GAReT: Cross-view Video Geolocalization with Adapters and Auto-Regressive Transformers

Supplementary Material

1 Overview

We organize our supplementary material into the following sections: Sec. 2 presents our implementation details comprehensively. In Sec. 3, we showcase the retrieval performance of our method followed by Sec. 5, where we compare and contrast GPS trajectories obtained using the nearest neighbor retrieval and our proposed TransRetriever. In Sec. 4, we present additional experiments and ablations for our method. Finally, in Sec. 6, we visually examine the attention outputs of our GeoAdapter module to gain a deeper understanding of our framework.

2 Implementation Details

2.1 General

We implement our method using PyTorch [1]. We use DeiT [6] as the image encoders for both views. For stable training, we utilize ASAM with Adam as the base optimizer with $\rho = 2.5$, weight decay of 0.03, and a learning rate of 10^{-4}. We also employ the global sampling strategy from [9–11] during training. All the models are trained for 50 epochs. For image pretraining, the street-view input resolution is (216×384) while the aerial-view input resolution is (256×256).

2.2 Architecture diagram of TransRetriever

Fig. 1 presents the architecture diagram of our proposed TransRetriever module.

2.3 Baseline 1 & Baseline 2

We use the DeiT transformer to implement both baselines. For Baseline 1, we individually obtain embeddings for all frames in a video and compute the average pooling operation to obtain the video embedding. As discussed in Sec. 3.3 (main paper), the large aerial images are preprocessed to form a sequence of inputs. For Baseline 2, we utilize the street-view branch of the image-pretrained model and add our GeoAdapter module. As for the aerial branch, we solely use the aerial-view branch of the image model and train it (without the adapter) entirely with the resized large aerial image as input. Note that, for Baseline 2, the large aerial

image is considered as one single image and is resized to 512×512 resolution before passing to the encoder.

As discussed in Sec. 4.3 (main paper), Baseline 1 performs well in Sequence-to-Image inference, but it doesn't generalize back to the Frame-to-frame inference. This is because the encoders in this baseline are trained entirely on the video data. In contrast, our approach freezes the encoders while the adapters are responsible for aggregating the temporal information. This procedure ensures that we can easily reuse the encoders for Frame-to-Frame inference by disengaging the GeoAdapter module.

Model	R@1 (↑)	R@5 (↑)	R@10 (↑)	R@1% (↑)
Baseline 1	25.01	45.67	55.22	70.43
Ours (T)	**54.64**	**70.45**	**76.36**	**91.92**

Table 1: Top-k retrieval recall score for frame-by-frame geo-localization on the GAMa dataset. For both methods, we construct the gallery $\mathcal{G}$ using the top 10 large aerial images. (Symbols follow the definition in Tab. 1 (main paper))

2.4 Dominant Sets strategy

To implement the dominant sets strategy for our problem, following [5], we consider each small aerial image candidate as a graph node and create an affinity matrix $A = w_{ij}$. As discussed in Sec. 3.4 (main paper), each node or vertex x_{ij} of the graph belongs to a cluster of neighborhoods N_i. These vertices are connected such that, no two vertices from the same cluster are connected, only the vertices of the temporally subsequent neighborhood are connected to the previous one and the distance between two connected vertices is less than t miles assuming that the motion between two subsequent video frames will not be more than t miles (we use $t = 1$ in all our experiments). Formally, we have an ordered set, $P = (N_1, ...N_{n-1}, N_n)$. Two vertices $x_{ij} \in N_i$ and $x_{kl} \in N_k$ are connected if and only if:

- $N_i \cap N_k = \emptyset$ or $i \neq k$
- $i < k$
- $d(x_{ij}, x_{kl}) \leq t$ where $d(x_{ij}, x_{kl})$ is the great-circle distance between two vertices (reference aerial image).

Following [5], we define the edge weight $w_{ij,kl} \in W$ between vertices x_{ij} and x_{kl} as:

$$w_{ij} = \begin{cases} d(x_{ij}, x_{kl}) + \frac{2}{(S_{ij}+S_{kl})}, \text{ if } (x_{ij}, x_{kl}) \in E \\ \infty, \qquad\qquad\qquad\qquad \text{otherwise} \end{cases} \tag{1}$$

where $0 \leq d(x_{ij}, x_{kl}) \leq 1$ is the scaled great-circle distance. Similar to previous works, we use replicator dynamics algorithm [2,3] to select a dominant set from the graph nodes. The final prediction of GPS is obtained using nodes from the dominant set.

3 Visualization of Retrievals

In this section, we present qualitative examples to illustrate the retrieval performance of our method during inference. Fig. 2 showcases the top-5 large aerial images retrieved by our method given a street-view video (Sequence-to-Image inference) while Fig. 3, we present the top-5 small aerial images retrieved given a street-view video frame (A), both by nearest neighbor (1-5) and our proposed TransRetriever module (B) (Frame-to-frame inference). Correct predictions are highlighted with a green outline, and the arrow designates the *small* aerial image chosen as the final prediction by our TOPS-based retrieval. Each column represents the predictions for a street-view frame. It's worth noting that there can be more than one correct prediction for a street-view frame due to the overlap discussed in Sec. 4.4 (main paper). For future reference and effective comparison, we provide the video-id for each example. The results here indicate that our TransRetriever module significantly outperforms nearest neighbor counterpart, improving the recall rate nearly 5 times in this scenario.

3.1 Retrieval with varying distance threshold

To further strengthen the efficacy of our approach, we compare the retrieval performance of our approach with [7] in Frame-to-frame inference in varying distance thresholds. Tab. 2 compares top-1 retrieval score of our method with GAMa [7] in Frame-to-frame inference with varying distance threshold. Following previous works, we show results when the distance threshold is $0.1, 0.2, 0.5$, and 1.0 mile. As expected, the performance improves as we increase the distance threshold, and our method consistently outperforms previous SOTA.

Model	R@0.1 (↑)	R@0.2 (↑)	R@0.5 (↑)	R@1.0 (↑)
GAMa-Net [7]	19.6	23.0	28.7	36.1
GAMa-Net (Hierarchical) [7]	23.5	27.8	34.9	43.6
Ours (T)	**60.73**	**65.34**	**70.19**	**74.68**

Table 2: Top-1 retrieval recall score for frame-to-frame geo-localization on the GAMa dataset with varying distance threshold. R@k represents the recall score when the distance threshold is k miles. (Symbols follow the definition in Tab. 1 (main paper))

4 Additional Experiments & Ablations

We perform an ablation experiment on the recall rate with varying numbers of street-view frames to establish the importance of the number of frames while adapting to video inputs. Tab. 3 (a) shows the result on the GAMa dataset when 2, 4, and 8 frames are used while optimizing our proposed *GeoAdapter* module.

We also compare our method with CVLNet [4] in Tab. 3 (b). To demonstrate the significance of `[START]` in our proposed module *TransRetriever*, we have made modifications to the module, leaving only the decoder and eliminating global context (GC) learning. Tab. 4(a) shows that the model performs poorly without the global context encoded in the `[START]` token. In Sec. 4.3 (main paper), we discussed how Baseline 1 performs well in large aerial inference. However, the encoders used for this purpose are trained exclusively on video data. Therefore, after localizing the larger aerial region, the same encoder cannot be used for frame-by-frame matching. We can see this demonstrated in Tab. 4 (b), where we attempt frame-to-frame localization using the trained Baseline 1 encoder. Finally, to evaluate the generalization abilities of our model, we perform cross-dataset comparison, similar to [8], of our method. Tab. 5 shows the results of the comparison on the GAMa and SeqGeo datasets.

Frames	**R@1**	**R@5**	**R@10**	**R@1%**
2	7.46	19.90	33.05	59.95
4	25.66	48.39	62.42	84.08
8	50.69	81.77	88.71	98.26

(a)

Model	**R@1**	**R@5**	**R@10**	**R@100**
Ours	18.39	43.67	57.82	93.53
CVLNet	21.80	47.92	64.94	99.07

(b)

Table 3: (a) Recall with varying number of street-view frames, (b) Retrieval recall of our method compared to CVLNet [4]

NN	**Ours w/o GC**	**Ours w/ GC**
54.64	58.40	67.66

(a)

Methods	**R@1**	**R@5**	**R@10**
Baseline 1	21.45	39.27	48.49
Ours	54.64	70.45	76.36

(b)

Table 4: Frame-to-frame retrieval on GAMa dataset

Paradigm	**R@1**	**R@5**	**R@10**	**R@1%**
GAMa → SeqGeo				
No Training	0.96	3.95	7.05	25.02
From Scratch	3.34	11.19	17.18	44.39
Finetuning	4.07	14.17	21.51	52.44

(a)

Paradigm	**R@1**	**R@5**	**R@10**	**R@1%**
SeqGeo → GAMa				
No Training	8.44	27.30	40.03	58.02
From Scratch	50.69	81.77	88.71	98.26

(b)

Table 5: Cross-dataset evaluation

5 GPS Trajectories

The primary objective of Cross-view Video Geo-localization (CVGL) is to obtain a GPS trajectory given a street-view video. In this section, we explore the quality of the trajectory obtained using our method, with and without our proposed TransRetriever module. Fig. 4 presents some examples of the trajectory obtained using our method when we use top-5 *large* aerial image during Sequence-to-Image inference. The first column depicts trajectories utilizing nearest neighbor (NN) retrieval, while the second column employs TransRetriever for retrieval. Note that the aerial images shown in the figure don't all have the same Ground Sampling Distance. As the NN predictions are way off from each other, we adjust the zoom level to incorporate the entire trajectory in the image. Our TransRetriever module, as expected, significantly stabilizes the trajectories and produces temporally consistent output.

To further establish the efficacy of our method, we compare the predicted trajectories with ground truth in Fig. 5 and 6. Our entire pipeline with GeoAdapter and TransRetriever predicts GPS trajectories with incredible precision. In Fig. 6, we can see our method filtering inconsistent predictions to obtain stable trajectories.

6 Attention Visualization

To have a deeper look at our proposed GeoAdapter module, we visualize the temporal attention maps computed in the final block of the unified model $\mathbf{U}$. In Fig. 7, for each street-view frame, we compute the temporal attention maps and overlay them on the large aerial image (shown below). Each square block is a patch ($I^a_{p_i}$) of the aerial image as described in Sec. 3.3 (main paper). The lighter regions depict higher similarity between the patch and the street-view frame, while darker regions depict lower similarity.

To compute the attention maps for each street-view frame, we first obtain its class token feature along with the class token features of all patches using respective feature encoders. Then we calculate self-attention by using the street-view feature as the query vector and the aerial patch features as the key and value vectors. We utilize the attention parameters from the penultimate block of the aerial-view transformer encoder. It can be inferred from these attention maps that our aggregation module finds relevant patches from large aerial image before aggregating the features to obtain the final embedding.

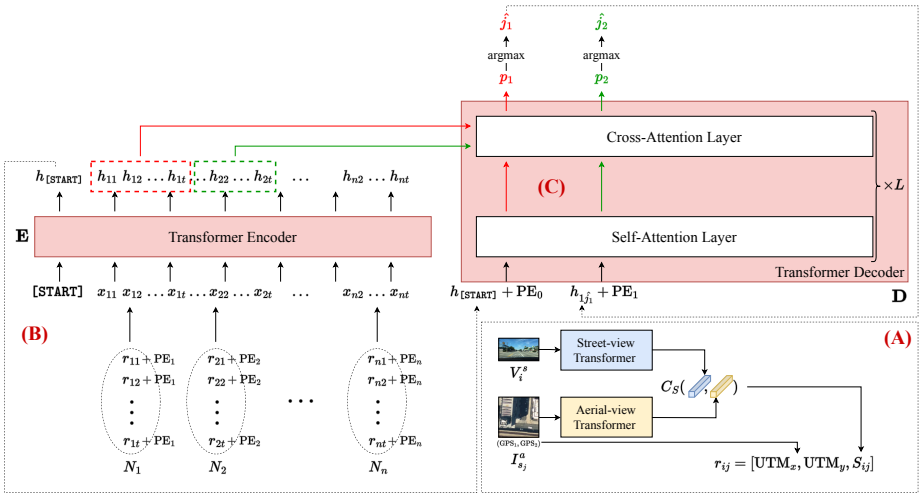

Fig. 1: Architecture of our TransRetriever module. **(A)** We begin by obtaining a 3-dimensional representation for each small aerial image $I^a_{s_j}$. Precisely, given a street-view frame V^s_i, we obtain t nearest neighbor small aerial images from the gallery. For each aerial image $I^a_{s_j}$, we first convert its GPS coordinates to UTM coordinates, followed by computing the cosine similarity C_S between its embedding and the embedding of the street-view frame V^s_i. The converted coordinates and the similarity score S_{ij} are concatenated to form a 3-dimensional vector r_{ij} for the aerial image. **(B)** We follow this procedure for all n street-view frames in a video to obtain n clusters of t elements each. We append a learnable token `[START]` and obtain input tokens to the encoder by adding positional embeddings to each cluster N_i. Note that elements of the same cluster are added with the same positional embedding. This ensures that the positional information is based on the ordering of the cluster and not the individual elements. The tokens are encoded using the transformer encoder **E** to obtain $h_{\texttt{[START]}}$ and $\{h_{ij}\}$. **(C)** The token $h_{\texttt{[START]}}$ is added with positional embedding and passed through the decoder **D** to obtain a probability distribution p_1. To make predictions in the decoder, two types of attention layers are used. First, a causal self-attention layer is computed which helps to predict tokens based on the previous tokens. Second, a cross-attention layer is used to take encoded tokens from corresponding clusters that are needed to make the prediction. For instance, to predict the first token, the cross-attention layer will be computed across the encoded tokens of cluster N_1. p_1 represents the probability of each encoded token in N_1 to be selected. The selected encoded token is then added with the next positional embedding and passed through the decoder to obtain the predictions for the second frame and so on.

Fig. 2: Top-5 large aerial images retrieved using our method for a given street-view video. The green outline indicates correct predictions. For effective comparison in future works, we have provided the video-id for each sample we present here.

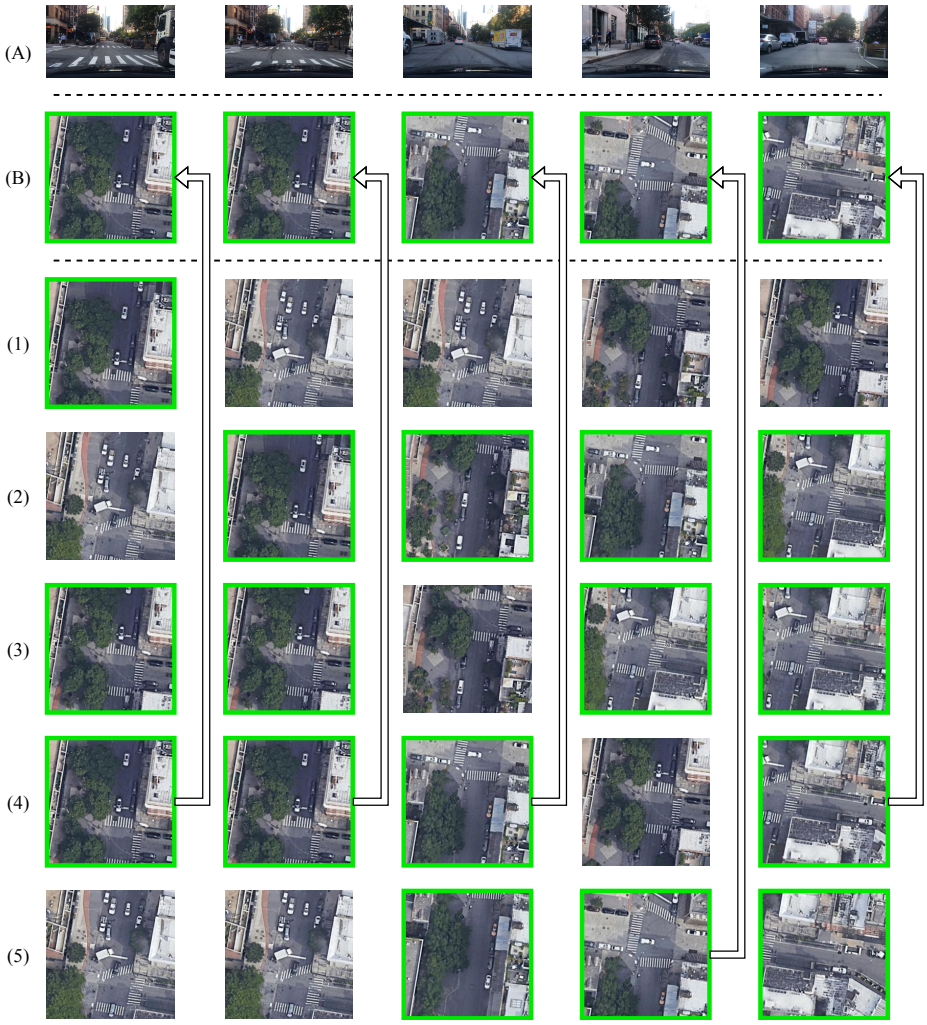

Fig. 3: Top-5 small aerial image retrieved given street-view video (`c1a07cae-67a51c91`) frames (A) by both nearest neighbor (1-5) and our proposed TransRetriever module (B) retrieval. The correct predictions are highlighted with a green outline. The arrow represents the *small* aerial image chosen as the final prediction using our TransRetriever module. Each column represents the predictions for a street-view frame.

Fig. 4: Example trajectories obtained using our method when we use top-5 large aerial images in Sequence-to-Image inference. The first column depicts trajectories utilizing nearest neighbor (NN) retrieval, and the second column employs our TransRetriever module. Note that the aerial images shown here don't all have the same Ground Sampling Distance due to the NN predictions being way off from each other. Our proposed TransRetriever module significantly stabilizes the trajectories, yielding temporally consistent output.

Fig. 5: Example trajectories obtained using our method compared with ground-truth trajectories. Note that the aerial images shown here don't all have the same Ground Sampling Distance due to the nearest neighbor (NN) predictions being way off from each other. Here, it's evident that our proposed TransRetriever can effectively recover temporally faithful trajectories with remarkable precision, even in cases where the nearest neighbor approach terribly fails to converge.

Fig. 6: Example trajectories obtained using our method compared with ground-truth trajectories. Here, we can see that our proposed TransRetriever module is able to filter temporally inconsistent predictions and stabilize the trajectory.

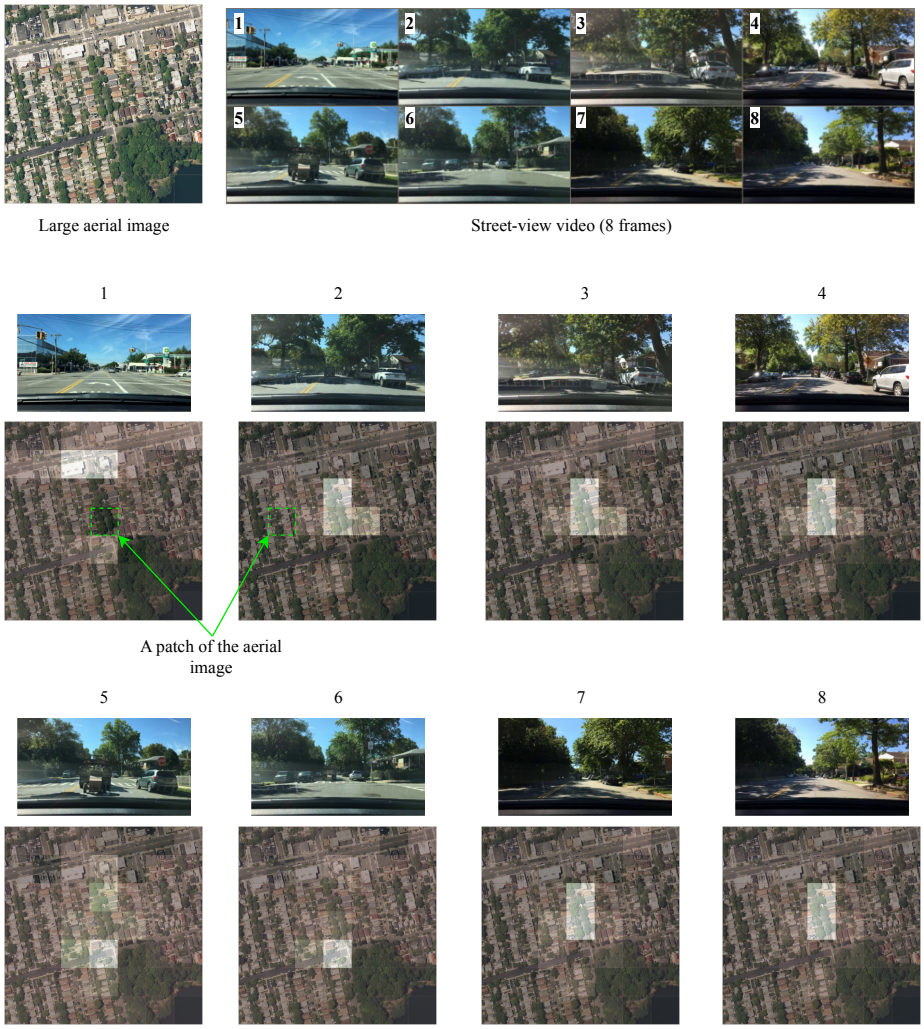

Fig. 7: The attention visualization computed for our proposed GeoAdapter module. The first row shows a matching pair of large aerial image and street-view video. For each street-view frame, we compute the temporal attention maps and overlay them on the large aerial image (shown below). Each square block is a patch ($I^a_{p_i}$) of the aerial image as described in Sec. 3.3 (main paper). The lighter regions depict higher similarity between the patch and the street-view frame, while darker regions depict lower similarity.

References

1. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al.: Pytorch: An imperative style, high-performance deep learning library. Advances in neural information processing systems **32** (2019)
2. Pavan, M., Pelillo, M.: A new graph-theoretic approach to clustering and segmentation. In: 2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings. vol. 1, pp. I–I. IEEE (2003)
3. Pavan, M., Pelillo, M.: Dominant sets and pairwise clustering. IEEE transactions on pattern analysis and machine intelligence **29**(1), 167–172 (2006)
4. Shi, Y., Yu, X., Wang, S., Li, H.: Cvlnet: Cross-view semantic correspondence learning for video-based camera localization. In: Asian Conference on Computer Vision. pp. 123–141. Springer (2022)
5. Tian, Y., Chen, C., Shah, M.: Cross-view image matching for geo-localization in urban environments. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3608–3616 (2017)
6. Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., Jegou, H.: Training data-efficient image transformers & distillation through attention. In: Meila, M., Zhang, T. (eds.) Proceedings of the 38th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 139, pp. 10347–10357. PMLR (18–24 Jul 2021), `https://proceedings.mlr.press/v139/touvron21a.html`
7. Vyas, S., Chen, C., Shah, M.: Gama: Cross-view video geo-localization. In: European Conference on Computer Vision. pp. 440–456. Springer (2022)
8. Zhang, X., Li, X., Sultani, W., Chen, C., Wshah, S.: Geodtr+: Toward generic cross-view geolocalization via geometric disentanglement. arXiv preprint arXiv:2308.09624 (2023)
9. Zhu, S., Shah, M., Chen, C.: Transgeo: Transformer is all you need for cross-view image geo-localization. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 1162–1171 (2022)
10. Zhu, S., Yang, T., Chen, C.: Revisiting street-to-aerial view image geo-localization and orientation estimation. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision. pp. 756–765 (2021)
11. Zhu, S., Yang, T., Chen, C.: Vigor: Cross-view image geo-localization beyond one-to-one retrieval. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 3640–3649 (2021)