Supplementary document for *csBoundary*

February 16, 2022

Zhenhua Xu, Yuxuan Liu, Lu Gan, Xiangcheng Hu,
Yuxiang Sun, Lujia Wang, Ming Liu

1 Introduction

In this supplementary document, we provide more details about our proposed methods csBoundary.

2 csBoundary

csBoundary is proposed for city-scale road boundary detection. Due to the limitation of computation resources, we first predict graphs within each image patch and then stitch the obtained graphs of patches into the final city-scale graph.

The system pseudocode is shown in Alg. 1.

Algorithm 1: The proposed csBoundary

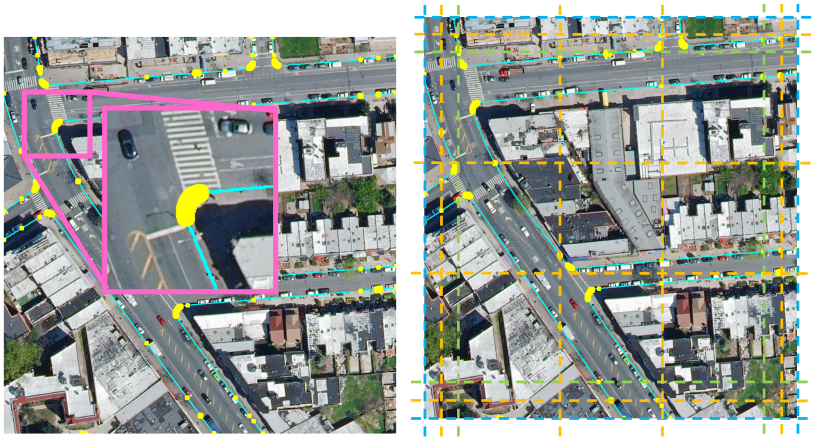
Input: A set of aerial image patches $A = \{I_i\}_{i=1}^{N}$
Output: A city-scale undirected & unweighted graph G representing road boundaries

begin
  $K_p \leftarrow \emptyset$, $G_p \leftarrow \emptyset$
  $A \leftarrow expand\_image\_patch(A)$ # Image patch expansion, refer to 2.2.1. Section III-B in our paper.
  while A *not empty* **do**
    $I \leftarrow A.pop()$
    $K \leftarrow FPN(I)$ # Segmentation map prediction by FPN network. Section III-C in our paper.
    $K_p.push(K)$
  end
  $K_p \leftarrow keypoint\_segmentation\_map\_stitching(K_p)$ # Keypoint segmentation map stitching, refer to 2.2.2. Section III-F in our paper.
  while K_p *not empty* **do**
    $K \leftarrow K_p.pop()$
    $V \leftarrow extract\_graph\_vertex(K)$ # Graph vertex extraction, refer to 2.1.3. Section III-D in our paper.
    $E \leftarrow AfANet(V|K, I)$# Adjacency matrix prediction, refer to 2.1.4. Section III-E in our paper.
    $G_p.push(G_i = (V, E))$
  end
  $G \leftarrow graph\_stitching(G_p)$# Patch-level Graph stitching, refer to 2.2.3. Section III-F in our paper.
  return G
end

2.1 Graph prediction within an image patch

2.1.1 Keypoint definition

Not like human pose estimation or road network detection that have keypoints with unique semantic meanings, it is hard to clearly define keypoints in road boundaries. In our paper, two types of keypoints are defined: (1) intersection keypoints, which are the intersection points of road boundaries and manually defined lines (e.g., image edges); (2) corner keypoints, which maybe gathered points at rounded-corners or isolated points at sharp corners. All keypoints are uniquely defined. Example visualization is shown in Fig. 1.



(a) Corner keypoints (b) Intersection keypoints

Figure 1: Visualization of keypoints (yellow points in subfigures). (a) Gathered corner keypoints. (b) Intersection keypoints. Blue dash lines are edges of the current image patch, green dash lines are edges of adjacent image patches and orange dash lines are extra lines to create more keypoints. Extra lines are used to prevent too sparse keypoints. Intersection keypoints are the intersection points of road boundaries and aforementioned dash lines.

2.1.2 Keypoint segmentation map prediction

In our work, keypoint segmentation map is predicted by FPN (feature pyramid network) [1]. More powerful backbone network could be utilized in the future to further enahnce the performance.

2.1.3 Graph vertex extraction

After semantic segmentation, we extract graph vertices from the predicted keypoint map. The pipeline is shown in Fig. 2.

2.1.4 Adjacency matrix prediction by AfANet

After obtaining graph vertices, we need to predict edges to create a graph. In our task, the expected graph is **undirected and unweighted**, thus we only need to predict the connection relationship between vertices. The connection relationship between two vertices could be seen as a binary classification problem. Therefore, predicting one adjacency matrix is sufficient for our task to annotate the road-boundary graph.

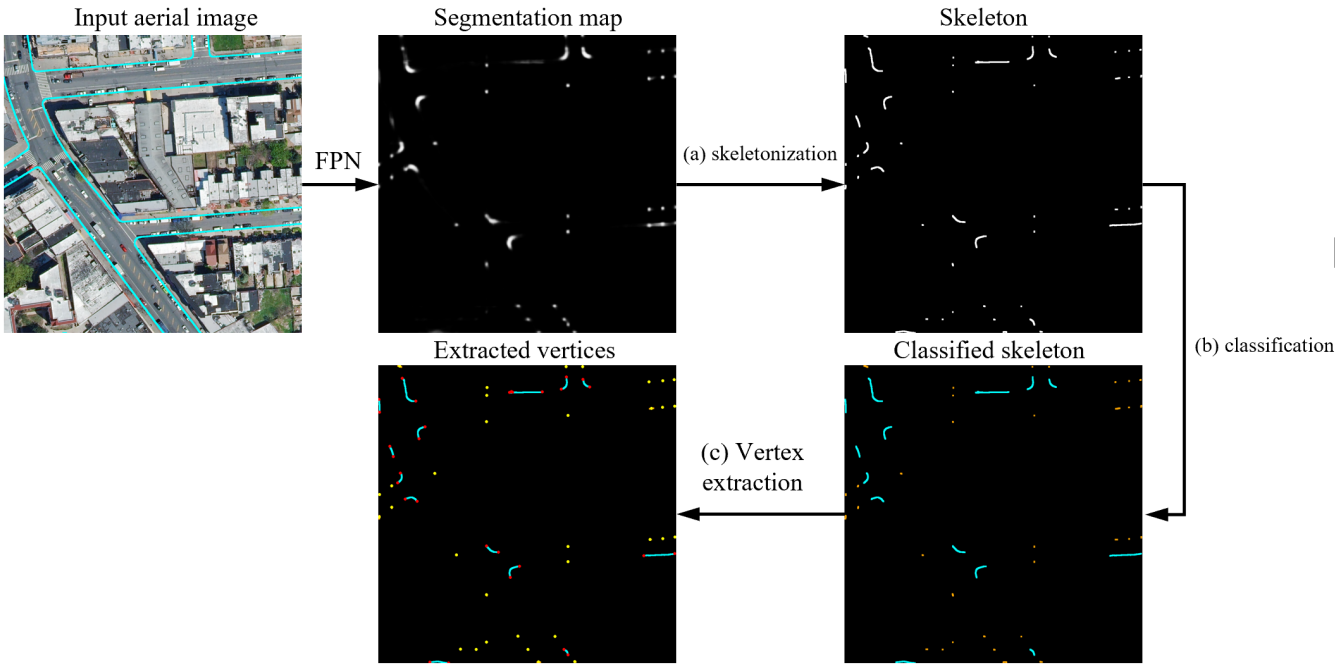


Figure 2: Pipeline of graph vertex extraction. (a) Skeletonization. We conduct a series of processing such as binarization, filtering and skeletonization to obtain the skeleton of the keypoint segmentation map. (b) Based on the area of each skeleton instance, we classify large skeletons as gathered keypoints (denoted by cyan lines) and small skeletons as isolated keypoints (denoted by orange lines). (c) We extracted graph vertices. For large skeletons, we only extract endpoints of them as vertices (denoted by red points), while for small skeletons, we treat their center points as graph vertices (denoted yellow points).

The proposed AfANet mainly relies on self-attention mechanism in transformer. It can effectively and efficiently predict the adjacency matrix in one-shot manner.

2.2 Graph stitching for city-scale graph

2.2.1 Image expansion

Image patch is expanded first to create overlapping areas, which is critical for afterwards graph stitching operations. The schematic diagram of image patch expansion is shown in Fig. 3, and a real-world example is visualized in Fig. 4.

2.2.2 Keypoint segmentation map stitching and normalization

Since the segmentation map of each image patch is calculated separately, the overlapping areas predicted by different patches may be different, thus normalization is needed. For instance, for a overlapping area shared by two image patches, the normalized result is the mean of segmentation maps predicted by two patches. An example is visualized in Fig. 5. After normalization, the overlapping area of adjacent image patches will be exactly the same, which will benefit the following graph stitching operation.

2.2.3 Graph stitching

After keypoint segmentation map stitching and normalization, the overlapping area of adjacent image patches will be the same, so that the vertices extracted from the overlapping area will be almost the same.

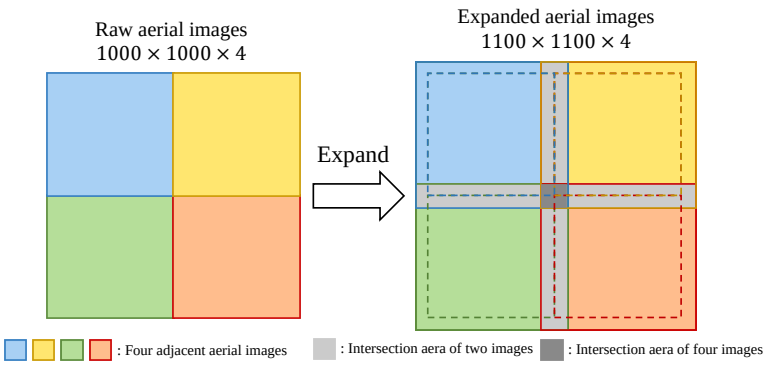


Figure 3: Schematic diagram of image patch expansion. In this paper, all aerial image patches are expanded into 1100 × 1100-size, which causes intersection areas between adjacent patches (light gray and dark gray areas). Dashed rectangles on the right represent the original image edges. These intersection areas will benefit the graph stitching process.

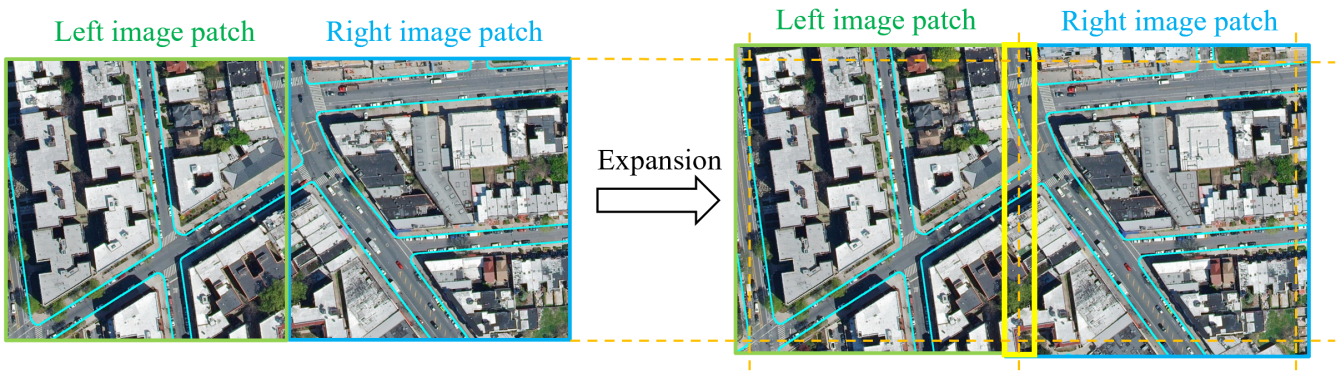


Figure 4: Example of image patch expansion. For graph stitching purpose, following BASISS (Broad Area Satellite Imagery Semantic Segmentation) methodology proposed in [2], we expand each aerial image into 1100 × 1100-sized to create overlapping areas (yellow rectangle, 1100 × 100-sized) between adjacent image patches. Orange dash lines mark the edge of image patches before expansion.

The vertices extracted from the overlapping area are shared by adjacent image patches, so that the graphs of adjacent image patches could be easily merged together by these shared vertices. The vertices are matched by minimizing Euclidean distance. The schematic diagram of graph stitching is shown in Fig. 6, and a real-world example is visualized in Fig. 7.

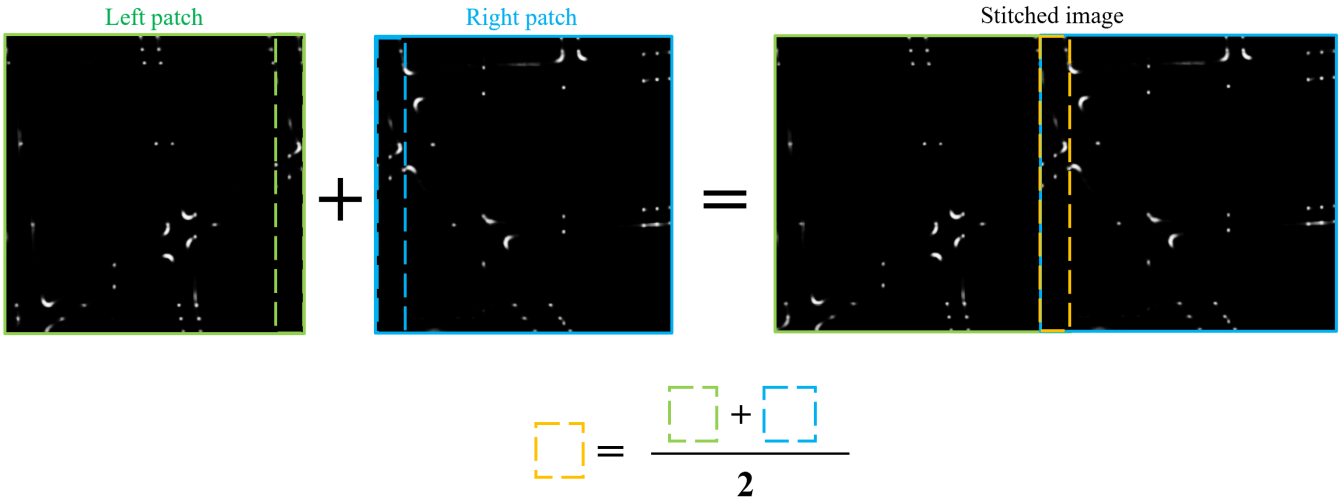


Figure 5: Visualization of keypoint segmentation map stitching and normalization. Due to the expansion of images, we create overlapping areas between adjacent image patches, thus the keypoint segmentation map of adjacent image patches also have overlapping areas (green and blue dashed rectangles). The overlapping area in the final stitched image is the mean of overlapping areas in different patches (i.e., in this example, the orange rectangle is the mean of the green one and the blue one).

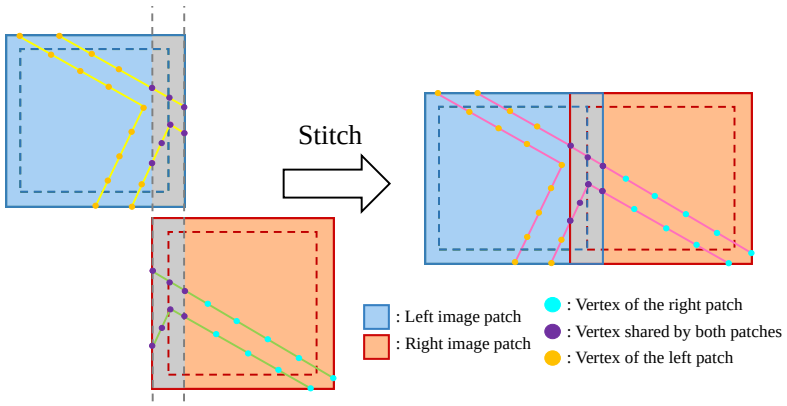


Figure 6: Schematic diagram of graph stitching. The blue image patch is adjacent to the red one (blue patch on the left and red patch on the right). For better visualization, they are placed vertically. The gray areas are the intersection areas. The graph of the blue patch (yellow edge and orange vertex) and the red patch (green edge and cyan vertex) are predicted separately. Purple vertices are shared by both patches. These two patches could be stitched together easily by connecting exclusive vertices of two patches with the shared vertices. This figure is best viewed in color.

3 Topology evaluation metrics

In this work, the topology correctness of predicted road-boundary graph is evaluated by two topology evaluation metrics: APLS [3] and TLTS [4].

APLS is based on finding the shortest paths of randomly sampled vertex pairs, which shares a very similar idea with TLTS. First, sample N vertex pairs (g_1, g_2) in the ground-truth road boundaries, and make sure that vertex g_1 and g_2 belong to the same road-boundary instance (g_1 can reach g_2). Find the length of the shortest path between g_1 and g_2 as $l(g_1, g_2)$. Then, find the corresponding vertex pairs (p_1, p_2) in

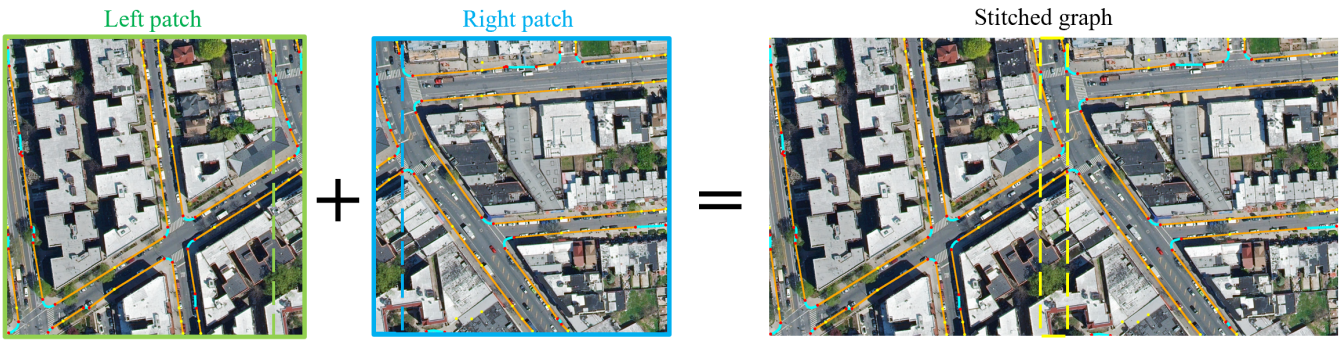


Figure 7: Example of graph stitching. The graph vertices within the overlapping area (green and blue rectangles) are shared by both image patches, so that the graphs of two patches could be merged together by these shared vertices.

the predicted road-boundaries (by minimum Euclidean distance). Find the length of the shortest path between p_1 and p_2 as $l(p_1.p_2)$. Finally, calculated APLS by

$$APLS = 1 - min(1, \frac{|l(g_1, g_2) - l(p_1, p_2)|}{l(g_1, g_2)}) \tag{1}$$

If there is no path between p_1 and p_2 or either p_1 or p_2 is too far from the ground-truth road boundary, APLS of this pair is 0 (largest punishment). Larger APLS indicates better topology correctness. It is widely used to evaluate the topology correctness in many past works, especially in road-network detection tasks. An example of APLS calculation is visualized in Fig. 8.

TLTS is a little bit different from APLS. After calculating shortest distance $l(g_1, g_2)$ and $l(p_1, p_2)$ for one vertex pair, if $|l(g_1, g_2) - l(p_1, p_2)| < l(g_1, g_2) \cdot \phi$, then we say this vertex pair is **not** *too long or too short*. The final TLTS score is the ratio of vertex pairs that are **not** *too long or too short*. ϕ reflects the level of error tolerance.

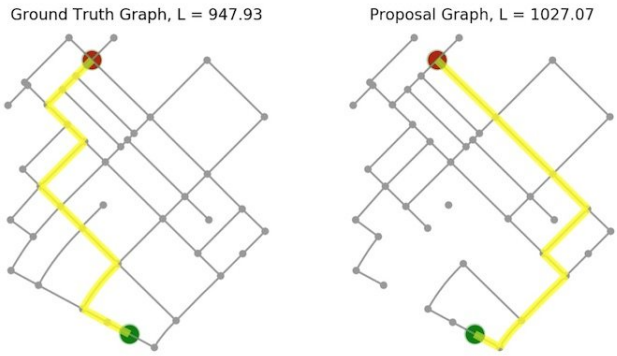


Figure 8: Visualization of the calculation of APLS metric [3]. The ground-truth graph is at the left side and the predicted one in at the right side. First randomly sampling two vertices $\{g_1, g_2\}$ in the ground-truth graph and calculate the shortest path distance between them as $l(g_1, g_2)$; then find the corresponding vertices in the predicted graph $\{p_1, p_2\}$, and calculate the shortest path distance as $l(p_1, p_2)$. The difference between $l(g_1, g_2)$ and $l(p_1, p_2)$ could illustrate the topology correctness of the predicted graph. APLS score of this vertex pair is $APLS = 1 - min(1, \frac{|l(g_1,g_2)-l(p_1,p_2)|}{l(g_1,g_2)})$. After sampling multiple vertices pairs, we have the final APLS which is the mean sum of the APLS score of each vertex pair. In this example, the APLS score of this vertex pairs is $APLS = 1 - min(1, \frac{|1027.07-947.93|}{947.93}) = 0.917$.

(a) Ground-truth (b) [5] (c) [6] (d) [7] (e) Ours

Figure 9: Qualitative visualization of sample image tiles (5000×5000-sized). (a) The ground truth (cyan lines). (b) Results of OrientationRefine (green lines). (c) Graph obtained by enhanced-iCurb (yellow points as vertices and orange lines as edges). (d) Graph obtained by Sat2Graph (orange lines). It cannot present reasonable results since its encoding scheme cannot be well adapted to our task. (e) Graph obtained by csBoundary. Yellow points are normal vertices, red points are rounded-corner vertices, orange lines are normal edges and cyan lines are edges connecting corresponding rounded-corner vertices. This figure is best viewed in color. Please zoom in for details.

4 Additional visualizations

We show the visualization of tile-level graphs in Fig. 9, which provides more details for comparison.

References

[1] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 2117–2125.

[2] A. V. 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*, 2020, pp. 1786–1795.

[3] A. Van Etten, D. Lindenbaum, and T. M. Bacastow, "Spacenet: A remote sensing dataset and challenge series," *arXiv preprint arXiv:1807.01232*, 2018.

[4] J. D. Wegner, J. A. Montoya-Zegarra, and K. Schindler, "A higher-order crf model for road network extraction," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 1698–1705.

[5] A. Batra, S. Singh, G. Pang, S. Basu, C. Jawahar, and M. Paluri, "Improved road connectivity by joint learning of orientation and segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 10 385–10 393.

[6] Z. Xu, Y. Sun, and M. Liu, "Topo-boundary: A benchmark dataset on topological road-boundary detection using aerial images for autonomous driving," *IEEE Robotics and Automation Letters*, vol. 6, no. 4, pp. 7248–7255, 2021.

[7] S. He, F. Bastani, S. Jagwani, M. Alizadeh, H. Balakrishnan, S. Chawla, M. M. Elshrif, S. Madden, and M. A. Sadeghi, "Sat2graph: road graph extraction through graph-tensor encoding," in *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIV 16*. Springer, 2020, pp. 51–67.