Regularized Primitive Graph Learning for Unified Vector Mapping

Lei Wang[1], Min Dai[1], Jianan He[1], Jingwei Huang[1]
[1]Riemann Lab, Huawei Technologies.
{leiwang4, daimin11, hejianan2, huangjingwei6}@huawei.com

Abstract

Large-scale vector mapping is the foundation for transportation and urban planning. Most existing mapping methods are tailored to one specific mapping task, due to task-specific requirements on shape regularization and topology reconstruction. We propose GraphMapper, a unified framework for end-to-end vector map extraction from satellite images. Our key idea is using primitive graph as a unified representation of vector maps and formulating shape regularization and topology reconstruction as primitive graph reconstruction problems that can be solved in the same framework. Specifically, shape regularization is modeled as the consistency between primitive directions and their pairwise relationship. Based on the primitive graph, we design a learning approach to reconstruct primitive graphs in multiple stages. GraphMapper can fully explore primitive-wise and pairwise information for shape regularization and topology reconstruction, resulting improved primitive graph learning capabilities. We empirically demonstrate the effectiveness of GraphMapper on two challenging mapping tasks for building footprints and road networks. With the premise of sharing the majority design of the architecture and a few task-specific designs, our model outperforms state-of-the-art methods in both tasks on public benchmarks. Our code will be publicly available.

1. Introduction

Up-to-date vector maps are essential for navigation and urban planning. Methods to automatically extract vector maps from aerial or satellite images have greatly advanced in recent years. However, state-of-the-art vector mapping methods are tailored for one specific type of target [39, 18, 30, 16]. Consequently, multiple models must be maintained for comprehensive mapping systems, which increases the burden of model development and limits the system extensibility.

Designing a unified method for multiple vector mapping tasks is challenging due to the difficulties in processing multiple types of geometric primitives. The method must generate shapes with precise location, well-regularized geometry, and correct topology, which are the three fundamental requirements for vector mapping. However, existing end-to-end methods either only focus on refining location accuracy [23], or learnable topology reconstruction [39, 34]; end-to-end shape regularization is still under explored.

Shape regularization in vector mapping is essentially reducing the variation of relative relationship between primitives; for example, the angles between line segments of a building polygon usually share a few distinctive values (i.e, 0°, 90°, etc.). Conventional methods rely on heuristics-based rules [10, 1, 31] to generate regularized shapes, which are not flexible and requires extensive tuning. Recent methods regularize shapes through contour optimization or deformation [16, 25, 12, 10, 39]. However, these methods are not utilizing global shape structure, which we think is essential for accurate and flexible shape regularization. None of these methods are explicitly enforce low variation of primitive relationships.

In this paper, we use primitive graph as a generic representation to build a *unified framework*, GraphMapper, for multi-type vector mapping. GraphMapper incrementally learns to refine primitives' locations and reconstruct their pairwise relationships. Effective shape regularization and topology reconstruction are achieved through the relationship classification of pairwise primitives. With our design, most existing mapping tasks can be converted to image-based primitive graph reconstruction tasks. As shown in Fig. 1, GraphMapper is mainly composed of a convolutional visual feature encoder and two primitive learning structures (PLS) using multi-head attention (MHA) network. We first extract visual features of input image using a convolutional encoder and sample primitives (i.e., points, line segments) from the segmentation maps and key points. Then, we refine the primitives' coordinates and predict their geometric directions, which uses a PLS for local and global shape context modeling. Finally, we use another PLS to classify the relationships of primitive pairs.

We apply GraphMapper to two typical mapping tasks: building and road mapping. For building, we find line segment primitives and their topology from segmentation re-

sults, and predict inlineness between line segments; the network learns shape regularization by enforcing consistency between the network predicted angle matrix and pairwise relationship matrix of line segments. For road, we predict point primitives and reconstruct road topology by classifying their pairwise connectivity relationship.

By sharing most network components with few task-specific designs, GraphMapper outperforms state-of-the-art methods in both tasks in public benchmarks. In summary, our main contributions are the following:

- We propose a unified learning framework based on primitive graph, which can be trained end-to-end and adapt to multi-type vector mapping tasks.
- We propose an adaptive shape regularization module that learns to balance between shape regularity and location accuracy through enforcing internal consistency of primitive graphs.
- We provide adaptations of our learning framework for building and road vector mapping, and achieved state-of-the-art performance on both tasks.

2. Related Works

Unified vector mapping. To the best of our knowledge, PolyMapper [22] is currently the only unified learning method for vector mapping. PolyMapper uses polygons as the unified representation for buildings and roads. A CNN-RNN structure is used to recurrently predict point sequences. However, representing roads as polygons leads to redundant points, and predicting point sequences can easily introduce geometric errors. In contrast, GraphMapper supports learning on different primitives, and holistically reconstructs the topology of sampled primitives.

Building mapping and shape regularization. Building mapping methods are now focusing on learning vector results from image inputs, where regularized shape representation is often a major concern. Conventional methods often rely on heuristic rules [1, 31] for shape regularization, which is limited to simple scenarios and requires extensive tuning. PolygonRNN and its extensions [9, 2, 22] achieved end-to-end vector mapping from images by recurrently predicting point sequences. However, RNN-based methods often fail to produce regularized shapes due to the lack of shape optimization design.

[25, 12, 16] try to optimize contours using ACM (Active Contour Model) [19] guided by learned semantic information. Specifically, DSAC [25] and DARNNet [12] regress the weights of the contour smoothness term based on semantic information for shape generalization, which tends to generate under-regularized shapes due to inadequate design in simplified representation; [16] further optimizes boundary skeletons to align with learned frame fields, but still requires complicated post-processing for shape regularization. Instead, PolygonCNN [10] and Polygon Transformer [23] predict point deformation of segmented contours, in which the results are regularized implicitly by improving the accuracy of building vertices; comparatively, our approach goes one step further by explicitly modeling shape regularity. Recently, PolyWorld [39] achieved great performance by first deforming key points' coordinates, and then reconstructing topology through classifying pairwise point connectivity relationships. Despite that GraphMapper follows a similar structure, the relationship classification module in our framework serves multiple purposes: topology reconstruction for road mapping, and explicit shape regularization for building mapping.

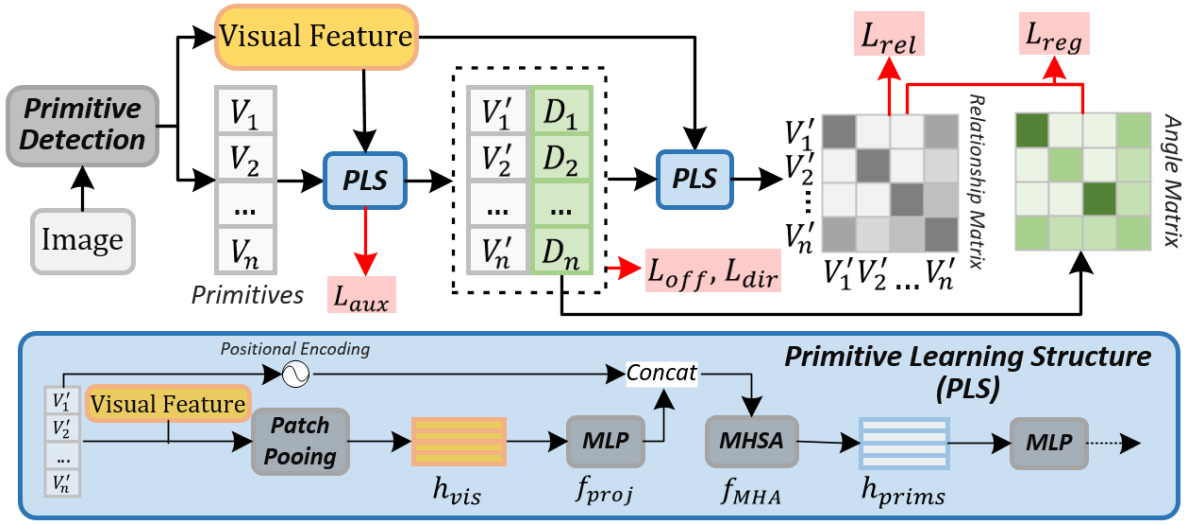
[40] tried to regularize building footprint segmentation masks using a GAN loss, which is complementary to our geometry-based regularization method.

Road mapping and topology reconstruction. Road mapping methods focus on improving topological correctness for navigation purposes. Various techniques are developed to improve the connectivity of road segmentation maps [27, 36, 29, 5, 5, 14]. Some other methods try to reconstruct road topology from defective segmentation results. [26] uses a binary decision classifier to predict the correctness of connections of nearby road endpoints from their image features, but it does not explore global road structure or shape prior. Graph-based methods [4, 32, 22] reconstruct road networks by iterative searching of the next point in the road graph, using a CNN or CNN-RNN structure. Shape prior and global road structure is implicitly modeled in the iterative searching process. Compared to graph-based methods, our method generates a road graph in one forward run with all road points available, which allows easier integration of global and local shape contextual information.

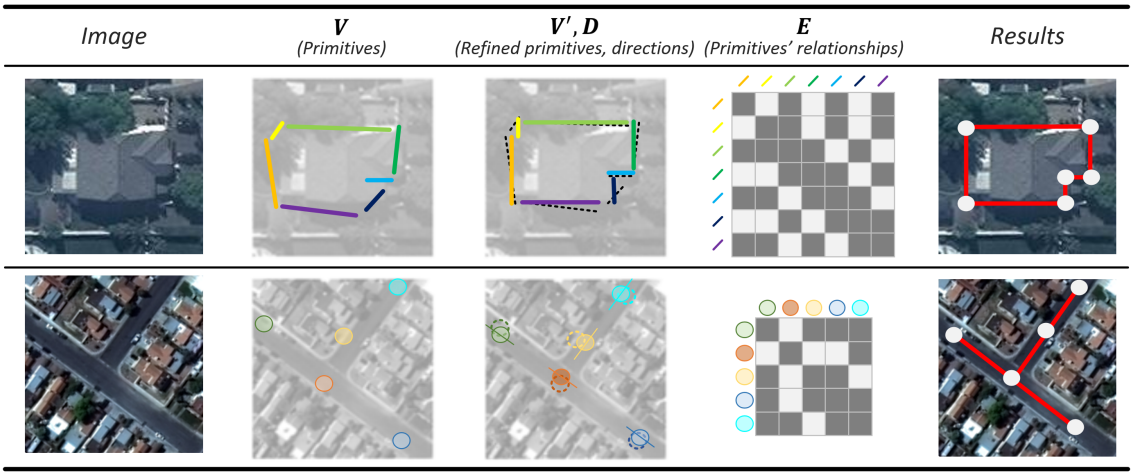
Recently, Sat2Graph [18] achieved state-of-the-art performance by connecting segmented key points along road direction with carefully designed searching rules. Uniformly distributed road points and their directions are predicted using a multi-task CNN. Several steps of post-processing are performed to reduce artifacts and false connections. Compared to Sat2Graph, our method learns connectivity end-to-end, without complicated post-processing.

3. GraphMapper

Our main idea is to turn various vector mapping problems into a unified primitive graph estimation problem. In the following sections, We will first introduce primitive graph in Section 3.1. Then we explain the design of GraphMapper's network architecture in Section 3.2, and training targets in Section 3.3. Lastly, we provide details of applying GraphMapper to road and building mapping in Section 3.4.



(a) GraphMapper workflow. We use two PLS modules sequentially for primitive refinement and relationship reconstruction. Shape regularization and topology reconstruction are achieved in this process.



(b) The stages of GraphMapper when applied to building and road mapping.

Figure 1: GraphMapper workflow and application to building and road mapping.

3.1. Primitive Graph

A primitive graph is a homogeneous un-directed graph:

$$G = \{V, E\}, V \in \mathbb{R}^{N \times d}, E \in \mathbb{Z}^{+N \times N} \quad (1)$$

where V represents N primitives with d-dimensional coordinates. A point primitive is represented by its image coordinate (x, y); a line segment primitive is represented by the coordinates of its two endpoints (x_1, y_1, x_2, y_2). E is the relationship matrix, in which E_{ij} represents the relationship between V_i and V_j. Example primitive graph representation for roads and buildings are shown in Fig. 2. Depending on the choice of primitives and pairwise relationships, a primitive graph can model various types of targets.

3.2. Network Architecture

The overall structure of GraphMapper is illustrated in Fig. 1. We reconstruct primitive graphs in three sequential steps. We first extract initial primitives from input images. Then, the locations of initial primitives are refined using a primitive learning structure (PLS). The direction or normal direction of each primitive is also estimated at this stage. Lastly, we reconstruct the relationship matrix between pairwise primitives using another PLS.

3.2.1 Primitive Detection

We detect primitives by sampling from semantic maps predicted from input images. Previous methods tried to sample points from key point heatmap [18, 39]. We found this

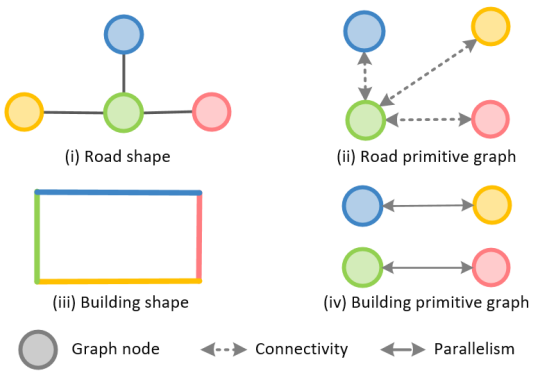


Figure 2: Example primitive graph representations of a road junction and a building polygon.

method often miss-detect points. Therefore, we sample additional key points from semantic segmentation maps.

More specifically, we use an FPN [24] network with a resnet backbone to encode input image $I \in \mathbb{R}_+^{3\times H\times W}$ into a multi-scale feature F. Then, we predict a semantic segmentation map $Y_{seg} \in \mathbb{R}_+^{S\times H\times W}$ and a key point heatmap $Y_{kp} \in \mathbb{R}_+^{K\times H\times W}$ from F using two FCN heads.

To sample point primitives, we first extract local maximum points using NMS on all non-background classes of Y_{seg} and Y_{kp}, resulting $S+K-2$ of candidate points. Then, we combine all candidate points and apply another NMS with category-specific priority scores to remove redundant points to get the sampled points $V' \in \mathbb{R}_+^{N\times 2}$, so that points of higher priority categories are kept over lower ones.

We sample line segments by connecting sampled key points. For polygon structures (i.e., buildings, forests), we trace contours from segmentation maps, which are then simplified using the Douglas–Peucker (DP) algorithm [13]. Points of the simplified contours are combined with sampled key points. We project points to their nearest contours and connect points according to their projections' sequence in contour [10]. Note that when the targets are represented by polygons, this method can accurately derive the relative sequence between line segments without learning the connectivity between line segments.

3.2.2 Primitive Graph Reconstruction

Given the initialized primitives as input, we first refine the coordinates of primitives to improve location accuracy and alleviate the difficulty of relationship learning. Then, we predict the pairwise relationship of the refined primitives. As both tasks benefit from shape context information, they share a common primitive learning structure. Different from previous methods [39], we reconstruct pairwise relationships based on refined primitives instead of initial primitives. Using refined primitives can reduce the ambiguity for relationship recognition and ground truth relationship generation, which is essential for high-quality primitive relationship reconstruction.

Primitive Learning Structure. The structure of Primitive Learning Structure (PLS) is illustrated in Fig. 1. Given primitives V and their visual features as input, visual features at primitives' locations are pooled using patch pooling. Patch pooling extracts a small crop of image features centered at each primitive, and compresses the cropped features using a small FCN network. The derived primitive features h_{vis} are flattened and individually projected using an MLP (f_{proj}), the result of which is fed into a multi-layer MHSA module f_{MHSA} [33] together with the positional encoding of primitive coordinates [8]. MHSA can fuse geometric and visual information and exchange information among primitives to generate local and global contextualized primitive features h_{prims}, which are used by MLP heads to generate output predictions.

Primitive Refinement. We use a PLS with two MLP heads to predict the coordinate deformation and directions $D \in \mathbb{R}^{N\times 2}$ (normal direction for points and line direction for line segments) of input primitives from PLS generated primitive feature h_{prims}. We get refined primitives V' by adding estimated deformation back to input coordinates. Note that line segments' direction can also be computed from their coordinates V'. However, we found network predicted D is more accurate, as it is easier to regress a direction than adjusting both vertices of a line segment to achieve the desired angle while staying close to the boundary of segmentation masks.

For direction regression, the discontinuity of rotation angles at 0 and 180° can lead to unstable learning [37] when naively applying *L2 loss* on direction angles D. Therefore, we regress the sine and cosine of a surrogate angle which is 2 times the target angle A as suggested in [37]:

$$\overline{D}_i = (\cos(2A_i), \sin(2A_i)). \tag{2}$$

For each image, the network predicts a two-dimensional tensor of size N × 2, normalizes it along the dimension to get $\overline{D}$. During inference, we recover the actual direction D' from the surrogate direction D as $(\arccos \overline{D}_i/||\overline{D}_i||_2)/2$.

Primitive Relationship Reconstruction. We input the refined primitives V' and visual feature into another PLS to predict relationship matrix E. For a pair of point primitives, we extract additional visual features on the line segment between them using LOI [38] from the visual feature, Y_{seg} and Y_{kp}. These extracted features are concatenated together with their point features in h_{prims} to form the point pair's feature. For a pair of line segment primitives, we simply concatenate their features in h_{prim} as the pair's feature. For N primitives, we get a relationship feature matrix $Q \in \mathbb{R}^{d_r\times N\times N}$. The MLP heads in PLS independently classify each pair's relationship using their feature in Q.

As only spatially neighboring primitives have a positive

relationship, the ratio of positive and negative relationships in relationship matrix E is often strongly biased. Therefore, only primitive pairs within a spatial distance threshold t are used for training to balance the positive-to-negative sample ratio in E.

Shape regularization. Primitive graph allows explicit representation of shape regularization as the consistency between primitives V and their relationship matrix E:

$$L_{reg} = \sum_{i,j,r} E_{ijr} |f_{prop}(V_i, V_j, r) - f_{prop}(\overline{V}_i, \overline{V}_j, r)| \quad (3)$$

, where f_{prop} computes the desired property between primitives. The term in $|\cdot|$ represents the difference between computed and desired property between V_i and V_j for relationship category r. $\overline{V}_i$ and $\overline{V}_j$ are the corresponding ground truth primitives for V_i and V_j. This formulation explicitly enforces low variation of primitives' relative properties. Additionally, by adjusting the strength of regularization according to the probability of its relationship type, the network can learn to balance between shape regularity and location accuracy.

3.3. GraphMapper Learning

GraphMapper is trained with a linear combination of the following losses:

$$(\mathcal{L}_{seg}, \mathcal{L}_{kp}, \mathcal{L}_{off}, \mathcal{L}_{dir}, \mathcal{L}_{ret}, \mathcal{L}_{reg}, \mathcal{L}_{aux}) \quad (4)$$

, which we explain below.

Shape regularization loss $\mathcal{L}_{reg}$. We enable shape regularization by training with L_{reg}. We set f_{prop} in Eq. 3 to

$$f_{prop}(D_i, D_j, r) = \begin{cases} \cos(2[D_i - D_j + 2\pi)//\pi]) & , r = 1 \\ 0 & , r = 0 \end{cases} \quad (5)$$

, which computes the cosine between two line directions. Here $r = 1$ means two primitives have a relationship with fixed relative angles, such as parallel or perpendicular. $r = 0$ means no relationship, therefore no regularization. Practically, we compute an angle matrix for all line segment pairs as shown in Fig. 1. L_{reg} is evaluated using the angle matrix and relationship matrix.

Relationship loss $\mathcal{L}_{rel}$. $\mathcal{L}_{rel}$ is a cross-entropy loss applied to elements in E:

$$\mathcal{L}_{rel} = -\frac{1}{|U|} \sum_{(i,j) \in U} E_{i,j} log(\overline{E}_{i,j}), \quad (6)$$

where $\overline{E}_{i,j}$ is the ground truth relationship matrix between primitives V_i', V_j') represented in one-hot format. U is the set of primitive pairs that have a distance less than t.

Primitive direction loss $\mathcal{L}_{dir}$. $\mathcal{L}_{dir}$ is a loss applied to normalized surrogate direction D:

$$\mathcal{L}_{dir} = \frac{1}{|D|} \sum (D_i - \overline{D}_i)^2 \quad (7)$$

where $\overline{D_i}$ is the unit direction vector of 2 times the ground truth angle for primitive V_i', which is the direction of its matched primitive in ground truth.

Auxiliary loss $\mathcal{L}_{aux}$. To facilitate the learning of geometric shape features in MHSA, we would like primitives' visual feature h_{vis} to be more related to geometric properties. Therefore, we add an auxiliary predictor on features pooled by patch pooling in primitive refinement PLS to predict coordinate offsets and primitive directions. Hence, $\mathcal{L}_{aux}$ is a linear combination of a deformation loss and a direction loss applied to the auxiliary predictions.

Additionally, We use linear combination of cross-entropy loss and Lovász-softmax loss [6] for $\mathcal{L}_{seg}$ and $\mathcal{L}_{kp}$ on segmentation maps. We use bi-projection loss [10] for shape deformation ($\mathcal{L}_{off}$). The point matches computed by bi-projection loss are re-used to find the ground truth relationship matrix between primitives and primitive directions. Please see Supplementary for more details.

3.4. Implementation Details

We only provide essential details here due to the limitation of space. Please refer to the Supplementary for more implementation details.

Training and testing. To provide reasonable shapes for primitive graph reconstruction modules, we pre-train primitive detection before training all modules end-to-end. We use Adam optimizer [20] with batch size 12 and initial learning rate 2e-4. The max number of primitives per image is set to 150 for training and 300 for inference. Extra sampled primitives are discarded.

Building mapping. We predict line segment primitives and pairwise inlineness for building mapping. We trace contours from Y_{seg}, and reconstruct a primitive graph for each group of line segments that belongs to the same contour, so that network can learn the context information within a building instance without the interference of other building geometries. At inference time, inline line segments are merged to simplify output polygons. Please see more details in Supplementary.

Road network mapping. We predict point primitives and pairwise connectivity for road mapping. We segment the buffered region of 5 pixels wide around road centerlines. Following [18], we predict four classes of key points in Y_{kp}: junctions, overlays (crossing points of overlapping roads), endpoints (endpoints of road line segments that are not junctions or overlays), and interpolated points (points of fixed intervals interpolated on road line segments). Here, we don't use $\mathcal{L}_{reg}$ as primitive refinement can already achieve

Figure 3: Qualitative result of road mapping on City-Scale dataset. Top row is from Sat2Graph, bottom row is our method. Our method shows to generate improved road topology compared to Sat2Graph in various scenarios.

the required shape regularization on road. To reconstruct road networks, we connect point V_i' to a maximum of t_i points that have connectivity probability in E larger than threshold t_r. We set $t_i = 3$ for junctions, and 2 for other points. We apply *L2 normalization* to the primitive features before the last MLP layer of the second PLS to improve feature embedding quality, as inspired by contrastive learning studies [17, 3, 11].

4. Experiments

4.1. Datasets And Metrics

Building. CrowdAI Mapping Challenge Dataset [28] (CrowdAI dataset): It contains 280741 annotated aerial images for training and 60317 for testing. Each image has a size of 300 × 300 pixels.

We use IoU (Intersection Over Union) and AP/AR (Average Precision/Average Recall) to evaluate the overall correctness of generated polygons. Since IoU and AP/AR cannot describe the cleanness of predictions at boundaries, we adopt Mean Max Tangent Angle Error (MTE) [16] and C-IoU [39] as additional evaluation metrics to IoU and AP/AR. MTE computes the max angle error of all line segments for each building and reports the average value over the entire dataset [16]. C-IoU is IoU weighted by polygon simplicity, where more points in the predicted polygon means lower polygon simplicity and lower C-IoU. Here, polygon simplicity is evaluated using *N ratio*, which is the ratio of the number of vertices between predictions and ground truth [39].

Road Network. (1) SpaceNet road dataset [15]: it contains 2549 satellite images of size 1300 × 1300 pixels with resolution around 0.3m. This dataset is challenging due to the diverse scenarios from 5 cities around the globe. (2) City-Scale Dataset [18]: it contains 180 tiles of size 2000 × 2000 with 1-meter spatial resolution.

Road network topology is evaluated using TOPO [7] and Average Path Length Similarity (APLS) [15]. TOPO measures the similarity of sub-graphs near seed points sampled from the inferred graph and ground truth graph. APLS measures the similarity of graphs using the shortest path between sampled point pairs on each graph, which is more sensitive to topology structure compared to TOPO.

4.2. Benchmark results

Building footprint extraction. Qualitative evaluation results are reported in Fig. 4. We compare GraphMapper with two recent methods, Frame Field Learning (FFL) [16] and PolyWorld [39], that represent the state-of-the-art in building polygon mapping. All methods can accurately capture small and simple building polygons. GraphMapper shows the best shape regularization, polygon simplicity, and accuracy.

Quantitative evaluation results are reported in Tab. 1. GraphMapper significantly outperforms Polyworld [39] by 9.5/7.7 in COCO AP/AR. The improvements are largely contributed by primitive refinement and relationship reconstruction, which improved AP/AR by 9.2/7.2 to segmentation mask (*GraphMapper, mask*). We report IoU, C-IoU and MTA evaluation results in Tab. 2. GraphMapper outperforms all competing methods in all metrics. The improved C-IoU, and MTA numerically demonstrated that

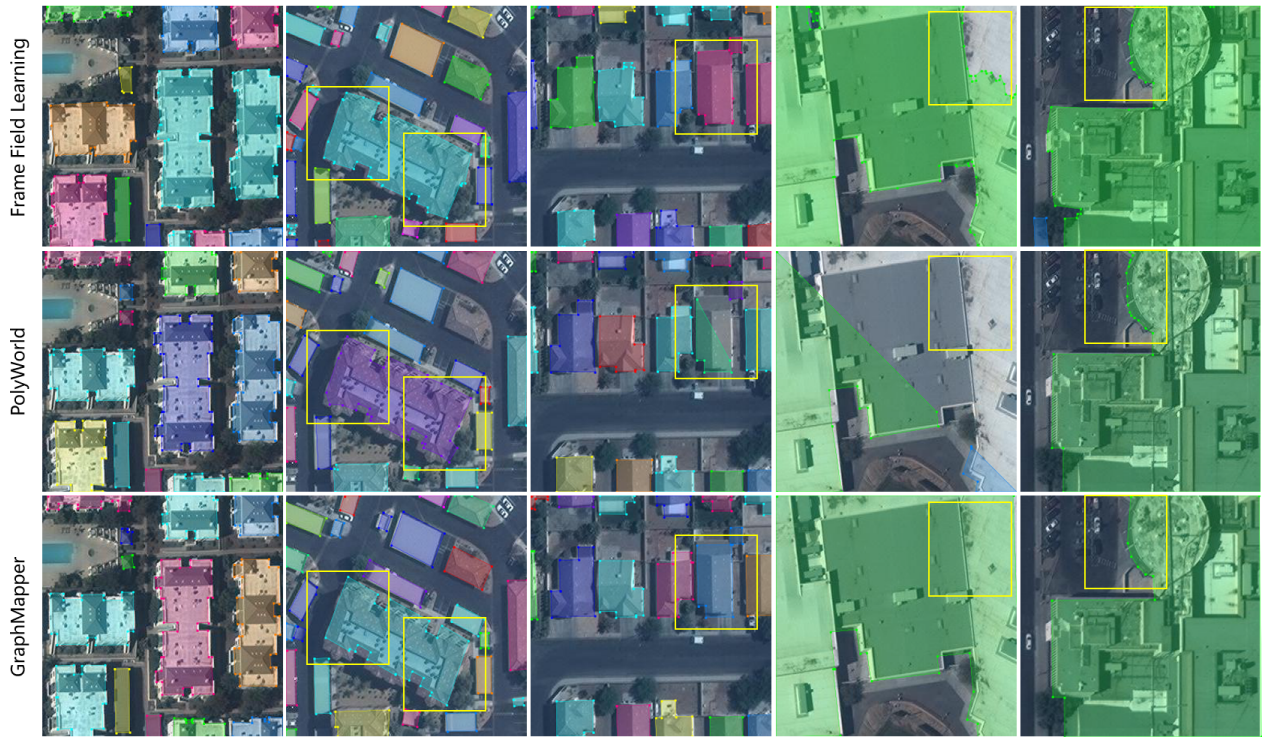


Figure 4: Qualitative evaluation of building footprint mapping results. From top to bottom are the results from Frame Field Learning [16], Polyworld [39], and GraphMapper. Yellow boxes highlight the differences between different methods.

Method	AP	AP_{50}	AP_{75}	AP_S	AP_M	AP_L	AR	AR_{50}	AR_{75}	AR_S	AR_M	AR_L
FFL (with field), mask	57.7	83.8	66.3	33.8	73.8	81.0	68.1	91.0	77.7	47.5	80.0	86.7
FFL (with field), simple poly	61.7	87.6	71.4	35.7	74.9	83.0	65.4	89.8	74.6	42.5	78.6	85.8
FFL (with field), ACM poly	61.3	87.4	70.6	33.9	75.1	83.1	64.9	89.4	73.9	41.2	78.7	85.9
PolyWorld (offset off)	58.7	86.9	64.5	31.8	80.1	85.9	71.7	92.6	79.9	47.4	85.7	94.0
PolyWorld (offset on)	63.3	88.6	70.5	37.2	83.6	87.7	75.4	**93.5**	83.1	52.5	88.7	95.2
GraphMapper, mask	63.6	88.3	69.6	35.6	85.8	93.9	75.9	93.1	82.8	50.7	90.2	98.1
GraphMapper, refine	72.7	89.3	79.7	46.8	91.2	95.2	83.1	93.7	88.1	61.6	95.4	98.6
GraphMapper, final	**72.8**	**89.1**	**79.7**	**46.6**	**90.6**	**91.3**	**83.1**	93.3	**88.1**	**61.5**	**95.2**	**97.2**

Table 1: COCO evaluation results for building on CrowdAI Dataset. mask: the segmentation mask; refine: shapes with deformed point location and line direction from segmentation masks; final: shapes generated with pairwise line relationship predictions from refine.

Method	IoU	C-IoU	MTA	N ratio
FFL (no field), simple poly	83.9	23.6	51.8°	5.96
FFL (with field), simple poly	84.0	30.1	48.2°	2.31
FFL (with field), ACM poly	84.1	73.7	33.5°	1.13
PolyWorld (offset off)	89.9	86.9	35.0°	0.93
PolyWorld (offset on)	91.3	88.2	32.9°	0.93
GraphMapper	**93.9**	**88.8**	**30.4°**	1.01

Table 2: IoU, MTA, C-IoU, and N ratio evaluation results on CrowdAI Dataset.

GraphMapper can generate more regularized vector shapes compared to previous methods, which is consistent with our visual comparison. Our N ratio is also closer to 1 compared to other methods, which suggests our method generates polygons with more similar complexity to ground truth.

Road network extraction. We compare GraphMapper with Sat2Graph [18] in Fig. 3, which is considered the state-of-the-art method in road network mapping. GraphMapper can generate visually comparable results to Sat2Graph, but with fewer redundant roads in both datasets. Also, GraphMapper doesn't require tedious post-processing.

Quantitatively, GraphMapper consistently shows im-

Method	City-Scale Dataset				SpaceNet Roads Dataset			
	Prec. ↑	*Rec.* ↑	F_1 ↑	*APLS* ↑	*Prec.* ↑	*Rec.* ↑	F_1 ↑	*APLS* ↑
DRM[26]	76.5	71.3	73.8	54.3	82.8	72.6	77.3	62.3
SO[5]	75.8	68.9	72.2	55.3	81.6	71.4	76.1	58.8
RoadTracer[4]	78.0	57.4	66.2	57.3	78.6	62.5	69.6	56.0
Sat2Graph[18]	80.7	72.3	76.3	63.1	85.9	76.6	80.9	64.4
GraphMapper	**89.6**	**82.6**	**85.6**	**68.0**	**90.7**	**84.6**	**87.1**	**68.8**

Table 3: Comparison of road TOPO and APLS evaluation metric.

Method	IoU	C-IoU	MTA	N ratio
simple poly	92.3	81.6	38.5°	1.26
GraphMapper	**94.3**	**89.5**	**30.4°**	**1.01**
w/o L_{aux}	94.1	89.0	30.9°	1.08
w/o L_{dir}	93.5	80.8	33.1°	1.18
w/o L_{reg}	**94.4**	82.7	31.5°	1.28
w/o incremental	93.8	78.3	34.6°	1.11

Table 4: Building ablation study on CrowdAI Dataset.

Method	*Pre.*	*Rec.*	F_1	*APLS*
w/o sort	87.2	80.1	82.5	66.2
w/o incremental	89.6	81.2	84.8	67.6
GraphMapper	**89.9**	**82.9**	**85.9**	**68.9**

Table 5: Road Ablation study on City-Scale dataset.

provement for APLS and TOPO on both road datasets (Tab. 3) compared to Sat2Graph [18], where TOPO F1 is improved by $6.2 \sim 9.3$, and APLS is improved by $4.4 \sim 4.9$.

4.3. Ablation Study and Discussion

We report ablation study results in Tab. 4 for buildings and Tab. 5 for roads. In these two tables, *GraphMapper* is our proposed model; *simple poly* uses DP algorithm [13] over traced polygons without primitive refinement or relationship reconstruction; *w/o incremental* refines primitives and predicts relationships in parallel with shared MHSA encoder; *w/o* L_{aux}, *w/o* L_{dir} and *w/o* L_{reg} removes the corresponding loss during training; *w/o sort* classifies relationship by thresholding relationship probability in E.

Shape regularization. Relationship reconstruction for shape regularization shows to improve C-IoU by 6.8 and MTA by 1.1° compared to without shape regularization loss (*w/o reg*) for building mapping, which suggests that relationship learning with consistency between primitives' properties and their relationships can effectively regularize shapes for vector mapping. The network shows to adjust the intensity of regularization in a fashion similar to human labelers.

Even directions can be easily computed from the end points of line segments, regressed directions shows much improved direction accuracy (*w/o* L_{dir}'s MTA drops by 2.7°). Direction regression may be easier to learn compared to point deformation, as the deformation ground truth varies according to segmentation accuracy, while the ground truth of direction is fixed.

Auxiliary loss L_{aux} on primitive's image features shows to slightly improve performance in Table 4. This is consistent with previous studies that uses auxiliary loses in intermediate MHA layers [21, 35].

Incremental Reconstruction. For building mapping, incremental modeling (*GraphMapper*) shows to significantly improve C-IoU by 11.2% and MTA by 4.2% compared to parallel modeling (*w/o incremental*). Similarly for road mapping, TOPO and ALPS are improved with incremental reconstruction. We believe that the improved performance of incremental modeling is caused by the improved accuracy of ground truth matching for relationship learning when using refined primitives instead of initial primitives.

Sorting in embedding space. We compare our point connections strategy with naive connectivity relationship classification (*w/o sort*) in road mapping. Our method is shown to improve TOPO F1 by 3.4 and ALPS by 2.7 compared to standard relationship classification (*w/o sort*). We found the improvement is mainly due to reduced redundant connections between points, and reduced sensitivity to the threshold (See supplementary materials) of connectivity probability.

5. Conclusions

We propose GraphMapper, an end-to-end model for unified vector mapping from satellite images. By converting vector mapping tasks into primitive graph estimation tasks, GraphMapper can explicitly model shape regularization and topology reconstruction within the same framework. We applied GraphMapper to building and road mapping with few task-specific designs and achieved favorable performance to existing methods. The simplicity and strong performance of GraphMapper effectively reduced the complexity of comprehensive mapping tasks.

For future study, we plan to improve the compatibility between shape features and image features for better co-learning of vector shapes and image features.

References

[1] Microsoft US building footprints. https://github.com/microsoft/USBuildingFootprints. Accessed: 2018-07-14. 1, 2

[2] David Acuna, Huan Ling, Amlan Kar, and Sanja Fidler. Efficient interactive annotation of segmentation datasets with polygon-rnn++. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 2

[3] Adrien Bardes, Jean Ponce, and Yann LeCun. Vicreg: Variance-invariance-covariance regularization for self-supervised learning, 2021. 6

[4] Favyen Bastani, Songtao He, Sofiane Abbar, Mohammad Alizadeh, Hari Balakrishnan, Sanjay Chawla, Sam Madden, and David DeWitt. Roadtracer: Automatic extraction of road networks from aerial images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 2, 8

[5] Anil Batra, Suriya Singh, Guan Pang, Saikat Basu, C.V. Jawahar, and Manohar Paluri. Improved road connectivity by joint learning of orientation and segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 2, 8

[6] Maxim Berman, Amal Rannen Triki, and Matthew B Blaschko. The lovász-softmax loss: A tractable surrogate for the optimization of the intersection-over-union measure in neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4413–4421, 2018. 5

[7] James Biagioni and Jakob Eriksson. Inferring road maps from global positioning system traces: Survey and comparative evaluation. *Transportation Research Record*, 2291(1):61–71, 2012. 6

[8] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, pages 213–229, Cham, 2020. Springer International Publishing. 4

[9] Lluis Castrejon, Kaustav Kundu, Raquel Urtasun, and Sanja Fidler. Annotating object instances with a polygon-rnn. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 2

[10] Qi Chen, Lei Wang, Steven L. Waslander, and Xiuguo Liu. An end-to-end shape modeling framework for vectorized building outline generation from aerial images. *ISPRS Journal of Photogrammetry and Remote Sensing*, 170:114–126, 2020. 1, 2, 4, 5

[11] Xinlei Chen and Kaiming He. Exploring simple siamese representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 15750–15758, June 2021. 6

[12] Dominic Cheng, Renjie Liao, Sanja Fidler, and Raquel Urtasun. Darnet: Deep active ray network for building segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 1, 2

[13] David H Douglas and Thomas 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*, 10(2):112–122, 1973. 4, 8

[14] Adam Van Etten. City-scale road extraction from satellite imagery v2: Road speeds and travel times. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, March 2020. 2

[15] Adam Van Etten, Dave Lindenbaum, and Todd M. Bacastow. Spacenet: A remote sensing dataset and challenge series, 2019. 6

[16] 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 (CVPR)*, pages 5891–5900, June 2021. 1, 2, 6, 7

[17] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, Bilal Piot, koray kavukcuoglu, Remi Munos, and Michal Valko. Bootstrap your own latent - a new approach to self-supervised learning. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 21271–21284. Curran Associates, Inc., 2020. 6

[18] Songtao He, Favyen Bastani, Satvat Jagwani, Mohammad Alizadeh, Hari Balakrishnan, Sanjay Chawla, Mohamed M. Elshrif, Samuel Madden, and Mohammad Amin Sadeghi. Sat2graph: Road graph extraction through graph-tensor encoding. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, pages 51–67, Cham, 2020. Springer International Publishing. 1, 2, 3, 5, 6, 7, 8

[19] Michael Kass, Andrew Witkin, and Demetri Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 1:321–331, 1988. 2

[20] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2014. 5

[21] Feng Li, Hao Zhang, Shilong Liu, Jian Guo, Lionel M Ni, and Lei Zhang. Dn-detr: Accelerate detr training by introducing query denoising. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13619–13627, 2022. 8

[22] Zuoyue Li, Jan Dirk Wegner, and Aurelien Lucchi. Topological map extraction from overhead images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019. 2

[23] Justin Liang, Namdar Homayounfar, Wei-Chiu Ma, Yuwen Xiong, Rui Hu, and Raquel Urtasun. Polytransform: Deep polygon transformer for instance segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 1, 2

[24] Tsung-Yi Lin, Piotr Dollar, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017. 4

[25] Diego Marcos, Devis Tuia, Benjamin Kellenberger, Lisa Zhang, Min Bai, Renjie Liao, and Raquel Urtasun. Learning deep structured active contours end-to-end. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 1, 2

[26] Gellert Mattyus, Wenjie Luo, and Raquel Urtasun. Deeproadmapper: Extracting road topology from aerial images. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. 2, 8

[27] Volodymyr Mnih and Geoffrey E. Hinton. Learning to detect roads in high-resolution aerial images. In Kostas Daniilidis, Petros Maragos, and Nikos Paragios, editors, *Computer Vision – ECCV 2010*, pages 210–223, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg. 2

[28] 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

[29] Agata Mosinska, Pablo Márquez-Neila, Mateusz Koziński, and Pascal Fua. Beyond the pixel-wise loss for topology-aware delineation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 2

[30] Nelson Nauata and Yasutaka Furukawa. Vectorizing world buildings: Planar graph reconstruction by primitive detection and relationship inference. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, pages 711–726, Cham, 2020. Springer International Publishing. 1

[31] Wojciech Sirko, Sergii Kashubin, Marvin Ritter, Abigail Annkah, Yasser Salah Eddine Bouchareb, Yann Dauphin, Daniel Keysers, Maxim Neumann, Moustapha Cisse, and John Quinn. Continental-scale building detection from high resolution satellite imagery, 2021. 1, 2

[32] Yong-Qiang Tan, Shang-Hua Gao, Xuan-Yi Li, Ming-Ming Cheng, and Bo Ren. Vecroad: Point-based iterative graph exploration for road graphs extraction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 2

[33] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. 4

[34] Fuyang Zhang, Nelson Nauata, and Yasutaka Furukawa. Conv-mpn: Convolutional message passing neural network for structured outdoor architecture reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 1

[35] Hao Zhang, Feng Li, Shilong Liu, Lei Zhang, Hang Su, Jun Zhu, Lionel Ni, and Heung-Yeung Shum. Dino: Detr with improved denoising anchor boxes for end-to-end object detection. In *The Eleventh International Conference on Learning Representations*, 2022. 8

[36] Lichen Zhou, Chuang Zhang, and Ming Wu. D-linknet: Linknet with pretrained encoder and dilated convolution for high resolution satellite imagery road extraction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2018. 2

[37] Yi Zhou, Connelly Barnes, Jingwan Lu, Jimei Yang, and Hao Li. On the continuity of rotation representations in neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 4

[38] Yichao Zhou, Haozhi Qi, and Yi Ma. End-to-end wireframe parsing. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019. 4

[39] 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 (CVPR)*, pages 1848–1857, June 2022. 1, 2, 3, 4, 6, 7

[40] Stefano Zorzi, Ksenia Bittner, and Friedrich Fraundorfer. Machine-learned regularization and polygonization of building segmentation masks. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 3098–3105, 2021. 2