Rethinking Language Models for Building Outline Extraction from Remote Sensing Imagery

Kuanren Qian* Yang He* Mohamed Moustafa
Amazon Last Mile
Bellevue, Washington, USA
{kuanqian,yanhea,mmoustm}@amazon.com

Abstract

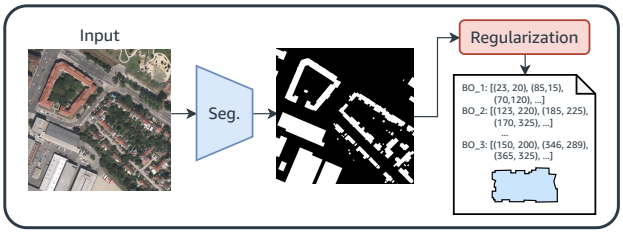
Building outline extraction from remote sensing imagery traditionally relies on segmentation or detection followed by post-processing to derive polygonal geometries. Despite advances in sequential prediction methods [2, 20], end-to-end extraction remains challenging, often missing buildings or requiring additional refinement steps.

In this work, we reformulate building outline extraction as next-coordinate prediction using decoder-only large language models (LLMs). We show that effective serialization of all building vertices within an image is critical for unified generation. To support spatial reasoning, we learn a distance-aware coordinate token space, and introduce a smoothed loss formulation to improve training stability and robustness to dataset noise. Our tailored LLM further incorporates coordinate token embeddings with triplet regularization to enforce spatial consistency, enabling direct vertex sequence generation. The proposed approach produces complete and concise building polygons directly from imagery without post-processing, outperforming state-of-the-art methods on four benchmarks (INRIA, SpaceNet2, CrowdAI, and WHU) and demonstrating stronger cross-dataset generalization. Extensive analysis highlights how LLMs can be fundamentally adapted for structured geometric generation beyond natural language tasks.

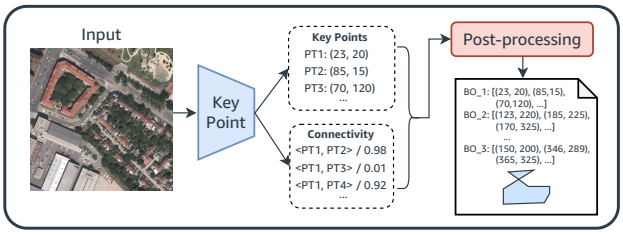
1. Introduction

From urban planning to disaster response, building footprint extraction from satellite imagery underpins numerous real-world applications [16, 22, 30]. Despite decades of research [10, 40], the task remains challenging due to urban complexity and spectral ambiguity.

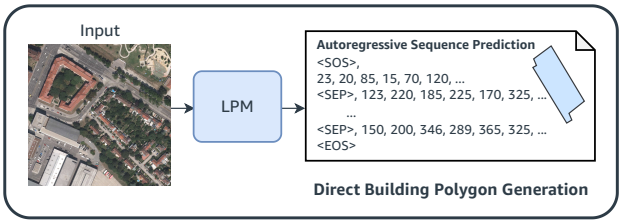
Most existing approaches adopt multi-stage pipelines: they first produce pixel-wise segmentation maps [8] or classification maps [2], and then rely on heuristic post-processing to convert intermediate network outputs into vectorized polygons [31]. Such indirect designs inevitably accumulate errors and often yield geometries with redundant details or unrealistic shapes. Although recent works attempt to simplify the pipeline using keypoints and connectivity modeling [2, 33, 41], a truly end-to-end frame-



(a) Mask-based methods



(b) Keypoint-based methods



(c) Direct building outline generation using our LPM

Figure 1. From multi-stage pipelines to direct polygon sequence generation. (a) Conventional mask-based and (b) keypoint-based methods require post-processing steps, whereas (c) our Large Polygon Language Model (LPM) directly generates building polygons as coordinate sequences using an LLM.

*These authors contributed equally.

work that directly outputs building polygons is still lacking.

In this work, we formulate building footprint extraction as a next-coordinate prediction problem using decoder-only large language models (LLMs) [25]. Inspired by the strong sequential modeling capability of LLMs [19, 26, 27], we reinterpret polygon generation as a sequence modeling task. While prior works explore sequential prediction for polygons [2, 20], they still depend on post-processing and cannot directly generate complete building outlines.

We argue that effective serialization of all polygon vertices within an image is key to end-to-end learning. Similar to how language models rely on consistent grammatical structures, polygon generation requires a consistent vertex ordering rule to capture geometric dependencies across buildings. Based on this insight, we design a tailored decoder-only architecture — the Large Polygon Language Model (LPM) — for direct building outline generation. As illustrated in Fig. 1, LPM treats building polygons as discrete coordinate token sequences conditioned on the image. With specialized coordinate tokenization, spatial encoding, and customized generation strategies, LPM directly outputs vectorized building outlines without post-processing. Moreover, we design a robust training regularization scheme for next-coordinate prediction to handle prevalent noisy annotations, where building outlines are not perfectly aligned with image boundaries. We summarize the main contributions as follows:

- The first successful end-to-end method for remote sensing building polygon extraction using LLMs;
- Novel architectural adaptations, including coordinate tokenization, robust training strategies, and customized sequence generation for geometric computer vision tasks; and
- State-of-the-art performance with consistent metric improvements across multiple benchmarks, along with detailed ablation studies, opening a new avenue for geometric computer vision.

2. Related Work

Mask-based Methods. Conventional deep learning approaches treat building extraction as pixel-wise semantic segmentation, which is leveraged to construct global-scale building outline datasets [40]. Foundational segmentation models like SAM [18] and SAM2 [28] provide segmentation capabilities across more classes, with SAMPoly-Build [31] specifically SAM for building extraction. However, the fundamental limitation remains: rasterized masks necessitate additional regularization steps [35] to generate vectorized polygons needed for applications. This step is non-differentiable and prone to error, breaking end-to-end learning and often come with artifacts that degrade polygon quality.

Keypoint-based Methods. Considering segmentation limitations, recent methods lean towards predicting polygon vector representation (Figure 1b). These approaches decompose polygon generation into a multi-step procedure with geometric primitives detection and connectivity-based assembly [12]. PolyWorld [41] uses Graph Neural Networks for vertex connectivity, P2PFormer [38] employs transformer attention for line segment assembly, and Poly-Building [14] introduces polygon queries for end-to-end detection. PolyRCNN [17] adapts region-based object detection frameworks for polygon prediction. The current SOTA, Pix2Poly [2], treats polygon generation as graph vertex connection problem, generating vertices and then connecting them (Figure 1b). However, this still requires solving two distinct sub-problems: vertex detection and connectivity. This decomposition can lead to connectivity ambiguities in complex scenarios with dense or overlapping buildings.

Foundation Models and Multimodal LLMs. Foundation models have revolutionized natural language processing and computer vision [4]. LLMs like GPT [6, 25] excel at autoregressive sequence generation and structured prediction through next-token prediction. Multi-modal LLMs extend these capabilities to vision-language reasoning [36]. CLIP [27] pioneered vision-language alignment, while LLaVA [19] and GPT-4V [1] demonstrate sophisticated visual understanding. However, research on adapting these methodologies to structured geometric tasks in remote sensing like polygon generation has not been explored.

Sequential Prediction for Polygon Extraction. Several efforts have been made to detect objects with a polygon representation. Pix2Poly [2] utilizes a transformer to generate all the vertex candidates in an orderless manner, however, it still relies on extra modules to connect predicted vertices, possibly resulting in unrealistic polygon shapes or merged polygons between neighboring buildings. PolyFormer [20] predicts polygon key vertices in an ordered manner, but it is designed for referring image segmentation with language descriptions. Therefore, it is not straightforward to extract all the building outlines in an image with their formulation. Recently, the concurrent work, MARS [37] leverages a decoder-only LLM to build a foundation model for map generation and is able to produce building polygons with user interactions.

Different from previous works, **the proposed LPM** aims to extract all the building outlines from an image with no auxillary information (e.g., user interactions, language descriptions) and post-processing steps. Furthermore, we dive deep into training details and present a tailored LLM for geometric generation, with a distance-aware coordinate token space and robust training strategies to data noise.

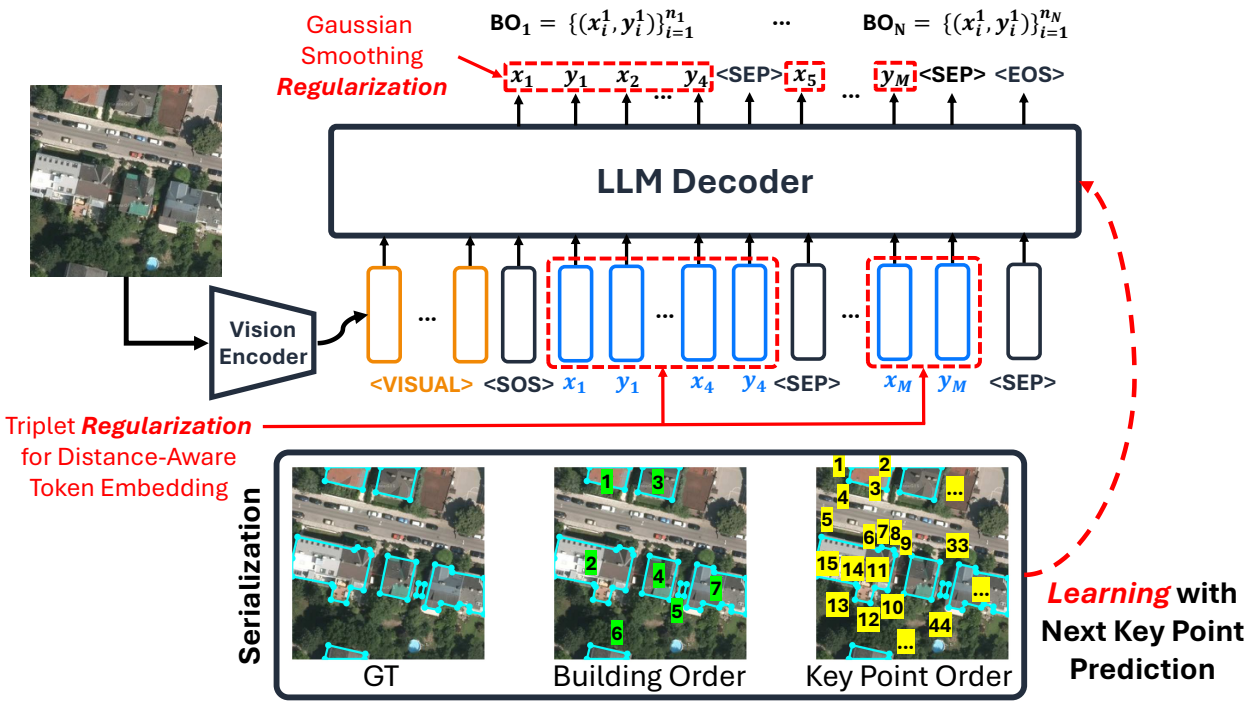


Figure 2. The diagram of the proposed LPM framework for end-to-end building outline extraction. We develop a tailored LLM with task-specific regularization terms to predict M key points from N buildings, which is achieved through serialization of all the key points. The LPM predicts all the key points autoregressively for the buildings $\mathrm{BO}_1, \cdots, \mathrm{BO}_N$.

3. Method

Large language models gain knowledge and paragraph understanding capability by learning from conversations, literatures, source codes, etc. Consistency in vocabulary, grammar, and semantics provides the foundation for learning powerful language models for various generative tasks. In this section, we depict our large polygon language model (a.k.a., LPM) for direct building outline extraction, which rethinks building extraction from the language generation perspective by seeking a rule for representing building outlines.

3.1. Direct Polygon Sequence Generation

LPM combines a vision encoder with a decoder-only language model. Our framework is adaptable to different vision backbones and decoder-only language models. In this paper, we leverage SAM2's vision encoder [28] and DistilledGPT2 [29], which has 82M parameters and serves as a lightweight language model. Because we do not aim at answering diverse questions or extracting thousands of object categories, we only employ DistilledGPT2 to showcase the effectiveness of our framework at low costs.

Although previous work [2] attempted sequential building polygon extraction, it produced unordered key points and failed to learn the logic of building outline generation. In contrast, Figure 2 illustrates the LPM architecture, where it takes encoded remote sensing image features as `<VISUAL>` tokens and generates all the building key points autoregressively with the language model. We split the key points from individual buildings, by inserting the token `<SEP>` between different buildings. The generation is triggered and stopped, when the tokens `<SOS>` / `<EOS>` are given (i.e., training) or predicted (i.e., inference). As a result, all the buildings in an image can be extracted from the predicted sequence, where the model is trained by a serialization step of all the building key points.

3.1.1. Serialization of building polygons

Consistent building polygon serialization is crucial, similar to language grammar [5] or programming logic [24]. Figure 2 shows how our method processes ground truths and leverages them to train a model for building outline generation. The serialization consists of 2 steps: determining the building orders, and determining the key point orders.

For the building outline orders, we calculate the minimum distance of each building to the top-left (TL) image corner, and then rank the minimum distance to determine the building orders. After that, we order the key points in a building in the clockwise manner, starting from the point with the minimum distance to the TL image corner. We then serialize all key points from different buildings into a single sequence. To indicate different building outlines, we apply `<SEP>` between different buildings. Therefore, a

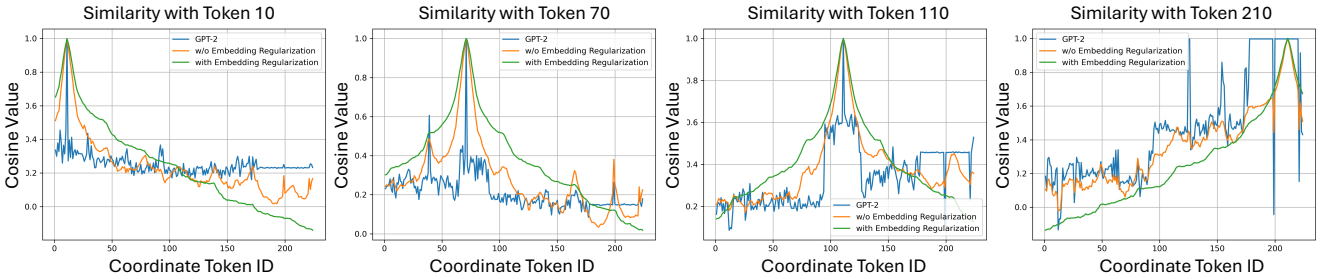


Figure 3. Similarity curves between token embeddings and the embedding of target locations. We clearly observe our triplet loss regularizes the coordinate token embedding space well, leading to a smoother and more reasonable token space.

ground truth token sequence is created by adding <SOS> and <EOS> tokens, for LLM training. For instance, the image in Figure 2 has 44 key points from 7 buildings. The serialization process creates a GT sequence [<SOS>, x_1, y_1, ..., x_4, y_4, <SEP>, x_5, y_5, ..., x_{44}, y_{44}, <SEP>, <EOS>] from the original GT outline.

From our study, it is observed that a consistent rule to represent building key points is important, which is similar to the consistent words, grammar, and paragraph structures for learning powerful language models in chatbot [9, 15] or AI coding system [7]. To verify the importance of a successful serialization, we provide ablation studies on different orders in Sec. 4.4.

3.2. Vision Encoder

We extract image tokens from remote sensing images to enable LLM inference. The extracted features are reshaped into 2D matrices of size $h_{LLM} \times K$, where h_{LLM} is the LLM hidden dimension and K is the total number of image embeddings.

The pretrained SAM2 vision encoder is applied to extract hierarchical features from 4 different stages, capturing both fine-grained details from early layers and high-level semantics from later layers. Given an image of size $H \times W$, we obtain features at stages $\{F_i\}_{i=1}^4$, where $F_i \in \mathbf{R}^{\frac{H}{k_i} \times \frac{W}{k_i}}$ and k_i is the downsampling factor for the i-th stage. To fuse multi-stage features, a fusion module is learned, which consists of several convolution blocks and concatenation of multi-stage features. The fusion module produces a dense feature map at the dimension of $h_{LLM} \times \frac{H}{16} \times \frac{W}{16}$, where h_{LLM} is the hidden dimension of the LLM and the spatial dimension is downsampled by 16. Furthermore, we also adopt PSP module [39] to aggregate hierarchical context information, which is popular in semantic segmentation. The PSP module outputs hierarchical features at the dimension of $h_{LLM} \times 1 \times 1$, $h_{LLM} \times 2 \times 2$, and $h_{LLM} \times 4 \times 4$.

Eventually, considering a 224×224 image as the example, 217 (i.e., $(\frac{224}{16})^2 + 1^2 + 2^2 + 4^2$) tokens, followed with LayerNorm, are fed into the LLM decoder (i.e., <VISUAL>) to generate building outlines. As DistilledGPT2 is adopted, where $h_{LLM} = 768$, our context tokens have the dimension of 768×217. This design enables seamless integration of visual features with language model processing, allowing the transformer to capture local geometric details and global spatial patterns, for better localization of building outlines.

3.3. LLM Decoder as the Polygon Generator

3.3.1. Coordinate Tokens

Unlike traditional language models that use a tokenizer to partition input sentences, our approach explicitly defines tokens to represent image indices for the polygon coordinate prediction. In addition to image token <VISUAL>, and special tokens <SOS>, <SEP>, and <EOS>, we define coordinate tokens as <INDEX=i>, where $i = 0, \cdots, N-1$ and N is the maximum image size. For a 224×224 image, our model needs 226 predefined tokens, by incorporating special tokens: <SOS> marks sequence start, <SEP> separates individual buildings, and <EOS> indicates sequence completion. Additionally, the coordinate tokens are shared across both x- and y-axes, and the LLM learns to predict coordinate sequences by alternating between x and y values in pairs, as illustrated in Figure 2.

3.3.2. Training

A fundamental limitation of pretrained LLMs on geometric tasks is that token embeddings do not sufficiently capture the distance property for the coordinates. Intuitively, the embedding distance between two nearby coordinates should be smaller than that between distant coordinates. For example, coordinate token "10" is supposed to be closer to "9" or "11" compared with "100" in the embedding space. However, such relationships are not explicitly modeled in many LLM pre-training or post-training. This poses a challenge for accurate coordinate prediction or results in less model robustness, as the model fails to provide strong continuity in coordinate embedding space. To address this limitation, we introduce specialized training innovations that enable a distance-aware token space for more accurate and robust generation.

Table 1. Comparison results against SOTA on four public datasets. We highlight **the best** and the second best methods.

Dataset	Model	IoU ↑	c-IoU ↑	$N_{ratio} = 1$	MTA ↓	PoLiS ↓	IoU^{topo} ↑	F_1^{topo} ↑	PA^{topo} ↑
INRIA *val*	FFL [11]	68.30	49.80	2.29	35.62	2.865	43.38	58.78	89.67
	HiSup [33]	74.90	66.10	1.13	43.86	2.438	53.51	67.94	93.20
	Pix2Poly [2]	79.46	71.73	1.08	34.31	1.914	61.08	74.29	94.37
	SAM2-UNet [32]	80.07	-	-	-	-	-	-	-
	Ours	**80.16**	**71.79**	**1.01**	**33.28**	**0.749**	**61.20**	**74.75**	**94.39**
SpaceNet2 *Vegas*	FFL	76.00	57.60	1.97	36.29	2.398	49.46	65.00	91.1
	HiSup	82.10	75.20	1.10	33.89	1.722	59.56	73.43	**93.8**
	Pix2Poly	81.81	75.05	1.04	33.40	1.717	60.31	74.20	**93.8**
	SAM2-UNet	71.20	-	-	-	-	-	-	-
	Ours	**83.80**	**78.15**	**0.99**	**31.42**	**1.255**	**60.43**	**74.59**	93.5
CrowdAI *val*	FFL	84.10	73.70	-	33.5	3.454	-	-	-
	HiSup	94.27	89.67	1.02	31.9	0.726	84.08	91.14	98.05
	Pix2Poly	95.03	89.85	1.11	23.1	0.479	89.05	93.75	**98.62**
	SAM2-UNet	94.99	-	-	-	-	-	-	-
	Ours	**95.11**	**93.15**	**0.99**	**18.3**	**0.328**	**89.07**	**93.88**	98.52
WHU *test*	FFL	77.61	32.19	5.09	35.27	1.783	56.59	70.02	94.01
	HiSup	87.12	79.62	1.15	34.75	1.158	72.11	82.47	96.80
	Pix2Poly*	87.60	81.38	1.10	32.34	0.935	75.50	86.20	96.91
	SAM2-UNet	85.63	-	-	-	-	-	-	-
	Ours	88.69	**83.01**	**0.98**	33.83	**0.648**	**77.24**	**86.52**	96.81

Notes: ↑/↓ indicate higher/lower is better. N_{ratio} closer to 1.0 indicates optimal vertex efficiency. SAM2-UNet only reports IoU as other metrics require vertex extraction steps. *Results are obtained by ourselves using authors' code and the same data split to ours.

Triplet Loss Regularization: We propose coordinate-aware embedding regularization that fundamentally transforms how LLMs understand spatial relationships. Our triplet loss creates geometrically meaningful coordinate embeddings by enforcing that spatially closer coordinates have higher embedding similarity. For coordinate tokens i, p, and q, where $|i - p| > |i - q|$ (p is farther from i than q):

$$\mathcal{L}_{\text{triplet}} = \max\left(0,\ \cos(E_i, E_q) - \cos(E_i, E_p)\right), \quad (1)$$

where E_i, E_p, and E_q are the embedding vectors for coordinate tokens <INDEX=i>, <INDEX=p>, and <INDEX=q> respectively. This encourages the anchor coordinate i to be more similar to the closer coordinate q than the farther coordinate p. Figure 3 shows that our triplet loss creates smooth, spatially reasonable embeddings, contrasting with standard GPT-2 token embeddings which exhibit fluctuations due to natural language pretraining.

Gaussian Smoothing: Ground-truth vertex locations extracted from raster masks often contain inevitable localization noise (e.g., imaging angles, vegetation occlusions), making hard target training suboptimal for coordinate prediction. We apply Gaussian smoothing to create soft target distributions that accommodate spatial uncertainty, enabling more robust coordinate learning. Note that the Gaussian smoothing is applied only to coordinate while keeping special tokens (<SOS>, <SEP>, <EOS>) in standard next token prediction. The smoothed cross-entropy loss becomes:

$$\mathcal{L}_{\text{smooth}} = -\sum_{i=1}^{N}\sum_{j=0}^{V-1} p_{ij}\log(\text{softmax}(z_{ij})), \quad (2)$$

where N is the sequence length, V is the vocabulary size, z_{ij} are the model logits, and p_{ij} follows a Gaussian distribution centered at the ground truth coordinate. This probabilistic approach improves training stability by accommodating annotation uncertainty. Consequently, our final loss function becomes:

$$\mathcal{L}_{\text{final}} = \mathcal{L}_{\text{smooth}} + \lambda_t \mathcal{L}_{\text{triplet}}. \quad (3)$$

where λ_t controls the triplet regularization strength. We employ Eq. (3) as the default loss to train our language model under the next token prediction formulation.

Two-Phase Training Strategy: We employ a two-phase training strategy with pretrained SAM2 and DistilledGPT2 models. In phase 1, we only train the multi-scale fusion component by freezing the SAM2 vision encoder and the DistilledGPT2 decoder, for 10 epochs with a base learning rate $2e^{-3}$. This allows the fusion component to bridge visual and language modalities, resulting in a better initialization for phase 2. In phase 2, we use a base learning rate for $1e^{-3}$ to train the entire model end-to-end for 240 or 90 epochs, depending on the size of datasets.

3.3.3. Inference

The model passes an image into the vision encoder to obtain <VISUAL>. Then, with a <SOS> token, it starts to generate the output sequence token-by-token until an <EOS> token is generated or the maximum length is reached. After that, we partition the generated sequence, at the separator token <SEP>, into a list of sub-sequences, where each sub-sequence represents the key points of a building outline in the clockwise order of consecutive (x, y) pairs. In de-

Table 2. Comparison against Pix2Poly [2] under cross-domain scenarios. We test the models trained with larger datasets on smaller ones. We highlight **the best** and the second best methods.

Train	Test	Model	IoU ↑	c-IoU ↑	$N_{ratio} = 1$	MTA ↓	PoLiS ↓	IoU^{topo} ↑	F_1^{topo} ↑	PA^{topo} ↑
INRIA	WHU	Pix2Poly	80.90	74.23	**0.973**	35.69	1.379	**66.42**	**78.49**	**94.84**
		LPM (Ours)	**81.85**	**74.61**	0.930	**35.55**	**0.957**	64.42	77.54	94.26
	SpaceNet2	Pix2Poly	**74.34**	**67.17**	1.070	**31.42**	2.693	**47.40**	**62.94**	**90.85**
		LPM (Ours)	72.09	64.42	**1.006**	42.16	**1.871**	40.83	56.81	88.31
CrowdAI	WHU	Pix2Poly	32.89	26.78	1.213	-	2.633	24.90	35.26	90.39
		LPM (Ours)	**70.51**	**63.34**	**0.924**	**33.01**	**1.676**	**52.85**	**65.54**	**93.31**
	SpaceNet2	Pix2Poly	75.90	65.62	1.316	32.11	1.736	55.48	68.90	93.66
		LPM (Ours)	**83.62**	**75.71**	**1.174**	**29.39**	**1.655**	**61.54**	**74.91**	**94.97**

Notes: ↑/↓ indicate higher/lower is better. N_{ratio} closer to 1.0 indicates optimal vertex efficiency.

Table 3. Ablation study on polygon vertex and building serialization strategies. We serialize building outlines or vertices, starting from a random one or using the proposed method to start from the top-left (TL).

First Vertex	Building Order	SpaceNet2 IoU ↑	SpaceNet2 PoLiS ↓	SpaceNet2 F_1^{topo} ↑	WHU IoU ↑	WHU PoLiS ↓	WHU F_1^{topo} ↑	INRIA (20%) IoU ↑	INRIA (20%) PoLiS ↓	INRIA (20%) F_1^{topo} ↑
Random	Random	42.19	2.241	44.62	46.71	0.780	51.09	70.72	**2.027**	66.46
Random	TL	47.10	4.422	46.06	45.59	1.576	84.70	74.62	2.075	68.84
TL	Random	83.64	1.367	74.34	88.10	0.791	86.05	72.86	2.078	67.30
TL	TL	**83.80**	**1.255**	**74.56**	**88.69**	**0.648**	**86.52**	**75.33**	2.096	**69.30**

Table 4. Ablation study on coordinate label smoothing regularizations.

Smoothing Type	SpaceNet2 IoU ↑	SpaceNet2 PoLiS ↓	SpaceNet2 F_1^{topo} ↑	WHU IoU ↑	WHU PoLiS ↓	WHU F_1^{topo} ↑	INRIA (20%) IoU ↑	INRIA (20%) PoLiS ↓	INRIA (20%) F_1^{topo} ↑
None (Baseline)	**83.77**	1.403	**74.59**	88.26	**0.697**	86.07	74.98	**2.056**	69.19
Gaussian ($\sigma = 0.5$)	83.67	**0.955**	74.46	**88.71**	0.902	**86.45**	75.05	2.059	69.11
Gaussian ($\sigma = 1.0$)	83.74	1.303	74.57	88.40	0.779	86.23	**75.44**	2.078	**69.50**
Gaussian ($\sigma = 2.0$)	83.64	1.418	74.45	88.25	0.978	86.04	75.29	2.073	69.12

tail, we use nucleus sampling [13] with $top_p = 0.99$ and a temperature $T = 0.01$. We set the repetition penalty to 1.0 (no penalty) to allow repetition, since same coordinate value may appear multiple times (i.e., vertical or horizontal buildings).

3.4. Implementation

We use the AdamW optimizer [21] with weight decay of 0.01. The model is trained on 8 V100 GPUs using distributed data parallel with batch size of 6 per GPU and gradient accumulation steps of 8. We apply gradient clipping with maximum norm of 1.0 and use cosine learning rate scheduling with 8% warmup startup. Loss component weights are set to $\lambda_t = 5.0$ for triplet regularization. Gaussian smoothing employs $\sigma = 0.5 - 2.0$ in our test.

4. Experiments

4.1. Dataset

We conduct comparison experiments on four public datasets for building polygon extraction: (1) the INRIA Aerial Image Labeling dataset [22], which contains 360 high-resolution tiles (0.3 m spatial resolution), using the official train/validation split; (2) the SpaceNet2 dataset [30] (Las Vegas subset), following [2], with 52,374 images for training and 9,242 for testing; (3) the CrowdAI dataset [23], comprising 280,741 training images and 60,317 testing images; and (4) the WHU Building dataset [16], which includes aerial imagery from New Zealand with 4,736/1,036/2,416 images (of size 512×512) for training, validation, and testing, respectively. For all datasets, we crop the raw images into 224×224 patches for both training and evaluation, following the protocol of prior work [2]. Due to the substantially larger scale of the INRIA and CrowdAI datasets compared to SpaceNet2 and WHU, we train for 90 epochs on INRIA and CrowdAI, and for 240 epochs on SpaceNet2 and WHU.

4.2. Evaluation Metrics

For pixel-wise accuracy, we use intersection over union (IoU) and complexity-aware IoU ($c - IoU$) [41] to measure overlap between predicted and ground truth building outlines. For geometric quality, we use polygons and line segments ($PoLiS$) [3] to measure average error between predicted vertices and ground truth, Max Tangent Angle Error (MTA) [11] to cover polygon corner regularity, N_{ratio}

Table 5. Ablation study on individual and combined loss function components in LPM training.

Label Smoothing	Triplet	SpaceNet2			WHU			INRIA (20%)		
		IoU ↑	PoLiS ↓	F_1^{topo} ↑	IoU ↑	PoLiS ↓	F_1^{topo} ↑	IoU ↑	PoLiS ↓	F_1^{topo} ↑
		83.77	1.403	**74.59**	88.26	0.697	86.07	74.81	2.079	69.10
✓		83.67	**0.955**	74.46	**88.71**	0.902	86.45	75.00	2.098	68.98
	✓	83.65	1.381	74.52	88.69	**0.648**	**86.52**	74.46	**2.078**	68.70
✓	✓	**83.80**	1.255	74.56	88.38	0.691	86.18	**75.28**	2.112	**69.44**

to compare predicted and ground truth vertex counts, and topological metrics (IoU_{topo}, $F1_{topo}$, PA_{topo}) based on buffers that penalize incorrect topologies [34].

4.3. Comparison with SOTA

Table 1 shows that LPM achieves SOTA performance, compared with popular methods. On INRIA, we reach 80.16% IoU compared with the previous best from SAM2-UNet. On SpaceNet2, we reach 83.80% IoU versus HiSup's previous best. On CrowdAI, we reach 95.11% IoU with 93.15% complexity-aware IoU. On WHU, we obtain 88.69% IoU, competitive with the 89.15% achieved by Pix2Poly, while significantly outperforming our reproduced baseline of 87.60%. Beyond IoU, LPM also improves geometric quality. Most notably, our INRIA $PoLiS$ score of 0.749 represents a 61% improvement over Pix2Poly's 1.914, indicating significantly improved polygon outline accuracy. Across all datasets, LPM maintains near-optimal vertex counts ($N_{ratio} \approx 1.0$) and achieves superior topological correctness, with strong results on WHU and CrowdAI for topological metrics.

Direct generation advantage. Unlike segmentation methods requiring complex post-processing [8, 31] or keypoint methods needing multi-stage vertex detection and assembly [41], LPM generates complete ordered polygon sequences directly. This architectural advantage eliminates error accumulation from multi-stage pipelines and produces geometrically consistent outputs with appropriate vertex counts ($N_{ratio} \approx 1.0$ across all datasets), reducing storage requirements and enabling efficient geometric operations.

Inference efficiency. LPM achieves 2.09 fps per V100 GPU, scaling linearly to 16.70 fps across 8 GPUs. INRIA validation dataset inference requires only 20 minutes on 8 V100-16GB GPUs, compared to 1.5 hours for keypoint-based approaches [2] from our test. This efficiency stems from direct vertex prediction rather than keypoint detection and assembly, eliminating bottlenecks.

4.4. Ablation Studies

Robustness in cross-dataset evaluation. In practice, domain shift from training to testing data commonly exists. To understand the model robustness, we compare the proposed LPM with the previous best building extraction model (i.e., Pix2Poly [2]) in the cross-domain scenario, that evaluates the performance of WHU and SpaceNet2 using the models trained on INRIA and CrowdAI, as listed in Table 2. From this table, we can see that our model is affected less by the domain shift, compared with Pix2Poly. Even though Pix2Poly is slightly better than LPM in some metrics, in particular for the setting "INRIA→SpaceNet2", our method works more stably across all the settings. For example, the model trained on CrowdAI performs still well in WHU and SpaceNet2, but Pix2Poly degenerates tremendously (e.g., the IoU score is reduced to 32.89 on WHU). We believe that our LLM encodes the dependency between different vertices and captures the nature of a building outline, where our model tries to produce a closed region with vertices in the clockwise manner. The acquisition of building outline knowledge helps us achieve more robust results across different datasets.

Importance of serialization. Table 3 compares different orderings for both starting vertex and buildings, using random or our proposed order described in Sec. 3.1.1. Random order yields poorest performance across all datasets, indicating that serialization is essential. In contrast, starting from Top-Left (TL) performs significantly better, that TL&TL proves most effective in general on the detection accuracy (i.e., IoU) and outline quality (i.e., PoLiS and F_1^{topo}), achieving the best performance on all the datasets. Besides, vertex ordering exhibits stronger influence than building ordering, that TL&Random only drops a little on SpaceNet2/WHU but Random&TL degenerates significantly. Consequently, this experiment clearly demonstrates the effectiveness of our approach and provides evidence supporting our contribution.

Effect of coordinate label smoothing. Table 4 reveals dataset-dependent optimal smoothing configurations. WHU benefits from modest smoothing ($\sigma = 0.5$) achieving best IoU performance, while SpaceNet2 shows substantial boundary quality improvement with $\sigma = 0.5$ as PoLiS improves from 1.403 to 0.955 despite slightly lower IoU. INRIA with more annotation misalignment requires stronger smoothing ($\sigma = 1.0$) for optimal performance. The smoothing strategy demonstrates the correlation to annotation quality, critical for training on real-world data. Generally speaking, we recommend increasing the Gaussian variance value if the key points or building outline edges do not align with training images very well. Otherwise, a smaller variance should be adopted.

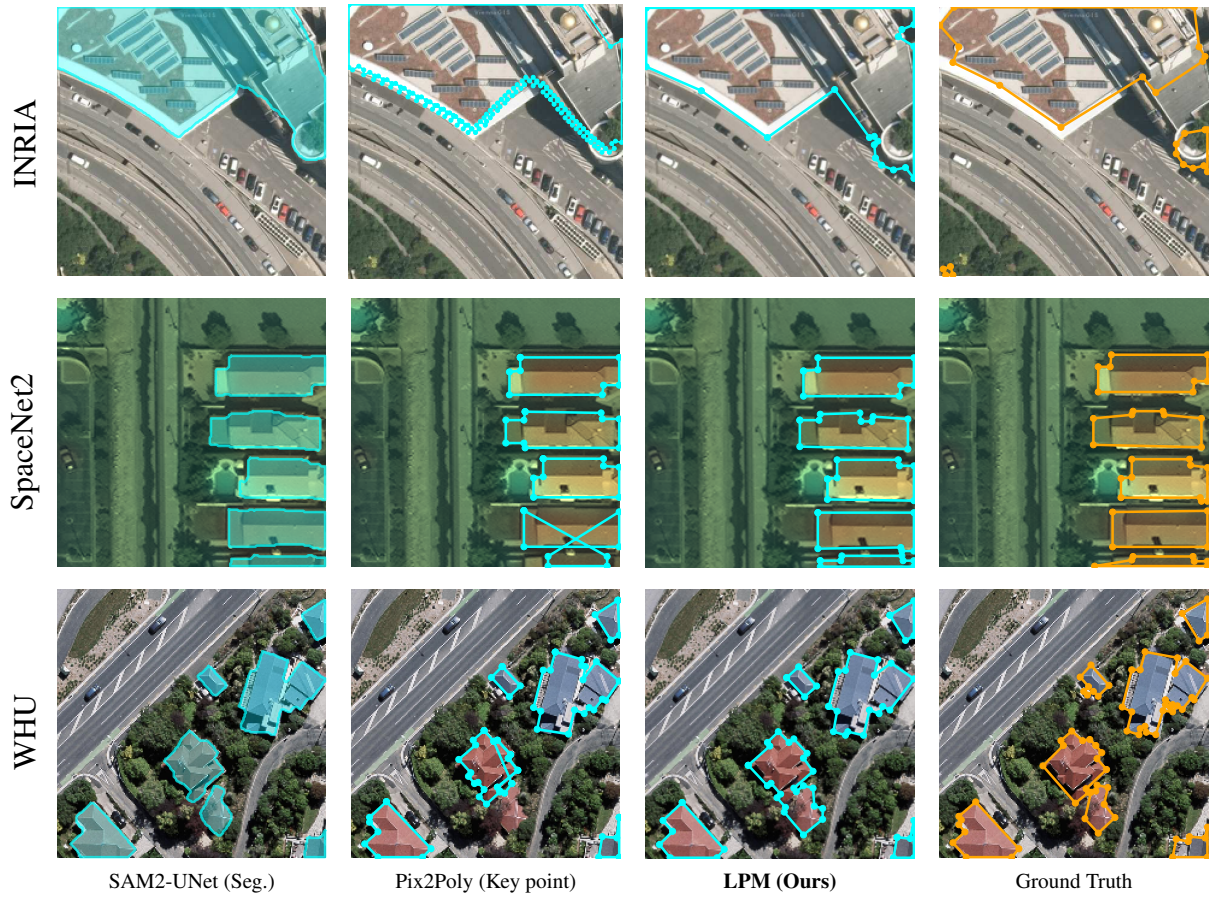


Figure 4. Qualitative comparison across three popular datasets. Cyan outlines are predictions, orange outlines are ground truth. LPM produces cleaner polygons with fewer vertices while maintaining accurate building coverage.

Loss function components. Table 5 reveals the contribution of each component. Label smoothing provides boundary quality improvements on SpaceNet2 where PoLiS improves dramatically from 1.403 to 0.955. Triplet loss consistently delivers better geometric quality, achieving the best PoLiS score of 0.648 on WHU, which demonstrates our motivation to learn a distance-aware coordinate space. The combined approach achieves improved IoU performance of 83.80% on SpaceNet2 and 75.28% on INRIA while maintaining competitive geometric quality across all datasets.

4.5. Qualitative Analysis

Figure 4 presents qualitative results from our LPM model. The examples demonstrate that LPM successfully generates high-quality building polygons that closely match the ground truth, achieving high IoU scores. Our method is capable of handling a variety of building shapes, from simple rectangles to more complex L-shaped and multi-part structures. The generated polygons exhibit sharp, well-defined corners and straight edges, confirming the effectiveness of our approach in learning the geometric regularities of man-made structures. While the model performs robustly across diverse scenes, some limitations are observed in dense urban areas with overlapping buildings or heavy occlusion from vegetation, hinting valuable future work directions.

5. Conclusion

In this paper, we introduce LPM, a decoder-only large language model that fundamentally reimagines building outline extraction from remote sensing imagery as direct autoregressive sequence generation. Through specialized coordinate tokenization, triplet loss regularization, and tailored training strategies, LPM directly generates building polygons from images, eliminating widely used post-processing steps. Our method achieves state-of-the-art performance across four benchmark datasets, with notable improvements in geometric quality. Through extensive ablation studies, we demonstrate the effectiveness of our overall architecture and the contributions of individual design components in LLM training. We believe that our work opens new directions for applying LLMs to structured geometric generation and reasoning tasks, with potential applications in object detection, segmentation, and geospatial analysis.

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023. 2

[2] Yeshwanth Kumar Adimoolam, Charalambos Poullis, and Melinos Averkiou. Pix2poly: A sequence prediction method for end-to-end polygonal building footprint extraction from remote sensing imagery. In *WACV*, 2024. 1, 2, 3, 5, 6, 7

[3] Janja Avbelj, Rupert Müller, and Richard Bamler. A metric for polygon comparison and building extraction evaluation. *IEEE Geoscience and Remote Sensing Letters*, 12(1):170–174, 2014. 6

[4] Rishi Bommasani. On the opportunities and risks of foundation models. *arXiv preprint arXiv:2108.07258*, 2021. 2

[5] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *NeurIPS*, 2020. 3

[6] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020. 2

[7] Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde De Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, et al. Evaluating large language models trained on code. *arXiv preprint arXiv:2107.03374*, 2021. 4

[8] Bowen Cheng, Ishan Misra, Alexander G Schwing, Alexander Kirillov, and Rohit Girdhar. Masked-attention mask transformer for universal image segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1290–1299, 2022. 1, 7

[9] Gheorghe Comanici, Eric Bieber, Mike Schaekermann, Ice Pasupat, Noveen Sachdeva, Inderjit Dhillon, Marcel Blistein, Ori Ram, Dan Zhang, Evan Rosen, et al. Gemini 2.5: Pushing the frontier with advanced reasoning, multimodality, long context, and next generation agentic capabilities. *arXiv preprint arXiv:2507.06261*, 2025. 4

[10] Mohamed Barakat A Gibril, Rami Al-Ruzouq, Abdallah Shanableh, Ratiranjan Jena, Jan Bolcek, Helmi Zulhaidi Mohd Shafri, and Omid Ghorbanzadeh. Transformer-based semantic segmentation for large-scale building footprint extraction from very-high resolution satellite images. *Advances in Space Research*, 73(10):4937–4954, 2024. 1

[11] Nicolas Girard, Dmitriy Smirnov, Justin Solomon, and Yuliya Tarabalka. Polygonal building extraction by frame field learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5891–5900, 2021. 5, 6

[12] Yang He, Ravi Garg, and Amber Roy Chowdhury. Td-road: Top-down road network extraction with holistic graph construction. In *ECCV*, 2022. 2

[13] Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. In *International Conference on Learning Representations (ICLR)*, 2020. 6

[14] Yuan Hu, Zhibin Wang, Zhou Huang, and Yu Liu. Polybuilding: Polygon transformer for building extraction. *ISPRS Journal of Photogrammetry and Remote Sensing*, 199:15–27, 2023. 2

[15] Aaron Hurst, Adam Lerer, Adam P Goucher, Adam Perelman, Aditya Ramesh, Aidan Clark, AJ Ostrow, Akila Welihinda, Alan Hayes, Alec Radford, et al. Gpt-4o system card. *arXiv preprint arXiv:2410.21276*, 2024. 4

[16] Shunping Ji, Shiqing Wei, and Meng Lu. Fully convolutional networks for multi-source building extraction from an open aerial and satellite imagery dataset. *IEEE Transactions on Geoscience and Remote Sensing*, 57(2):1349–1363, 2018. 1, 6

[17] Weiqin Jiao, Claudio Persello, and George Vosselman. PolyR-CNN: R-CNN for end-to-end polygonal building outline extraction. *ISPRS Journal of Photogrammetry and Remote Sensing*, 218:33–43, 2024. 2

[18] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. Segment anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4015–4026, 2023. 2

[19] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36:34892–34916, 2023. 2

[20] Jiang Liu, Hui Ding, Zhaowei Cai, Yuting Zhang, Ravi Kumar Satzoda, Vijay Mahadevan, and R Manmatha. Polyformer: Referring image segmentation as sequential polygon generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18653–18663, 2023. 1, 2

[21] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017. 6

[22] Emmanuel Maggiori, Yuliya Tarabalka, Guillaume Charpiat, and Pierre Alliez. Can semantic labeling methods generalize to any city? the INRIA aerial image labeling benchmark. In *2017 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*, pages 3226–3229. IEEE, 2017. 1, 6

[23] Sharada Prasanna Mohanty, Jakub Czakon, Kamil A Kaczmarek, Andrzej Pyskir, Piotr Tarasiewicz, Saket Kunwar, Janick Rohrbach, Dave Luo, Manjunath Prasad, Sascha Fleer, et al. Deep learning for understanding satellite imagery: An experimental survey. *Frontiers in Artificial Intelligence*, 3, 2020. 6

[24] Erik Nijkamp, Bo Pang, Hiroaki Hayashi, Lifu Tu, Huan Wang, Yingbo Zhou, Silvio Savarese, and Caiming Xiong. Codegen: An open large language model for code with multi-turn program synthesis. *arXiv preprint arXiv:2203.13474*, 2022. 3

[25] Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. Improving language understanding by generative pre-training. 2018. 2

[26] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 2019. 2

[27] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PmLR, 2021. 2

[28] Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, et al. Sam 2: Segment anything in images and videos. *arXiv preprint arXiv:2408.00714*, 2024. 2, 3

[29] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. In *NeurIPS EMC2 Workshop*, 2019. 3

[30] Adam Van Etten, Daniel Lindenbaum, and Todd M Bacastow. Spacenet: A remote sensing dataset and challenge series. *arXiv preprint arXiv:1807.01232*, 2018. 1, 6

[31] Chenhao Wang, Jingbo Chen, Yu Meng, Yupeng Deng, Kai Li, and Yunlong Kong. SAMPolyBuild: Adapting the segment anything model for polygonal building extraction. *ISPRS Journal of Photogrammetry and Remote Sensing*, 218: 707–720, 2024. 1, 2, 7

[32] Xinyu Xiong, Zihuang Wu, Shuangyi Tan, Wenxue Li, Feilong Tang, Ying Chen, Siying Li, Jie Ma, and Guanbin Li. Sam2-unet: Segment anything 2 makes strong encoder for natural and medical image segmentation. *arXiv preprint arXiv:2408.08870*, 2024. 5

[33] Bowen Xu, Jiakun Xu, Nan Xue, and Gui-Song Xia. Hisup: Accurate polygonal mapping of buildings in satellite imagery with hierarchical supervision. *ISPRS Journal of Photogrammetry and Remote Sensing*, 198:284–296, 2023. 1, 5

[34] Bingnan Yang, Mi Zhang, Zhan Zhang, Zhili Zhang, and Xiangyun Hu. TopDiG: Class-agnostic topological directional graph extraction from remote sensing images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1265–1274, 2023. 7

[35] Min Yang, Renwei Zou, Tinghua Ai, and Xiongfeng Yan. Regularizing building outlines extracted from remote sensing images by integrating multiple algorithms. *Geocarto International*, 39(1):2370322, 2024. 2

[36] Shukang Yin, Chaoyou Fu, Sirui Zhao, Ke Li, Xing Sun, Tong Xu, and Enhong Chen. A survey on multimodal large language models. *National Science Review*, 11(12): nwae403, 2024. 2

[37] Qi Zhang, Suvam Bag, Rupanjali Kukal, Mikael Figueroa, Rishi Madhok, Nikolaos Karianakis, and F. Yu. Mars: A foundational map auto-regressor. In *ICLR*, 2026. 2

[38] Tao Zhang, Shiqing Wei, Yikang Zhou, Muying Luo, Wenling Yu, and Shunping Ji. P2pformer: A primitive-to-polygon method for regular building contour extraction from remote sensing images. *IEEE Transactions on Geoscience and Remote Sensing*, 2024. 2

[39] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *CVPR*, pages 2881–2890, 2017. 4

[40] Xiao Xiang Zhu, Qingyu Li, Yilei Shi, Yuanyuan Wang, Adam J Stewart, Jonathan Prexl, and Fahong Zhang. Globalbuildingmap—unveiling the mystery of global buildings. *Scientific Data*, 2026. 1, 2

[41] Stefano Zorzi, Shabab Bazrafkan, Stefan Habenschuss, and Friedrich Fraundorfer. PolyWorld: Polygonal building extraction with graph neural networks in satellite images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1848–1857, 2022. 1, 2, 6, 7