problem. These methods achieve high accuracy but still face challenges, especially in complex urban areas with varied building shapes, roof materials, and occlusions. Pixel-based segmentation may result in inaccurate boundaries when buildings are adjacent or partially hidden. The lack of height data in orthophotos also limits extraction precision. While tools like ArcGIS Pro provide pretrained models, they are not tailored for UAV imagery or Vietnam's architectural diversity, and few methods offer an end-to-end solution compatible with GIS workflows. The novelty of this approach is highlighted in several key aspects:

Proposed algorithm: A two-stage algorithm, YOLO-SAM, is proposed by integrating YOLOv11 and SAM2 to address the specific challenges of building segmentation, significantly enhancing both accuracy and detail in the results.

(1) Full pipeline integration: The approach seamlessly integrates AI-driven segmentation with GIS post-processing, ensuring a streamlined workflow that generates a refined shapefile.

(2) Vietnam-specific adaptation: The proposed method is customized for the Vietnamese context by leveraging UAV imagery and a compact, locally-sourced dataset, ensuring scalability and effectiveness in resource-constrained environments.

The paper is structured as follows: Section 2 describes the methodology, Section 3 presents the results, and Section 4 concludes with key findings and contributions.

2. Methodology

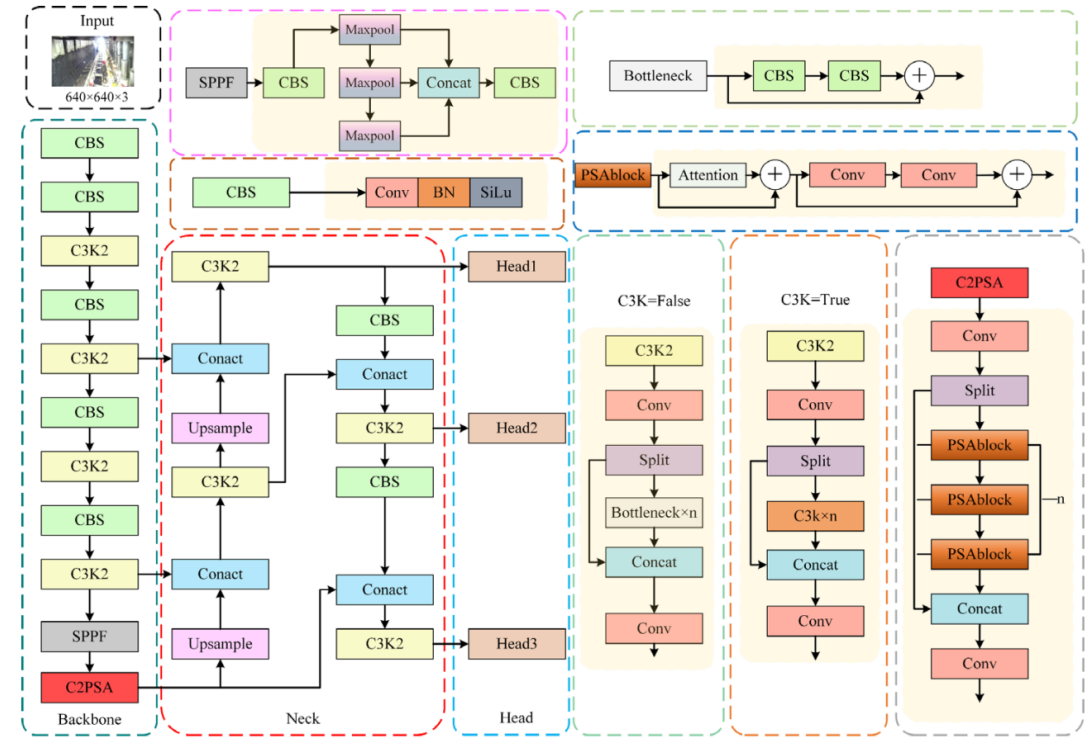
2.1. Deep learning algorithms

In this work, the authors utilize two core deep learning algorithms: YOLOv11 for object detection and SAM2 for image segmentation.

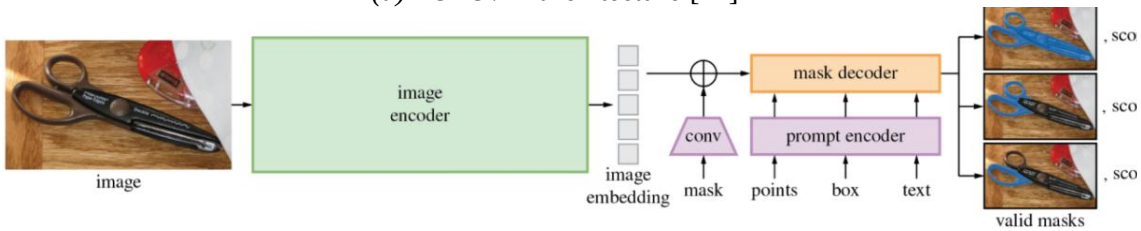
YOLOv11 [9] (Fig. 1a), which stands for "You Only Look Once version 11" is an advanced real-time object detection model designed to detect objects within an image in a single pass. It divides the input image into an S×S grid, where each cell predicts multiple bounding boxes, including the center coordinates (x,y), width (w), height (h), an objectness score, and class probabilities. This structure enables YOLOv11 to efficiently detect multiple objects of varying sizes and categories within an image. The architecture of YOLOv11 is composed of three main components: the Backbone, Neck, and Head. The Backbone processes the input (e.g., 640×640×3) using CSP blocks (C2PSA, C3K2), SPPF, and CBS with SiLU to extract feature maps. The Neck enhances features with upsampling, concatenation, and PSABlock attention, combining multi-scale features via

Bottleneck. The Head generates three prediction heads (Head1, Head2, Head3) with Conv and Split operations for bounding boxes, scores, and class probabilities at different scales. This compact design with attention and multi-scale fusion suits real-time detection, including geographic objects from UAV or LiDAR data.

SAM2 [10] (Fig. 1b), which stands for "Segment Anything Model 2", is a powerful and flexible image segmentation algorithm designed to generate precise pixel-level masks for any object in an image, regardless of predefined class labels. The model first extracts rich visual features through convolutional backbone networks, capturing both local and global image context. These features are then passed through a segmentation head, which predicts binary or multi-class masks that delineate object boundaries. To optimize performance, SAM2 is trained using a pixel-wise cross-entropy loss function, enabling accurate segmentation even in complex or diverse visual scenes.



(a) YOLOv11 architecture [11]



(b) The Segment Anything Model (SAM)

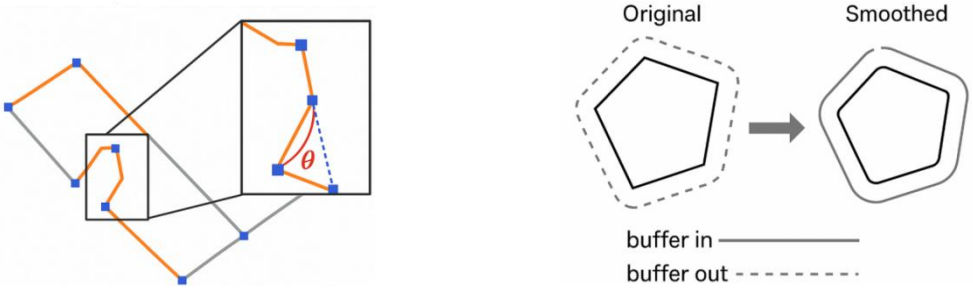
Fig. 1. The two main deep learning algorithms of the YOLO-SAM model.

2.2. The post-processing algorithms

The Douglas-Peucker algorithm [12] (Fig. 2a) is a classic geometric simplification method used to reduce the number of points in a polygonal chain while retaining its essential shape. Given a sequence of points representing an object contour, the algorithm recursively eliminates points that contribute the least to the overall geometric structure based on a defined tolerance threshold ϵ. It begins by connecting the first and last points in the sequence with a straight line segment and then calculates the perpendicular distance from each intermediate point to this segment. The point with the maximum distance d_k is identified using the formula:

$$d_k = \frac{\left|(x_n - x_0)(y_0 - y_k) - (x_0 - x_k)(y_n - y_0)\right|}{\sqrt{(x_n - x_0)^2 + (y_n - y_0)^2}} \quad (1)$$

where $P_0 = (x_0, y_0)$ and $P_n = (x_n, y_n)$ are the endpoints of the segment, and $P_k = (x_k, y_k)$ is any intermediate point in the sequence. This formula computes the perpendicular (shortest) distance from P_k to the line segment P_0P_n. If $d_k > \epsilon$, that point is preserved, and the algorithm is recursively applied to the two new segments: from the start to the farthest point and from the farthest point to the end. If all distances are below ϵ, intermediate points are discarded. This approach effectively retains major shape-defining vertices while removing redundant data, making it particularly useful for simplifying complex boundaries in image segmentation outputs.



(a) The Douglas-Peucker algorithm *(b) The buffer in - buffer out technique*

Fig. 2. The post-processing algorithms.

The buffer-in buffer-out technique (Fig. 2b) is a morphological post-processing method used to smooth the contours of segmented objects. This technique consists of two sequential geometric operations applied with the same buffer radius. In the first step, the object boundary is expanded outward (buffer-in), effectively filling small gaps and bridging narrow indentations. In the second step, the expanded region is contracted back to its original scale (buffer-out), removing minor protrusions and sharp corners that may

have resulted from pixel-level irregularities. By combining expansion and contraction, this approach produces smoother and more visually coherent object boundaries, while preserving the essential structure and shape of the segmented region.

3. The proposed method

The pipeline of the proposed integration process for deep learning and geometric algorithms to extract building boundaries from UAV images consists of two main stages: (1) Identifying building objects using the YOLO-SAM deep learning model, and (2) Post-processing segmentation results to refine building boundaries with geometric algorithms. The details of the steps are shown in Fig. 3.

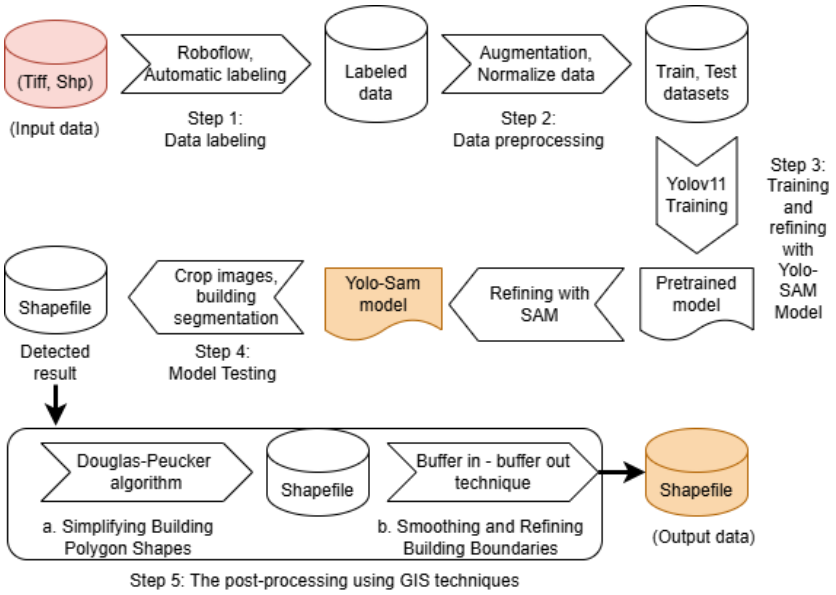


Fig. 3. The pipeline of the proposed method.

The input image data consists of UAV flight area orthophotos (in .tif format) and vector data of these areas (in .shp format). The detailed processing steps for stage 1 are as follows:

Step 1. Data labeling

In this study, the authors employed two methods for data labeling. The first method involved using Roboflow to manually label each building in the images. The second method was automatic labeling using pre-digitized GIS data of buildings from UAV orthophoto maps of Vietnam. The authors developed a tool that enables automatic labeling on UAV orthophotos using existing vector data layers. The labels are normalized to match the input format required by the YOLO-SAM algorithm.

Step 2. Data preprocessing

It is important to note that processing the entire image at once can be very time-consuming, memory-intensive, and especially unsuitable for deep learning algorithms. To solve this issue, rather than using the entire image, we divide the image into grid cells to reduce the input size, speeding up the processing time and improving efficiency. In this study, the solution involves cropping the images and their corresponding labels into 256×256 pixel grid cells. The labels are then further normalized to fit the deep learning model's input format.

Step 3. Training and refining with YOLO-SAM model

The YOLO-SAM model consists of two main processing steps. First, the YOLOv11 model is trained and then used to detect coarse building regions in the images. These roughly detected building regions are subsequently refined using the SAM2 model. In the first step, the dataset is divided into three subsets: training (train), validation (validation), and testing (test) in a ratio of 7:2:1. The training set is used to update the weights of the neural network, the validation set is used to evaluate the current model to detect if overfitting is happening, which allows us to select the optimal point to stop training, and the test set is used to evaluate the model's final performance on new data. A sufficiently large, diverse, and accurately labeled dataset is a key factor in ensuring that the YOLO-SAM model learns effectively and generalizes well.

In the second step, the coarse building regions detected by the YOLOv11 model are used as input prompts for the SAM2 (Segment Anything Model) to perform fine-grained instance segmentation. SAM2 is a powerful transformer-based model capable of producing high-quality segmentation masks given various types of prompts such as bounding boxes, points, or text. In this context, the bounding boxes generated by YOLOv11 serve as box prompts that guide SAM to extract precise building boundaries. The integration of SAM2 enables the refinement of the building shapes, addressing the limitations of YOLO's bounding box output, which typically lacks accurate contour information. SAM processes each detected region and generates a binary mask that closely matches the actual shape of the building, effectively enhancing the segmentation performance, especially in complex urban scenes with irregular or adjacent structures.

Step 4. Model testing

In this step, the test orthophoto is divided into smaller images according to the same grid pattern used for training. Each smaller image is then passed through the YOLO-SAM model to detect building objects. Once the processing of all small images is complete, the results are stitched together. It is important to note that when the dataset is divided into grid cells, the images are cropped with overlaps using a sliding window technique. This

ensures that objects located at the edges of the image tiles are fully captured. However, this overlap leads to some objects being detected multiple times. To address this, the Non-Maximum Suppression (NMS) algorithm [13] is applied to eliminate redundant detections. NMS works by selecting the bounding box with the highest confidence score and suppressing all other overlapping boxes based on a predefined Intersection over Union (IoU) threshold.

After the building objects are recognized in the large orthophoto, the coordinates of these objects are still in image coordinates (row, column format). The next step is to reconstruct the actual geographic coordinates of the building polygons based on the corner coordinates of the bounding boxes and the pixel resolution of the image. This converts the image-based representation into a vector-based format. The final output is a shapefile map containing building polygons. This map will be further processed in the next stage to generate the final result.

Step 5. The post-processing using GIS techniques

a) Simplifying building polygon shapes

The building polygons obtained from YOLO-SAM often have a large number of vertices and jagged edges. The Douglas-Peucker algorithm is applied to simplify the polygons by reducing the number of vertices while maintaining the overall shape of the building. The algorithm iteratively selects key vertices that are further than a specified epsilon threshold from the straight line connecting the two endpoints.

b) Smoothing and refining building boundaries

The smoothing method applies a buffer in - buffer out technique to round sharp corners and smooth the edges of the polygon. This process helps eliminate small jagged details and improves the visual quality of the object shape. After smoothing, the system ensures geometric validity and retains only the main polygon if the result is a multiPolygon. The final result is stored as a GIS data layer with both geometric and semantic attributes.

4. Results and discussion

4.1. Experimental datasets

The training dataset consists of images from various regions, which were preprocessed and reformatted for model compatibility. Specifically, it is a combination of three different datasets:

- Kaggle Mapping Challenge [14]: A random selection of 15,000 images from a larger dataset of approximately 280,000 images (Segments and Maps for Identifying Objects).

- Roboflow Labeled Dataset [15]: A pre-labeled dataset containing 16,429 images from the Buildings Instance Segmentation Computer Vision Project.

- UAV Orthophoto Dataset (Vietnam): Manually labeled orthophotos captured in 2022 by the DJI Matrice 300 RTK at an estimated altitude of 300-500 m in Go Cong town, Tien Giang province, with a spatial resolution of 0.2 m/pixel in the RGB band.

4.2. Input parameter configuration

Below is the input parameter configuration table (Tab. 2) for the YOLO-SAM algorithm:

Tab. 2. YOLO-SAM input parameter configuration

Parameter	Value	Description
tile_size	256	Tile size in pixels
batch_size	8	Number of tiles per batch
epochs	200	The number of times the dataset is used in training
iou	0.7	Intersection over Union (IoU) threshold for object detection
confidence	0.25	Minimum confidence threshold for YOLO (0-1)
generalization_tolerance	0.5	Generalization tolerance for polygons (meters)
buffer_size	0.5	Buffer size for polygon smoothing

4.3. Experimental results

a) Training time

To evaluate training time, the proposed algorithm with the configuration was executed on a machine equipped with an AMD Ryzen 5 5600H CPU, 32 GB RAM, and an NVIDIA GeForce RTX 3050 Ti Laptop GPU. On a subset of 1,530 images (256×256 resolution), the algorithm achieved an average training time of 36.39 ± 1.46 seconds per epoch. Training time may vary depending on the size and complexity of the dataset.

b) Performance of the YOLO-SAM algorithm

The metric used to evaluate the model's accuracy is Average Precision (AP) in Eq. (2) [16], [17], which represents the area under the Precision-Recall (PR) curve at a specific Intersection over Union (IoU) threshold. In this study, AP is calculated at an IoU threshold of 0.5, meaning a prediction is considered correct if its overlap with the ground truth is at least 50%. A higher AP at this threshold indicates that the model is more accurate and reliable in detecting objects with reasonable localization precision.

$$AP = \sum_{i=0}^{n-1} (\text{Recall}_{i+1} - \text{Recall}_i) \times \text{Precision}_{i+1} \quad (2)$$

where $\text{Recall}_{i+1}, \text{Recall}_i$ are the recall rates at points $i+1$ and i on the curve (measures

how many actual buildings are correctly detected); Precision_{i+1} is the precision at point $i+1$ on the curve (measures how many detected buildings are actually correct). The detailed experimental results are shown in Figs. 4-9. Observing the error measurement plots in Fig. 4 during the training process, it can be seen that the training results of the YOLOv11 model show a stable decrease in all training losses (box, segmentation, classification, and distance focal loss), indicating effective learning. However, the validation losses, especially segmentation and box loss, show a slight upward trend after a certain number of epochs, suggesting signs of overfitting. Precision and recall metrics for both building detection and mask segmentation gradually improve, though the recall remains relatively stable. The model achieves a high mAP@50 for buildings (~0.75) and masks (~0.69), reflecting strong object detection performance. However, the lower and more fluctuating mAP@50–95 values suggest that the model's performance varies across different IoU thresholds. Overall, the model generalizes well but could benefit from early stopping or regularization to reduce overfitting and improve segmentation stability.

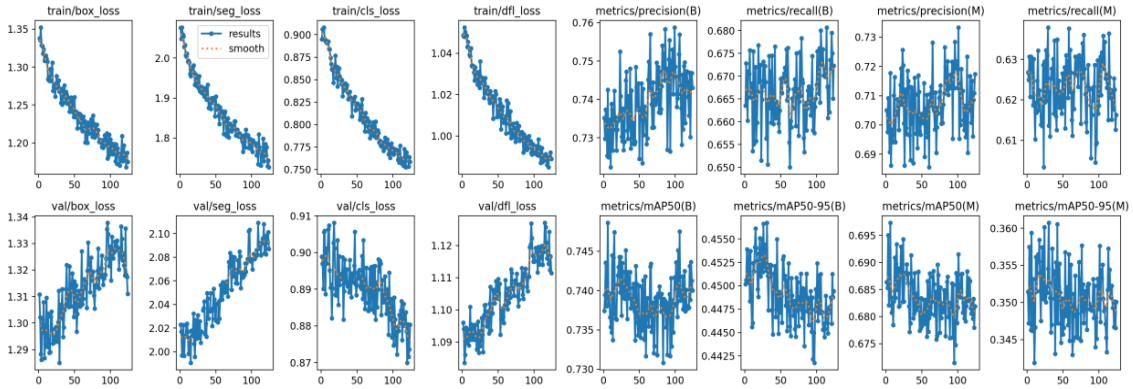
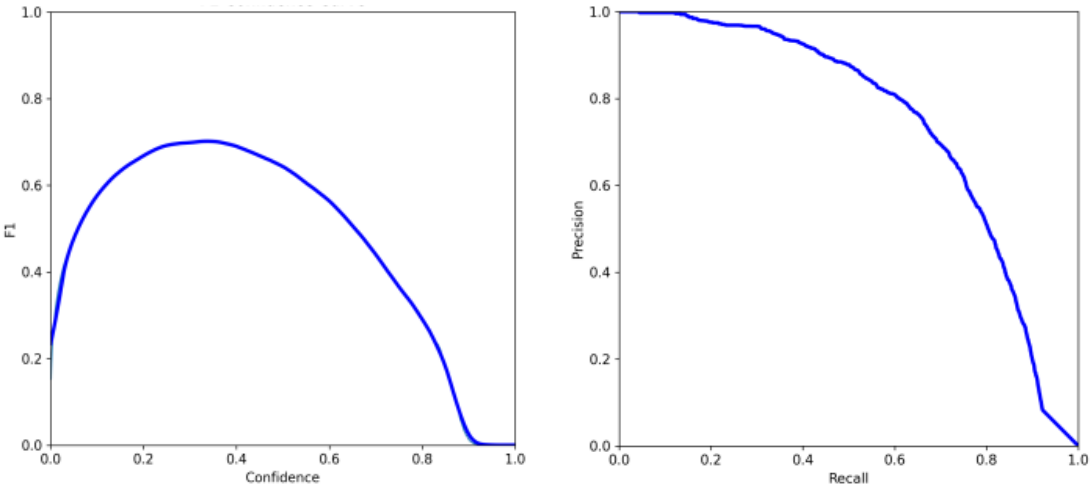


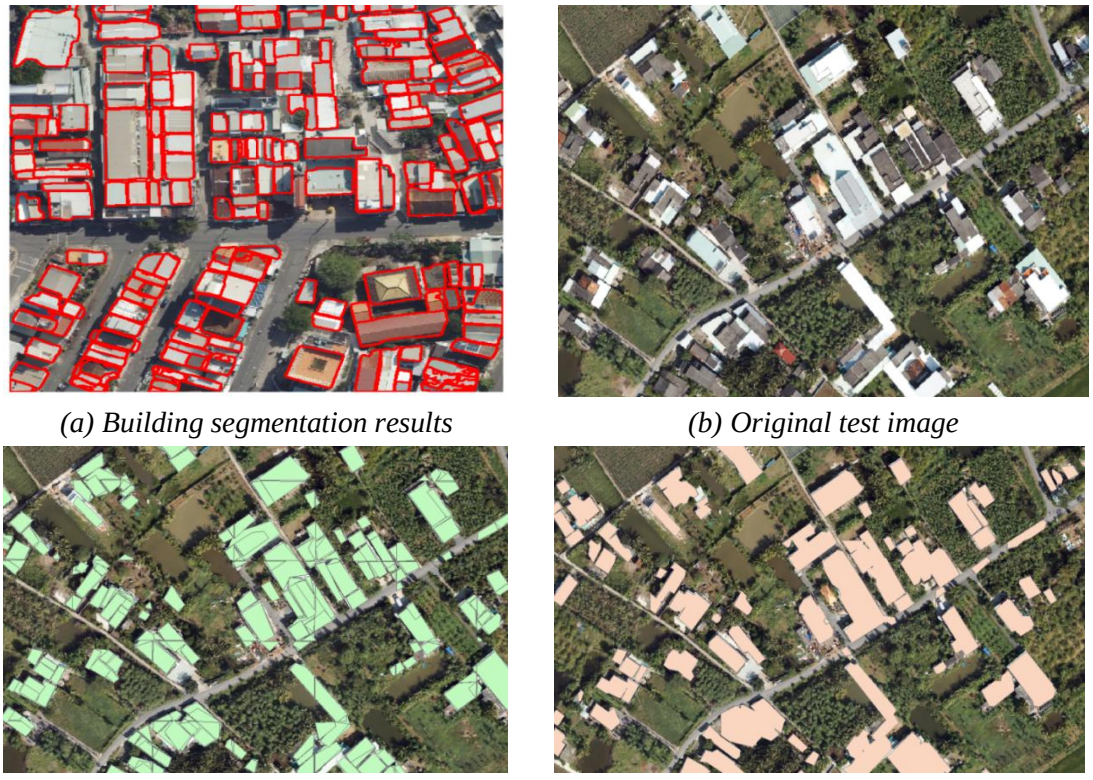
Fig. 4. Training model results plots.

Figure 5 illustrates the Precision-Confidence curve which shows how the model's precision varies with increasing confidence thresholds. Based on the chart, the model demonstrates good discrimination ability, achieving high precision at higher confidence levels, which indicates reliable predictions. However, a clear trade-off between precision and recall is observed: as confidence increases, precision improves while recall decreases, meaning more true positives are missed. The F1-score peaks around a confidence threshold of 0.3-0.4, representing the optimal balance between precision and recall. Overall, the model performs well, and adjusting the confidence threshold allows for tuning between accuracy and completeness depending on specific application requirements.



(a) F1-Confidence curve *(b) Precision-Recall curve*

Fig. 5. The model's precision curve.



(a) Building segmentation results *(b) Original test image*

(c) Segmented, stitched, and vectorized image results *(d) Simplification and smoothing processing result*

Fig. 6. Building segmentation and post-processing results.

Additionally, the results of segmenting building objects were qualitatively evaluated by visualizing certain areas in both the training and test images (Fig. 6).

YOLO-SAM successfully detected most of the building blocks, even for those with unusual shapes like L-shaped, U-shaped, or partially occluded structures. However, there are still some difficult cases, such as dilapidated houses, reflective tin roofs, or structures obscured by dense foliage, where detection was missed or segmentation was inaccurate. These are the main error cases that lead to a decrease in the average precision (AP) value.

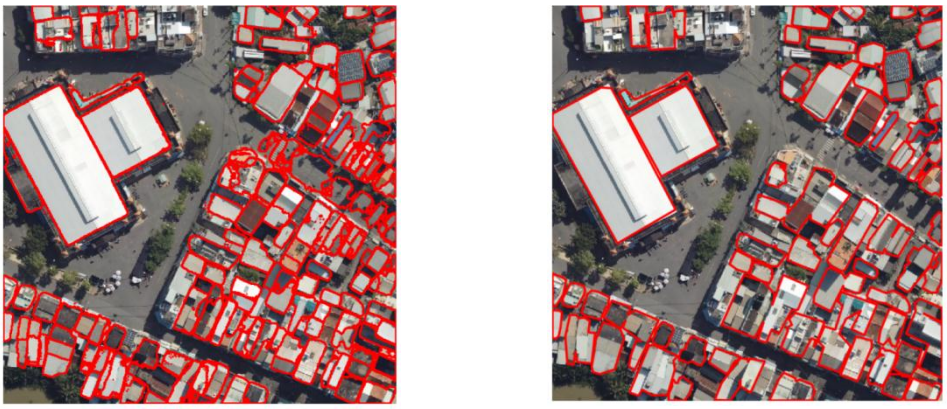
To visually illustrate the improvements in boundary accuracy, Fig. 7 compares the results of using YOLO only (left) versus the proposed YOLO-SAM method (right). It is evident that the YOLO-only approach produces rough segmentation masks with jagged edges and excessive detail, leading to irregular building outlines. In contrast, the YOLO-SAM method refines these boundaries, making them smoother, more regular, and better aligned with real-world structures. This improvement is particularly noticeable for buildings with complex shapes, where the integration of SAM helps enhance segmentation accuracy.

(a) Building boundary extraction using YOLO only

(b) Improved building boundary extraction using YOLO-SAM

Fig. 7. Comparison of building boundary extraction: YOLO only vs. YOLO-SAM.

Figure 8 illustrates the comparison between building segmentation results using YOLO-SAM (left) and the refined boundaries after applying the Douglas-Peucker algorithm (right). YOLO-SAM demonstrates fast and accurate building detection; however, the extracted boundaries still contain excessive details and distortions due to segmentation noise. After applying the Douglas-Peucker algorithm, the boundaries are smoothed, reducing unnecessary vertices while preserving the overall building shape. This combination optimizes the data, making it more suitable for GIS applications and automated urban mapping.



(a) Building segmentation results using YOLO-SAM

(b) Refined building boundaries after applying the Douglas-Peucker algorithm

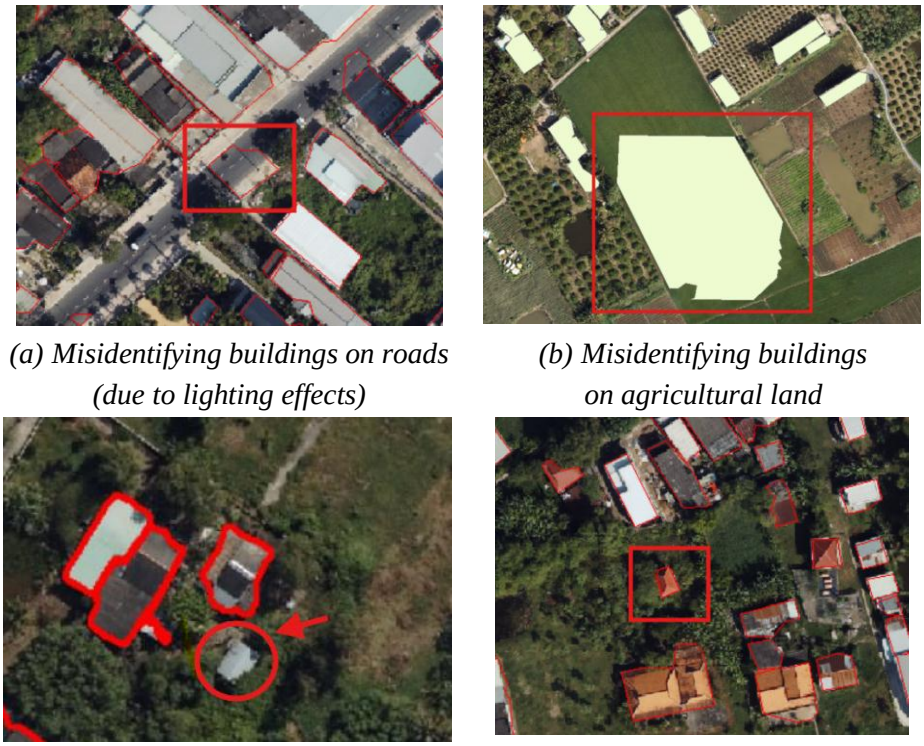
Fig. 8. Comparison of building segmentation results: YOLO-SAM vs. Refined boundaries with Douglas-Peucker simplification.

4.4. Discussion

Based on the achieved results, it can be observed that the proposed YOLO-SAM method offers significant advantages over traditional building boundary extraction techniques. By integrating deep learning with geometric algorithms, it automates most of the process, reducing reliance on manual digitization or fieldwork. This improves efficiency, cutting down time, effort, and workforce while enhancing large-scale mapping productivity.

YOLO-SAM leverages deep learning's ability to extract hierarchical image features, ensuring a clear inference process from object detection to smooth boundary refinement. Training on high-resolution UAV data enhances the method's adaptability to local contexts, but a more diverse dataset is needed to improve robustness. However, challenges remain in complex environments where buildings may be misclassified or missed (Fig. 9) due to dense vegetation, occlusion by other structures, or shadows. To mitigate these issues, integrating additional data sources, such as LiDAR-derived elevation models, including Digital Surface Models or Digital Terrain Models, could enhance detection accuracy by providing valuable height information.

In summary, despite these advancements, the proposed post-processing framework still has certain limitations. First, the final accuracy is highly dependent on the IoU threshold used in the YOLO-SAM model. If a building is missed during the initial detection, post-processing alone cannot recover it. Second, parameter selection for key steps such as tolerance and buffer size is currently manual, relying on experience and trial-and-error. Automating this tuning process for varying image conditions remains an open challenge. Last, the method struggles with highly occluded, fragmented, or irregularly shaped buildings, where segmentation errors may still occur.



(a) Misidentifying buildings on roads (due to lighting effects)

(b) Misidentifying buildings on agricultural land

(c) Failing to detect buildings

(d) Partially detecting buildings (due to occlusion by trees)

Fig. 9. Examples of challenging cases for the YOLO-SAM algorithm.

5. Conclusion

The rapid and accurate extraction of building boundaries is essential for updating geographic databases in fast-growing urban areas. This study proposes an automated two-stage solution that integrates deep learning (YOLO-SAM) with geometric algorithms (Douglas-Peucker, smoothing filters) to refine building boundaries from high-resolution UAV imagery. Initial experiments on some UAV datasets demonstrate that this method significantly improves segmentation accuracy while reducing the time, cost, and manual effort required compared to traditional approaches. By leveraging YOLOv11's fast detection and SAM's precise segmentation, YOLO-SAM efficiently extracts buildings at scale. However, challenges remain due to variations in building structures, occlusions, and environmental conditions. While geometric post-processing enhances boundary quality, further refinements are needed to fully automate the process. Future improvements should focus on enhancing model generalization, optimizing post-processing techniques, and integrating additional data sources for greater accuracy in complex environments.

Acknowledgement

This research is funded by Le Quy Don Technical University Research Fund under the grant number 24.1.59.

References

[1] S. Yilei *et al.*, "BFGAN-building footprint extraction from satellite images", *2019 Joint Urban Remote Sensing Event (JURSE)*, IEEE, 2019. DOI: 10.1109/JURSE.2019.8809048

[2] H. Yuhan and Y. Jin, "Aerial imagery-based building footprint detection with an integrated deep learning framework: Applications for fine scale wildland-urban interface mapping", *Remote Sensing*, Vol. 14, No. 15, 2022. DOI: 10.3390/rs14153622

[3] D. Ivica *et al.*, "U-net ensemble for enhanced semantic segmentation in remote sensing imagery", *Remote Sensing*, Vol. 16, No. 12, 2024. DOI: 10.3390/rs16122077

[4] W. Alsabhan and T. Alotaiby, "Automatic building extraction on satellite images using Unet and ResNet50", *Computational Intelligence and Neuroscience*, Vol. 2022, Iss. 1, 2022. DOI: 10.1155/2022/5008854

[5] D. Runqin *et al.*, "Hybrid task cascade-based building extraction method in remote sensing imagery", *Remote Sensing*, Vol. 15, No. 20, 2023, 4907. DOI:10.3390/rs15204907

[6] L. Wenbo and S. Zhao, "Semantic segmentation of buildings in high-resolution remote sensing images based on DeepLabV3+ algorithm", *Journal of Physics: Conference Series*, Vol. 2400, No. 1, IOP Publishing, 2022. DOI: 10.1088/1742-6596/2400/1/012037

[7] G. Ningbo *et al.*, "NPSFF-Net: Enhanced building segmentation in remote sensing images via novel pseudo-siamese feature fusion", *Remote Sensing*, Vol. 16, No. 17, 2024, 3266. DOI: 10.3390/rs16173266

[8] C. Fangzhe *et al*., "Method for building segmentation and extraction from high-resolution remote sensing images based on improved YOLOv5ds", *PLoS ONE*, Vol. 20, No. 3, 2025, e0317106. DOI: 10.1371/journal.pone.0317106

[9] K. Rahima and M. Hussain, "YOLOv11: An overview of the key architectural enhancements", *arXiv preprint arXiv*:2410.17725, 2024. DOI: 10.48550/arXiv.2410.17725

[10] R. Nikhila *et al.*, "SAM2: Segment anything in images and videos", *arXiv preprint arXiv*:2408.00714, 2024. DOI: 10.48550/arXiv.2408.00714

[11] Z. Lei *et al.*, "Research on mine-personnel helmet detection based on multi-strategy-improved YOLOv11.", *Sensors*, Vol. 25, No. 1, 2024, 170. DOI: 10.3390/s25010170

[12] D. H. Douglas and T. K. Peucker, "Algorithms for the reduction of the number of points required to represent a digitized line or its caricature", *Cartographica: The International Journal for Geographic Information and Geovisualization*, Vol. 10, No. 2, 1973, pp. 112-122. DOI: 10.3138/FM57-6770-U75U-7727

[13] G. Meiling *et al.*, "A review of non-maximum suppression algorithms for deep learning target detection", *Seventh Symposium on Novel Photoelectronic Detection Technology and Applications*, Vol. 11763, SPIE, 2021. DOI: 10.1117/12.2586477

[14] CrowdAI mapping-challenge dataset. https://www.kaggle.com/datasets/kmader/synthetic-word-ocr [Accessed on March 30, 2025].

[15] Buildings instance segmentation dataset. https://universe.roboflow.com/roboflow-universe-projects/buildings-instance-segmentation [Accessed on March 30, 2025].

[16] R. Padilla *et al.*, "A survey on performance metrics for object-detection algorithms", *2020 International Conference on Systems, Signals and Image Processing (IWSSIP)*, 2020, pp. 237-242. DOI: 10.1109/IWSSIP48289.2020.9145130

[17] J. Avbelj *et al.*, "A metric for polygon comparison and building extraction evaluation", *IEEE Geoscience and Remote Sensing Letters*, Vol. 12, No. 1, pp. 170-174, 2015. DOI: 10.1109/LGRS.2014.2330695

PHÂN ĐOẠN TỰ ĐỘNG ĐỐI TƯỢNG NHÀ TỪ ẢNH TRỰC GIAO UAV SỬ DỤNG CÁC KỸ THUẬT HỌC SÂU VÀ GIS

Vũ Văn Trường[1], Trương Thị Thu Trang[2]

[1]*Viện Kỹ thuật công trình đặc biệt, Trường Đại học Kỹ thuật Lê Quý Đôn*

[2]*Khoa Ngoại ngữ, Trường Đại học Kỹ thuật Lê Quý Đôn*

Tóm tắt: Bài báo trình bày một mô hình tự động để phân đoạn đối tượng nhà trong ảnh trực giao chụp bởi máy bay không người lái (UAV), sử dụng học sâu và các kỹ thuật GIS. Phương pháp đề xuất gồm hai giai đoạn: Giai đoạn 1 áp dụng phương pháp học sâu kết hợp (có tên là YOLO-SAM), trong đó YOLOv11 được sử dụng để phân đoạn sơ bộ công trình, sau đó thuật toán SAM2 tinh chỉnh và nâng cao kết quả. Giai đoạn 2 xử lý kỹ hơn tệp shapefile bằng các kỹ thuật hậu xử lý dựa trên hình học, bao gồm thuật toán Douglas-Peucker để đơn giản hóa biên và làm mượt đường cong nhằm tối ưu hóa hình dạng tổng thể. Phương pháp này đạt độ chính xác phân đoạn cao (~mAP khoảng 0,75) và cung cấp một giải pháp thực tiễn, có khả năng mở rộng để lập bản đồ tại Việt Nam. Khác với các phương pháp trước đây, phương pháp này nổi bật nhờ sử dụng thuật toán kết hợp YOLOv11 và SAM2 để phân đoạn chính xác, chi tiết, và nhờ tích hợp thuật toán học sâu YOLO-SAM với các kỹ thuật hậu xử lý dựa trên GIS trong một quy trình thống nhất, hiệu quả.

Từ khóa: *Phân đoạn nhà; UAV; học sâu; GIS.*

Received: 02/04/2025; Revised: 27/05/2025; Accepted for publication: 25/06/2025

❑