

Flexible and Efficient Spatio-Temporal Transformer for Sequential Visual Place Recognition

Yu Kiu (Idan) Lau, Chao Chen, Ge Jin, Chen Feng[✉]

Abstract—Sequential Visual Place Recognition (Seq-VPR) leverages transformers to capture spatio-temporal features effectively. In practice, a transformer-based Seq-VPR model should be flexible to the number of frames per sequence (seq-length), deliver fast inference, and have low memory usage to meet real-time constraints. However, existing approaches prioritize performance at the expense of flexibility and efficiency. To address this gap, we propose Adapt-STformer, a Seq-VPR method built around our novel *Recurrent Deformable Transformer Encoder (Recurrent-DTE)*, which uses an iterative recurrent mechanism to fuse information from multiple sequential frames. This design naturally supports variable seq-lengths, fast inference, and low memory usage. Experiments on the Nordland, Oxford, and NuScenes datasets show that Adapt-STformer boosts recall by up to 17% while reducing sequence extraction time by 36% and lowering memory usage by 35% relative to our best comparable baseline. Our code is released at <https://ai4ce.github.io/Adapt-STFormer/>.

I. INTRODUCTION

Sequential Visual Place Recognition (Seq-VPR) extends traditional VPR by leveraging multiple consecutive frames to construct more robust descriptors for a given location. By incorporating temporal information, Seq-VPR produces more consistent outputs, which is especially beneficial under challenging conditions such as low light or illumination changes [1, 2]. Methods for exchanging and aggregating spatial-temporal features across frames generally fall into two categories. The first type neglects complex cross-frame temporal interactions. For example, JIST collapses per-frame embeddings into a single descriptor via learnable mean pooling, without direct frame-to-frame interaction [1, 3, 4]. The second type draws inspiration from video understanding [5, 6, 7], where recent Seq-VPR methods integrate transformer modules to model both inter-frame (temporal) and intra-frame (spatial) interactions.

However, existing transformer-based Seq-VPR research prioritizes performance over flexibility and efficiency. In practice, a Seq-VPR model should be flexible with respect to the number of frames per sequence (seq-length). It should also be efficient in inference speed and memory usage to meet the real-time constraints of VPR systems. At present, we believe no existing transformer-based Seq-VPR model is both efficient and flexible. Some approaches use a standard transformer layer to learn temporal interactions after

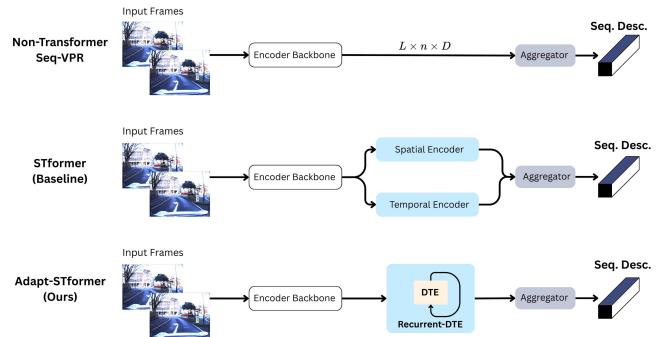


Fig. 1: Comparison of Seq-VPR architectures. Top: Non-transformer methods feed backbone features directly to an aggregator. Middle: STformer handles spatial and temporal modeling with separate encoders before aggregation. Bottom: The proposed Adapt-STformer unifies spatio-temporal modeling in the Recurrent-DTE module before aggregation.

extracting spatial features. However, as each additional frame introduces more tokens, the transformer’s computation grows quadratically. STformer [8] reduces computation by limiting temporal attention to local windows across frames. While this windowing lowers costs, it is not flexible with respect to seq-length. This constraint reduces practical usability: when frames are missing/distorted, the model can only rely on padding/dropping to satisfy the required seq-length. Moreover, STformer keeps separate spatial and temporal modules, adding significant overhead, leaving STformer neither flexible nor efficient.

To address this gap, we propose Adapt-STformer, a Seq-VPR framework centered on our novel Recurrent Deformable Transformer Encoder (Recurrent-DTE). Unlike prior works that process multiple frames in parallel [9, 10], our approach unifies spatio-temporal fusion within a single holistic module. The Recurrent-DTE recurrently applies a lightweight Deformable Transformer Encoder (DTE) [11] to frames in temporal order, from the earliest to the most recent. This design naturally encodes temporal dependencies, supports arbitrary seq-lengths, and achieves a favorable balance of flexibility, efficiency, and performance. We highlight the main contributions of this work as follows:

- We present Adapt-STformer, a Seq-VPR architecture that eliminates the restriction of fixed seq-length through a lightweight unified framework, thereby achieving both flexibility and efficiency.

[✉]Corresponding author. All authors are with the Center for Robotics and Embodied Intelligence (CREO) at New York University, Brooklyn, NY 11201, USA. cfeng@nyu.edu

The work was supported in part through NSF grants 2238968, 2322242, and 2514030.

- We introduce Recurrent-DTE, a recurrent mechanism embedded in a DTE that iteratively processes frames in temporal order, enabling spatio-temporal modeling within a unified module. This is an effective recombination of well-known components: recurrent sequence modeling from time-series learning and DTE from object detection, for seq-VPR.
- We conduct extensive experiments on the Nordland, Oxford, and NuScenes datasets, where Adapt-STformer achieves balanced improvements, increasing recall by 10% on average while reducing inference latency by 36% and lowering memory usage by 35% relative to our best comparable baseline.

II. RELATED WORKS

A. Classic Seq-VPR

Early Seq-VPR methods relied on handcrafted descriptors combined across frames, but these lacked robustness to environmental changes [12]. The introduction of learning-based approaches, particularly CNN-based models, marked a shift toward data-driven sequence aggregation. SeqNet [1] applied a lightweight 1D temporal convolution to merge a sequence into a single compact descriptor. Building on this idea, SeqVLAD [3] extended NetVLAD [13] by pooling features from all frames into a unified descriptor set and applying VLAD’s soft-assignment and residual aggregation, producing a fixed-length embedding adaptable to different sequence lengths. With a focus on efficiency, JIST [4] proposed SeqGeM, a learnable mean pooling mechanism for compact sequence representation. While effective, these non-transformer Seq-VPR methods model temporal information only through simple aggregation, limiting their ability to capture complex spatio-temporal dependencies and leading to performance drops in challenging conditions [8]. Our work seeks to retain their efficiency and flexibility while addressing these limitations.

B. Transformer-based Seq-VPR

With the rise of transformers in vision tasks [6, 7], Seq-VPR methods have also adopted them to better model spatio-temporal dependencies. STFormer [8] introduced a two-stage design with a spatial transformer for frame-level features and a sliding-window temporal transformer for sequence modeling, achieving strong performance but limited by fixed sequence lengths and high computational cost due to separate encoders and dense attention. More recently, CaseVPR [14] leveraged a DINOv2 backbone [15] and correlation-aware embedding, reaching SOTA results with seq-length flexibility, but at the expense of heavy computation, slow inference, and large memory use. In short, existing transformer-based Seq-VPR methods trade flexibility and/or efficiency for performance.

C. Computationally Efficient Transformer

Deformable Transformers, proposed in [11], employ deformable attention, where each query attends to a sparse set of learned offsets, thereby reducing the quadratic cost

of global self-attention on high-resolution features. Xu et al. [16] demonstrated the effectiveness of Deformable Transformer Encoders (DTE) for multi-camera VPR, although without modeling temporal features. BEVFormer [2] integrates temporal cues by recurrently fusing past BEV features and further employs a Temporal Self-Attention (TSA) layer in addition to its DTE for spatial lookup across multiple camera views. In contrast, our design unifies spatial and temporal reasoning directly within a DTE, avoiding the need for a separate TSA layer. We found that replicating BEVFormer’s additional TSA increases inference time and memory usage while not improving performance. Beyond these architectural differences, BEVFormer assumes a synchronized multi-camera setup with known camera intrinsics. In contrast, we target a practical monocular front-view scenario without access to calibration parameters, making our method applicable to real-world settings.

D. Computation efficient backbone

The choice of encoder backbone is crucial for our method’s performance and efficiency. Early Seq-VPR methods commonly use encoder backbones such as ResNet [17] valued for their computational efficiency but limited in representational power. In contrast, modern Vision Transformer (ViT) backbones [15, 18] offer rich, high-quality visual features learned from massive self-supervised training, yet they incur substantial computational cost. Compact Convolutional Transformer (CCT) [19] represents a hybrid of convolutional tokenization and transformer layers, showing promising performance while remaining relatively lightweight compared to pure ViT [20] counterparts. Motivated by this balance of efficiency and performance, we adopt CCT as the encoder backbone for our Seq-VPR framework.

METHODOLOGY

TABLE I: SUMMARY OF MAJOR NOTATIONS FOR ADAPT-STFORMER.

L	Number of frames/seq-length
t	Iteration/frame index, $t \in \{1, \dots, L\}$
f_t	The encoder output feature from the t th image
n	Number of tokens in image feature f_t
C	Number of VLAD clusters
D	Token embedding dimension
\hat{f}_t	DTE output at t
F	The set of encoder outputs across the sequence
(Q_t, K_t, V_t)	Query, Key, and Value at iteration t

A. Overall Architecture

Adapt-STformer takes as input a frame sequence $S = \{s_i\}_{i=1}^L$, with each $s_i \in \mathbb{R}^{H \times W \times 3}$, and outputs a NetVLAD descriptor $V \in \mathbb{R}^{C \times D}$. There are three intermediate stages between input and output.

- **Encoder Stage:** Encode S with CCT to obtain $F = \{f_t\}_{t=1}^L \in \mathbb{R}^{L \times n \times D}$.
- **Recurrent-DTE Stage:** Refine F iteratively via Recurrent-DTE, producing $\hat{F} = \{\hat{f}_t\}_{t=1}^L \in \mathbb{R}^{L \times n \times D}$.
- **Aggregation Stage:** Aggregate \hat{F} with SeqGeM and SeqVLAD into $V \in \mathbb{R}^{C \times D}$.

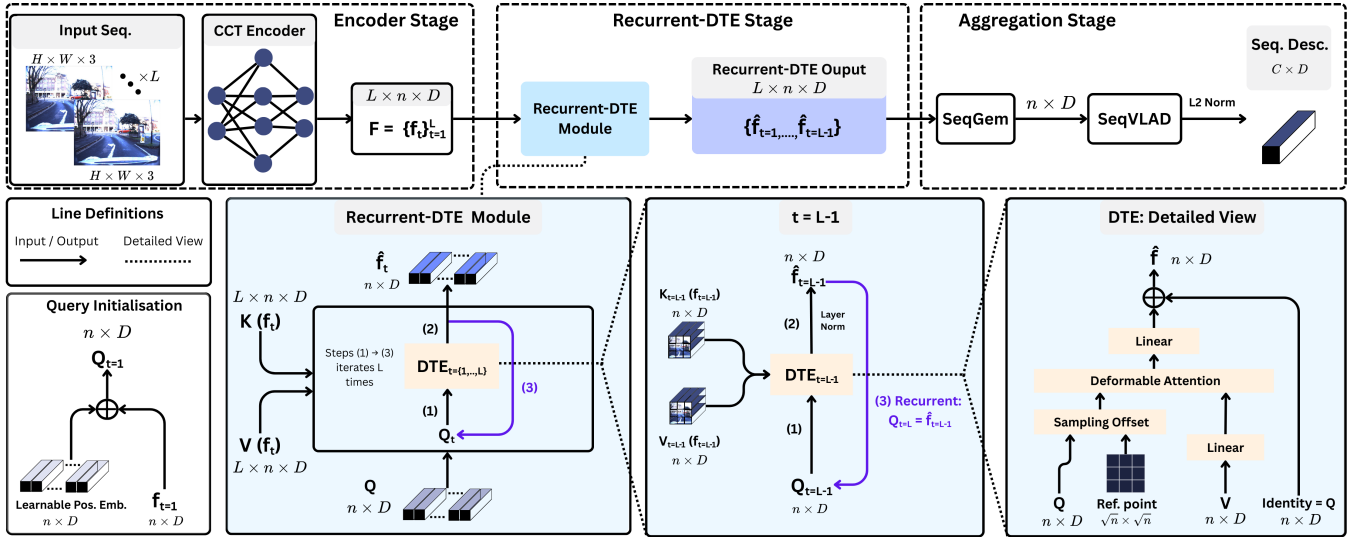


Fig. 2: **Proposed architecture of Adapt-STformer.** A frame sequence S enters the **Encoder Stage**, where the CCT384 backbone tokenizes each input frame, producing $F = \{f_t\}_{t=1}^L$. **Recurrent-DTE Stage** performs *iterative recurrent* spatio-temporal modeling on F : at t , the previous DTE output provides $Q_t = \hat{f}_{t-1}$ and the current frame supplies $K_t = V_t = f_t$, with $Q_{t=1} = f_{t=1} + \Delta$; this recurrence yields $\{\hat{f}_t\}_{t=1}^L$. **Aggregation Stage** stacks $\{\hat{f}_t\}$ into a tensor \hat{F} , applies SeqGeM to obtain \tilde{F} , and processes it with SeqVLAD into the final sequential descriptor.

B. Encoder Stage:

We adopt CCT384 as our encoder backbone to extract initial features. Given a sequence $S = \{s_i\}_{i=1}^L$ with $s_i \in \mathbb{R}^{H \times W \times 3}$, the encoder maps each frame into a tokenized feature representation.

$$F = \{f_t\}_{t=1}^L \in \mathbb{R}^{L \times n \times D},$$

The initial encoder output is arranged as $n \times D$, we reinterpret it as $D \times H \times W$. We then flatten $H \times W$ and permute the tensor back to $n \times D$ but with a reordered representation.

C. Recurrent-DTE Stage:

DTE: We choose DTE [11] for its deformable attention computation, where each query’s attention is restricted to its reference point’s neighboring keypoints. Following [11, 2], we overlay each feature map with a uniform 2D reference point grid and learn per query offsets to dynamically select sampling locations. In our setting, we use a single feature level instead of the multi-scale design in the original DTE [11] to reduce compute cost and keep our model architecture simple. Furthermore, unlike works that employ DTE for 3D perception [2, 16], we restrict our design to 2D feature modeling. This avoids the additional computation needed for 3D spatial modeling and needing known camera intrinsics, making our method deployable across diverse hardware setups.

Recurrent-DTE: To capture temporal dependencies across frames of S , we introduce a recurrent mechanism that, at frame index t , uses the previous frame’s DTE output \hat{f}_{t-1} together with the current frame’s encoder features f_t . Here, \hat{f}_{t-1} acts as a hidden state shaped by iterations $\{1, \dots, t-1\}$. At iteration t , we set \hat{f}_{t-1} as Q_t and f_t as K_t and V_t .

Because each iteration of Recurrent-DTE relies only on the output of the previous one, the module is seq-length agnostic. For initialization, $Q_{t=1}$ is obtained by adding a learnable positional embedding $\Delta \in \mathbb{R}^{n \times D}$ to $f_{t=1}$, Δ is applied only at $t = 1$.

$$\begin{aligned} \hat{f}_t &= \text{DTE}(Q_t = \hat{f}_{t-1}, K_t = V_t = f_t) \\ Q_{t=1} &= f_{t=1} + \Delta \end{aligned} \quad (1)$$

This design differs fundamentally from STformer [8], where spatial and temporal information are learned by separate transformer encoders. A simple equivalent formulation of STformer using our notation is

$$\{\hat{f}_t\}_{t=1}^L = \mathcal{T}_{\text{temporal}}\left(\mathcal{T}_{\text{spatio}}(\{f_t\}_{t=1}^L)\right), \quad (2)$$

where $\mathcal{T}_{(\cdot)}$ denotes a transformer encoder, with its modeling scope indicated by the subscript. In our approach, temporal information is instead implicitly captured within the same transformer operation as spatial modeling. Recurrent-DTE therefore unifies spatio-temporal modeling within a single module.

$$\hat{f}_t = \mathcal{T}_{\text{spatio-temporal}}(\hat{f}_{t-1}, f_t) \quad (3)$$

D. Aggregation Stage

We concatenate per-iteration outputs $\{\hat{f}_{t=1}, \dots, \hat{f}_{t=L}\}$ into the tensor $\hat{F} \in \mathbb{R}^{L \times n \times D}$ and permute \hat{F} to become $\hat{F}' \in \mathbb{R}^{n \times L \times D}$ (treating tokens as the batch dimension) and apply SeqGeM [4], a learnable mean pooling over the temporal axis, collapsing L to 1:

$$\tilde{F} = \text{SeqGeM}(\hat{F}') \in \mathbb{R}^{1 \times n \times D} \quad (4)$$

We then feed \tilde{F} into SeqVLAD [3], which aggregates the n D -dimensional embeddings into a descriptor $V \in \mathbb{R}^{C \times D}$. Both SeqGeM and SeqVLAD are flexible to the value of L .

EXPERIMENT SETUP & IMPLEMENTATION DETAILS

TABLE II: DATASET DETAILS SPECIFYING THE NUMBER OF IMAGES IN THE DATABASE AND QUERY SETS.

Dataset	Split	Database / Queries
NordLand	Train set	15000 / 15000
	Test set	3000 / 3000
Oxford-Easy	Train set	3619 / 3962
	Test set	3632 / 3921
Oxford-Hard	Train set	2322 / 2585
	Test set	2970 / 2920
NuScenes	Test set	4500 / 4000

A. Dataset

NordLand: The Nordland dataset [21] comprises a collection of images captured during rail journeys across four seasons. We use the Summer-Winter pair for training, and the Spring-Fall pair for testing.

Oxford RobotCar: The Oxford RobotCar dataset [22, 23] provides large-scale urban driving sequences under diverse conditions. We construct two sub-datasets: *Oxford-Easy*, following the Oxford2 split in [8], which uses all six route sections with 2 meter frame separation; and *Oxford-Hard*, built from day–night traversals with disjoint route segments (sections 1–2 for training, 3–4 for testing) and 1 meter frame separation. Oxford-Hard is more challenging due to both route mismatch and severe lighting changes between training and test sets.

NuScenes: The NuScenes dataset [24] is a large-scale autonomous driving benchmark collected in urban environments. We select the Singapore scenes to capture urban day–night traversals under extreme lighting transitions. We split scenes into day and night and treat pairs with less than 30° of angular difference as positives, following [25]. To assess *cross-dataset generalization*, we evaluate a model trained on Oxford-Easy directly on the NuScenes dataset.

B. Implementation Detail

Model hyperparameters: For the DTE, we use 8 attention heads, 8 sampling points, 2 levels (with only 1 feature level input), a dropout rate of 0.1, and 64 VLAD clusters.

Training: Following SeqVLAD [3], we initialize our CCT encoder from its authors’ pretrained weights [19] and use the ADAM optimizer [26] with a learning rate of 1×10^{-5} . Similar to [1, 3], we adopt the triplet loss from NetVLAD [13], selecting one non-trivial positive sample from descriptor space and five hardest negative samples from GNSS ground truths. During training, we set input image resolution to 384×384 , batch size to four, sequence length to five, and triplet margin to 0.1. Training stops early if Recall@5 does not improve for 5 straight epochs.

C. Baseline Seq-VPR Methods

As the choice of encoder backbone greatly influences performance and efficiency, we first compare Adapt-STformer (ours) with four representative Seq-VPR baselines without a foundational encoder backbone. Namely, we experiment with SeqNet [1], SeqVLAD [3], STformer [8], and JIST [4]. In section II-D, we evaluate Adapt-STformer against SOTA VPR methods [18, 14] that utilize a foundational encoder backbone. For a fairer comparison with such methods trained on much larger datasets, we also analyze their performance under inference time constraints, which produces further insights on the value of Adapt-STformer’s efficiency.

Evaluation Criteria. Following prior work, we measure VPR performance by Recall@K, which counts a query correct if at least one of the top-K retrieved candidates lies within 10 m for Oxford RobotCar and NuScenes, or within ± 1 frame for Nordland.

RESULTS

A. Recall comparison

We report the evaluation results of Adapt-STformer and Seq-VPR baselines in Table III. We calculate Recall@K for top K values of 1, 5, and 10 and measure the memory usage and duration of one forward pass for each model.

Nordland and Oxford-Easy: SeqVLAD, STformer, and our method achieve near-perfect recall on Nordland. This can be attributed to Nordland’s minor lighting variations between the query and the database, making it an easy dataset. On the Oxford-Easy dataset, we achieve a +3% R@1 gain over STformer. While Oxford-Easy presents more lighting challenges than Nordland, its train and test sets share overlapping locations, simplifying the task.

Oxford-Hard and NuScenes: Substantial gains are observed on harder benchmarks: Oxford-Hard (poor lighting, rain, and unseen sections) and NuScenes (poor lighting and cross-domain train-test splits). On Oxford-Hard, we achieve approximately +20% gains in R@1,5,10 over SeqVLAD. For NuScenes, we observe +10%,17%,17% improvements in R@1,5,10, respectively, over STformer.

PCA: We standardise descriptor sizes via PCA [27] for fairness. The relative performance ranking of the methods remains unchanged.

B. Resource Usage

Computational efficiency is critical for practical deployment of VPR systems and a central focus of our work. Table III reports the overall resource usage of all methods. In table IV, we make a per-module inference times comparison between Adapt-STformer (ours) and STformer, the most architecturally similar baseline. All experiments were conducted using PyTorch [28] on an NVIDIA Tesla V100.

Transformer Modules: For spatially attending to a single frame, Adapt-STformer’s DTE module is 71.3% faster than the non-deformable transformer encoder in STformer. Also when spatio-temporally processing an entire query, our Recurrent-DTE architecture uses 88.1% less time than STformer’s additional temporal transformer encoder.

TABLE III

ADAPT-STFORMER IS COMPARED WITH FOUR SEQ-VPR BASELINES IN TERMS OF RECALL, MEMORY, AND INFERENCE TIME PER SEQUENCE. BEST RESULTS ARE SHOWN IN **BOLD**, SECOND-BEST IN UNDERLINE. WE DON'T QUANTIFY PCA IMPACT ON RESOURCE USAGE BECAUSE IT IS POST DESCRIPTOR EXTRACTION. PUBLIC CHECKPOINT(PC)

Descriptor Dimension	Method	Backbone	NuScenes			Oxford-Hard			Oxford-Easy			Nordland			Resource Usage	
			R@1 ↑	R@5 ↑	R@10 ↑	R@1 ↑	R@5 ↑	R@10 ↑	R@1 ↑	R@5 ↑	R@10 ↑	R@1 ↑	R@5 ↑	R@10 ↑	Mem (MB) ↓	Time (s) ↓
24576	Seqvlad	CCT384	0.3236	0.4327	0.5142	0.4709	0.5601	0.6151	0.8275	0.9249	0.9541	0.9603	0.9947	<u>0.9947</u>	177.66	<u>0.0181</u>
	STformer ^{PC}	CCT384	<u>0.4461</u>	<u>0.5319</u>	<u>0.5945</u>	<u>0.5754*</u>	<u>0.6548*</u>	<u>0.7011*</u>	<u>0.8488</u>	<u>0.9323</u>	<u>0.9633</u>	<u>0.9697</u>	0.9947	0.9950	276.55	0.0277
	Ours	CCT384	0.5496	0.7035	0.7622	0.6940	0.7706	0.7969	0.8854	0.9528	0.9700	0.9763	<u>0.9943</u>	<u>0.9947</u>	<u>180.39</u>	0.0178
4096 (PCA)	Seqnet	NetVLAD	0.0827	0.1575	0.2031	0.3064	0.4505	0.5228	0.5363	0.7467	0.8241	0.7943	0.9013	0.9317	676.09	0.0014
	Seqvlad	CCT384	0.3236	0.4327	0.5146	0.4603	0.5550	0.6151	0.8280	0.9251	0.9557	0.9567	0.9947	<u>0.9947</u>	177.66	0.0181
	STformer ^{PC}	CCT384	0.4461	0.5319	0.5949	<u>0.5648*</u>	<u>0.6453*</u>	<u>0.6968*</u>	<u>0.8534</u>	<u>0.9372</u>	<u>0.9667</u>	0.9700	0.9947	0.9950	276.55	0.0277
	Ours	CCT384	0.5488	0.7031	0.7622	0.6874	0.7659	0.7918	0.8882	0.9544	0.9710	<u>0.9610</u>	<u>0.9923</u>	0.9937	<u>180.39</u>	<u>0.0178</u>
512 (PCA)	JIST ^{PC}	ResNet	0.2047	0.3146	0.3713	0.4423	0.5405	0.6139	0.5868	0.7416	0.8134	0.8210	0.9200	0.9450	159.65	0.0053
	Seqvlad	CCT384	0.3146	0.4236	0.5024	0.4383	0.5330	0.6049	0.8236	0.9244	0.9564	<u>0.9567</u>	0.9947	<u>0.9947</u>	<u>177.66</u>	0.0181
	STformer ^{PC}	CCT384	<u>0.4445</u>	<u>0.5370</u>	<u>0.5969</u>	<u>0.5377*</u>	<u>0.6198*</u>	<u>0.6720*</u>	<u>0.8513</u>	<u>0.9369</u>	<u>0.9667</u>	0.9637	<u>0.9933</u>	0.9950	276.55	0.0277
	Ours	CCT384	0.5551	0.7091	0.7717	0.6516	0.7459	0.7765	0.8870	0.9554	0.9731	0.9490	0.9900	0.9943	<u>180.39</u>	<u>0.0178</u>

STformer is evaluated using its pc since the full training code is unavailable. For most datasets, the STformer's pc was trained on the same training sets as ours. However, for Oxford-Hard (*), the training data differs, so we finetune STformer to ensure a fair comparison; these results are reported but not included in our analysis. JIST is also evaluated using a pretrained checkpoint (pc) because its training procedure differs from ours.

TABLE IV: PER-MODULE INFERENCE TIME BREAKDOWN AND OVERALL RESOURCE USAGE COMPARISON BETWEEN OUR METHOD AND STFORMER.

Method	Inference Time (ms)				Memory↓ (MB)
	Spatial Attn. ↓	Spatio-Temp. ↓	Aggr. ↓	Overall ↓	
STformer	1.71	23.02	3.98	27.7	276.55
Ours	0.49	2.73	1.72	17.8	180.39
% Improve	71.3%	88.1%	56.8%	35.7%	34.8%

Note: Our encoder backbone takes approx. 14 ms, accounting for the gap between module times and total inference time.

Aggregation Modules: While both methods employ SeqVLAD for sequence aggregation, Adapt-STformer first applies SeqGeM to collapse the aggregation input shape from $L \times n \times D$ into $1 \times n \times D$, introducing negligible overhead while accelerating the SeqVLAD operation by 56.8%.

Overall Comparison: Overall, Adapt-STformer extracts descriptors 36% faster per sequence and requires 35% less memory than STformer (Table IV). Notably, it has faster extraction than SeqVLAD, a non-transformer method.

D. Ablation Study

We conduct ablation studies on Oxford-Hard and NuScenes to assess the effects of sequence length L and the proposed Recurrent-DTE. All models were trained on $L = 5$ and evaluated at $L = 3$ and $L = 1$, demonstrating flexibility to shorter seq-lengths under limited inference resources [29]

Sequence Lengths: As shown in Table V, performance increases with longer sequences. On Oxford-Hard, our Recurrent-DTE rises from 0.7341 ($L = 1$) to 0.7706 ($L = 5$), while on NuScenes it improves from 0.6224 to 0.7039. This demonstrates the ability of our approach to exploit additional temporal context as L grows effectively.

Spatio-Temporal Methods: Table V further compares different spatio-temporal methods. In the non-recurrent variant (DTE Only), each frame is processed independently, with $Q_t, K_t, V_t = f_t$, yielding no cross-frame interaction. Our

TABLE V: ABLATION STUDY OF SEQ-LENGTHS L AND SPATIO-TEMPORAL METHODS. WE COMPARE: (I) **RECURRENT-TE** (NON-DEFORMABLE TRANSFORMER ENCODER WITH RECURRENCE), (II) **DTE+TT** (DTE WITH A SEPARATE TEMPORAL TRANSFORMER), (III) **DTE ONLY** (OURS WITHOUT RECURRENCE), AND (IV) **RECURRENT-DTE** (OURS).

L	Spatio-Temp. Method	Oxford-Hard	NuScenes	Resource Usage	
		R@5 ↑	R@5 ↑	Mem(MB) ↓	Time(s) ↓
1	Recurrent-TE	0.4776	0.1173	<u>98.15</u>	0.0120
	DTE + TT	0.7368	0.5106	106.02	0.0122
	DTE Only (non-recurrent)	0.7306	<u>0.5283</u>	88.71	0.0111
	Recurrent-DTE (Ours)	<u>0.7341</u>	0.6224	88.71	<u>0.0115</u>
3	Recurrent-TE	0.4552	0.1303	<u>136.75</u>	0.0135
	DTE + TT	<u>0.7384</u>	0.4764	152.03	0.0177
	DTE Only (non-recurrent)	0.7255	<u>0.4898</u>	134.39	<u>0.0126</u>
	Recurrent-DTE (Ours)	0.7549	0.6681	134.39	0.0128
5	Recurrent-TE	0.4085	0.111	<u>182.75</u>	0.0181
	DTE + TT	<u>0.7341</u>	0.4677	198.16	0.0267
	DTE Only (non-recurrent)	0.7247	<u>0.4693</u>	180.39	0.0176
	Recurrent-DTE (Ours)	0.7706	0.7039	180.39	<u>0.0178</u>

Disclaimer: Recurrent-TE and the DTE + TT ablation baselines were integrated with minimal tuning.

Recurrent-DTE consistently outperforms this baseline across all sequence lengths, with gains of up to 18% (NuScenes, $L = 3$), while requiring no additional memory and only negligible runtime overhead. We also evaluate using a separate temporal transformer layer (DTE + TT), similar to [8, 14]. Although competitive on Oxford-Hard ($L = 1$: 0.7368 vs. 0.7341), this approach is substantially weaker on NuScenes ($L = 3$: 0.4764 vs. 0.6681), while being 32% slower (0.0177s vs. 0.0128s) and 12% more memory usage (152.03MB vs. 134.39MB). These results demonstrate that our recurrent-DTE is efficient and effective at spatio-temporal modeling.

DTE Module: Finally, replacing our deformable transformer encoder with a non-deformable variant (Recurrent-TE baseline) degrades both performance and efficiency. As shown in Table V, the deformable design improves recall by +29.4% on Oxford-Hard ($L = 5$: 0.7706 vs. 0.4085) and +492% on

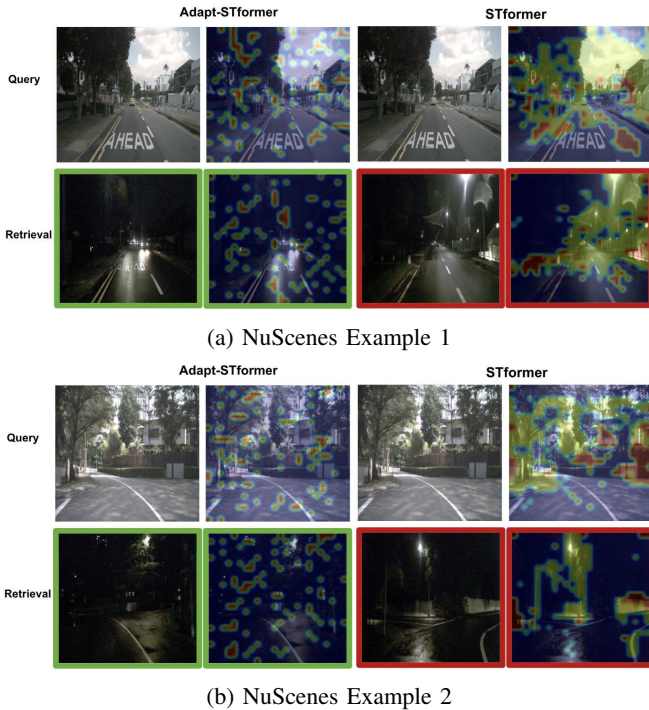


Fig. 3: Qualitative comparison of attention maps between our method and STformer on NuScenes. STformer fails in these cases, whereas our method succeeds in VPR matching. In the attention maps, blue denotes low-focus regions and yellow to red gradient denotes high-focus regions by the respective models.

NuScenes ($L = 5$: 0.7039 vs. 0.111), while simultaneously lowering memory (180.39MB vs. 182.75MB) and inference time (0.0178s vs. 0.0181s).

E. Qualitative Results

Based on the ablation results, DTE is effective at capturing spatial features. Figure 3 shows NuScenes examples where our model retrieves correctly while STformer fails. In both day and night frames, STformer produces smooth, concentrated activations over only a few regions. By contrast, our model displays scattered high-activation patches across the entire frame. This difference, we theorize, arises from the type of attention used. Standard attention, used by STformer, computes attention between every token pair, producing blob-shaped activations around globally salient regions. In contrast, deformable attention, used by our method, learns a small set of sparse offsets that jump directly to features relative to the query token, yielding scattered activations. Under poor lighting, many features are weak, making it crucial to capture any available geometric cues scattered across the scene. Under poor lighting, it is essential to exploit sparse geometric cues across the scene rather than depend on regions salient only under good illumination.

F. Attention Maps Across Adapt-STformer Stages

Figure 4 illustrates the evolution of attention map across the stages of Adapt-STformer when processing a lighting-

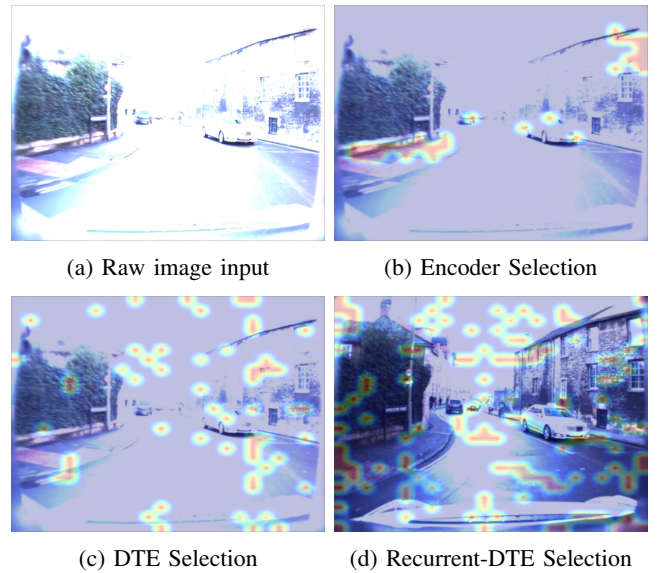


Fig. 4: Attention Maps Across Adapt-STformer Stages. Sub-figs. (b)–(d) show attention activation maps overlaid on the input frame, illustrating model focus at each stage. Color coding follows Fig. 3.

distorted frame (a) drawn from a sequence. After the encoder stage (b), the model focuses on only a few confident regions while largely ignoring areas affected by lighting distortion. Following DTE processing (c), the model attends to a broader and more discriminative set of VPR-relevant features, emphasizing stable structural cues (e.g., walls, road, vegetation) that remain largely invariant across time and environmental conditions. After the Recurrent-DTE stage (d), cross-frame interactions allow the model to incorporate information from other frames in the sequence. As illustrated in (d), where attention is overlaid on a clearer frame from the same sequence, activation expands to a broader set of stable VPR-relevant cues compared to (c), indicating that the model leverages complementary information across frames to recover features affected by lighting distortions.

G. Comparison with SOTA VPR Methods

Though Adapt-STformer outperforms Seq-VPR baselines without a foundational encoder backbone, it has yet to be compared against SOTA VPR methods. We benchmark Adapt-STformer against CricaVPR [18] and CaseNet [14]. We also include STformer, our method’s closest architectural counterpart, as best comparable baseline. Besides conventional recall (unlimited time budget), we analyze real-world performance under inference time constraints.

We stream queries at 36 FPS—matching the reported Tesla surround-camera rate—for results in Table VI and Fig. 6. Predictions arriving after the next query are not counted under Recall@K. Table VI reports (i) conventional Recall@5, (ii) the percentage of queries processed on time (OT%), and (iii) Recall@5 over the on-time subset. Fig. 5 further evaluates performance across 20–60 FPS, covering the typical driver-assistance range [30].

TABLE VI

PERFORMANCE OF ADAPT-STFORMER AND SOTA VPR METHODS. WE COMPUTE THE RECALL@5 (R@5), RECALL@5 GIVEN ON-TIME PROCESSING OF QUERIES AT 36 FPS (R@5-36FPS), THE PERCENTAGE OF QUERIES PROCESSED ON-TIME (OT%), AND RESOURCE USAGE.

Method	Nordland		Oxford-Easy		Oxford-Hard		NuScenes		OT% \uparrow	Resource Usage		
	R@5 \uparrow	R@5-36FPS \uparrow	R@5 \uparrow	R@5-36FPS \uparrow	R@5 \uparrow	R@5-36FPS \uparrow	R@5 \uparrow	R@5-36FPS \uparrow		GFLOPs \downarrow	Mem (MB) \downarrow	Time (s) \downarrow
CricaVPR ^{pc}	0.9900	0.4953	<u>0.9803</u>	0.4891	<u>0.8276</u>	<u>0.3822</u>	<u>0.8626</u>	0.8622	68.36	138.62	517.77	0.029
CaseVPR ^{pc}	0.9850	0.3280	0.9882	0.3280	0.8492	0.2549	0.8642	0.2081	33.34	503.62	641.33	0.070
STFormer ^{pc}	0.9947	<u>0.5153</u>	0.9372	<u>0.9136</u>	0.6453	0.3000	0.5319	0.5154	<u>84.78</u>	<u>127.74</u>	<u>276.55</u>	<u>0.028</u>
Ours	<u>0.9923</u>	0.9923	0.9618	0.9618	0.7659	0.7659	0.7031	<u>0.7031</u>	100.0	102.00	180.39	0.018

We use pc for CaseVPR and CricaVPR. Both methods were trained on substantially larger datasets than ours; additionally, the CaseVPR pc was further fine-tuned on a dataset equivalent to Oxford-Easy.

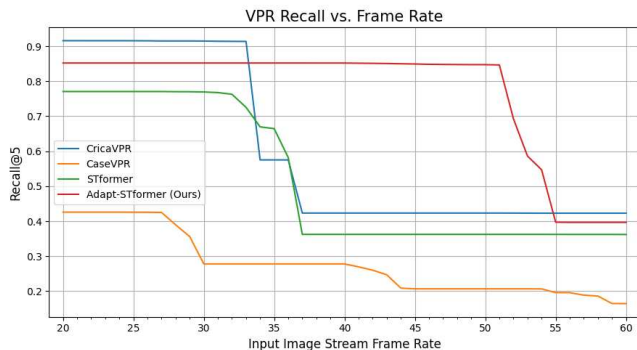


Fig. 5: **VPR Performance Under Time Constraints.** We analyze the impacts of changing inference time constraints on the performance of Adapt-STformer and other SOTA VPR methods by streaming queries from 20 FPS to 60 FPS.

Recall under Time Constraints: At 36 fps, only Adapt-STformer achieves OT%=100%, so recall remains equal to its conventional R@5 across all datasets (Table VI). Against STFormer (the next-fastest), our recall is higher by +47.7% on Nordland (0.9923 vs. 0.5153), +4.8% on Oxford-Easy, +46.6% on Oxford-Hard, and +18.8% on NuScenes (0.7031 vs. 0.5154). Although DINOv2-based CaseVPR and CricaVPR often lead in conventional R@5, their lower OT% (33.34% and 68.36%) causes large drops in recall under time constraints; an exception is NuScenes, where CricaVPR remains best at 0.8622. Overall, Adapt-STformer sustains high recall under time pressure while using the fewest GFLOPs, the least memory, and the lowest per-sequence inference latency.

Qualitative Trajectory Mappings: In Fig. 6, we plot predicted query locations on Oxford-Easy. Adapt-STformer produces the smoothest and densest trajectories, closely tracking ground truth. Here, creating visually accurate trajectory mappings requires both high recall and fast inference: methods with low recall (e.g., SeqVLAD) produce fragmented trajectories despite fast speeds, while slower methods (CricaVPR, CaseNet) omit many queries due to latency constraints, likewise degrading trajectory continuity. JIST and SeqNet are omitted as their behavior is similar to SeqVLAD but visually more fragmented.

Practical implication: When real-time constraints apply,

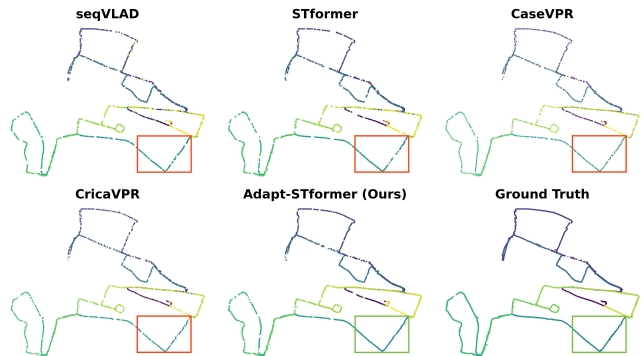


Fig. 6: Qualitative trajectories (36 FPS) mappings of various methods on Oxford-Easy, color-coded by frame index. Adapt-STformer aligns closest with GT, due to its high latency-adjusted recall (Table VI). We highlight an example region with a green box to show good alignment while red box indicates poor alignment.

a lighter backbone paired with an efficient spatio-temporal module yields better timely recognition – higher OT% and higher recall under time constraints – while also reducing memory footprint and inference time. This makes our approach a stronger choice for latency-critical VPR applications, even against more powerful alternatives.

III. CONCLUSION

Limitations & Future Works: We did not evaluate alternative encoder backbones, such as DINOv2 [15], within Adapt-STformer. Future work could explore alternative encoder backbones within our framework to preserve its compute efficient design while closing the performance gap with SOTA foundational based VPR models.

Summary: We present Adapt-STformer, a Seq-VPR framework that delivers competitive performance while remaining compute-efficient and flexible to variable sequence lengths, thanks to its core innovation—the Recurrent-DTE module—which unifies spatio-temporal modeling in a single module. We hope this work provides a foundation for future practical and powerful Seq-VPR research.

ACKNOWLEDGMENT

This work was supported by NYU IT High Performance Computing resources, services, and staff expertise. We also

thank Somik Dhar for his early contributions. Chen Feng holds concurrent appointments as an NYU Professor and as an Amazon Scholar. This paper describes work performed at NYU and is not associated with Amazon.

REFERENCES

- [1] S. Garg and M. Milford, “Seqnet: Learning descriptors for sequence-based hierarchical place recognition,” *IEEE Robot. Automat. Lett.*, vol. 6, no. 3, pp. 4305–4312, 2021. 1, 2, 4
- [2] Z. Li, W. Wang, H. Li, E. Xie, C. Sima, T. Lu, Q. Yu, and J. Dai, “Bevformer: Learning bird’s-eye-view representation from multi-camera images via spatiotemporal transformers,” in *Computer Vision – ECCV 2022*, ser. Lecture Notes in Computer Science, vol. 13669. Springer, 2022, pp. 1–18. 1, 2, 3
- [3] R. Mereu, G. Trivigno, G. Berton, C. Masone, and B. Caputo, “Learning sequential descriptors for sequence-based visual place recognition,” *IEEE Robot. Automat. Lett.*, vol. 7, no. 4, pp. 10383–10390, 2022. 1, 2, 4
- [4] G. Berton, G. Trivigno, B. Caputo, and C. Masone, “Jist: Joint image and sequence training for sequential visual place recognition,” *IEEE Robot. Automat. Lett.*, vol. 9, no. 2, pp. 1310–1317, 2024. 1, 2, 3, 4
- [5] G. Bertasius, H. Wang, and L. Torresani, “Is space-time attention all you need for video understanding?” in *Proc. 38th Int. Conf. Mach. Learn. (ICML)*, ser. Proceedings of Machine Learning Research, vol. 139, 2021, pp. 813–824. 1
- [6] C. Feichtenhofer, A. Pinz, and R. P. Wildes, “Spatiotemporal multiplier networks for video action recognition,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2017, pp. 4769–4778. 1, 2
- [7] A. Cherian, J. Wang, C. Hori, and T. K. Marks, “Spatio-temporal ranked-attention networks for video captioning,” in *2020 IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, 2020, pp. 1606–1615. 1, 2
- [8] J. Zhao, F. Zhang, Y. Cai, G. Tian, W. Mu, C. Ye, and T. Feng, “Learning sequence descriptor based on spatio-temporal attention for visual place recognition,” *IEEE Robot. Automat. Lett.*, vol. 9, no. 3, pp. 2351–2358, 2024. 1, 2, 3, 4, 5
- [9] R. M. Schmidt, “Recurrent neural networks (rnns): A gentle introduction and overview,” *arXiv preprint arXiv:1912.05911*, 2019. 1
- [10] J. M. Fácil, D. Olid, L. Montesano, and J. Civera, “Condition-invariant multi-view place recognition,” *arXiv preprint arXiv:1902.09516*, 2019. 1
- [11] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, and J. Dai, “Deformable DETR: Deformable transformers for end-to-end object detection,” in *Int. Conf. Learn. Representations (ICLR)*, 2021. 1, 2, 3
- [12] M. J. Milford and G. F. Wyeth, “Seqslam: Visual route-based navigation for sunny summer days and stormy winter nights,” in *Proc. IEEE Int. Conf. Robot. Automat. (ICRA)*, 2012, pp. 1643–1649. 2
- [13] R. Arandjelović, P. Gronat, A. Torii, T. Pajdla, and J. Sivic, “Netvlad: Cnn architecture for weakly supervised place recognition,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2016, pp. 5297–5307. 2, 4
- [14] H. Li, G. Peng, J. Zhang, M. Wen, Y. Ma, and D. Wang, “Casevpr: Correlation-aware sequential embedding for sequence-to-frame visual place recognition,” *IEEE Robot. Automat. Lett.*, vol. 10, no. 4, pp. 3430–3437, 2025. 2, 4, 5, 6
- [15] M. Oquab, T. Darcet, T. Moutakanni, H. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby, M. Assran, N. Ballas, W. Galuba, R. Howes, P.-Y. Huang, S.-W. Li, I. Misra, M. Rabbat, V. Sharma, G. Synnaeve, H. Xu, H. Jégou, J. Mairal, P. Labatut, A. Joulin, and P. Bojanowski, “Dinov2: Learning robust visual features without supervision,” *arXiv preprint arXiv:2304.07193*, 2023. 2, 7
- [16] X. Xu, Y. Jiao, S. Lu, X. Ding, R. Xiong, and Y. Wang, “Leveraging bev representation for 360-degree visual place recognition,” *arXiv preprint arXiv:2305.13814*, 2023. 2, 3
- [17] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, June 2016. 2
- [18] F. Lu, X. Lan, L. Zhang, D. Jiang, Y. Wang, and C. Yuan, “Cricavpr: Cross-image correlation-aware representation learning for visual place recognition,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, June 2024. 2, 4, 6
- [19] A. Hassani, S. Walton, N. Shah, A. Abuduweili, J. Li, and H. Shi, “Escaping the big data paradigm with compact transformers,” *arXiv preprint arXiv:2104.05704*, 2021. 2, 4
- [20] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, “An image is worth 16×16 words: Transformers for image recognition at scale,” in *Proc. 9th Int. Conf. Learn. Representations (ICLR)*, 2021. 2
- [21] N. Sünderhauf, P. Neubert, and P. Protzel, “Are we there yet? challenging seqslam on a 3000 km journey across all four seasons,” in *Proc. IEEE ICRA Workshop on Long-Term Autonomy*, 2013. 4
- [22] W. Maddern, G. Pascoe, C. Linegar, and P. Newman, “1 year, 1000km: The oxford robotcar dataset,” *Int. J. of Robotics Research*, vol. 36, no. 1, pp. 3–15, 2017. 4
- [23] W. Maddern, G. Pascoe, M. Gadd, D. Barnes, B. Yeomans, and P. Newman, “Real-time kinematic ground truth for the oxford robotcar dataset,” *arXiv preprint arXiv:2002.10152*, 2020. 4
- [24] H. Caesar, V. Bankiti, A. H. Lang, S. Vora, V. E. Liong, Q. Xu, A. Krishnan, Y. Pan, G. Baldan, and O. Beijbom, “nuscenes: A multimodal dataset for autonomous driving,” in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2020, pp. 11618–11628. 4
- [25] F. Ge, Y. Zhang, S. Shen, Y. Wang, W. Hu, and J. Gao, “Bev²pr: Bev-enhanced visual place recognition with structural cues,” in *Proc. 2024 IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, 2024, pp. 13274–13281. 4
- [26] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014. 4
- [27] H. Jégou and O. Chum, “Negative evidences and co-occurrences in image retrieval: The benefit of pca and whitening,” in *Proc. European Conf. Comput. Vis. (ECCV)*, 2012. 4
- [28] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, “Pytorch: An imperative style, high-performance deep learning library,” in *Advances in Neural Inf. Process. Syst. 32 (NeurIPS 2019)*, 2019, pp. 8024–8035. 4
- [29] C. Malone, A. Vora, T. Peynot, and M. Milford, “Dynamically modulating visual place recognition sequence length for minimum acceptable performance scenarios,” in *Proc. 2024 IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, 2024, pp. 3340–3347. 5
- [30] D. Gehrig and D. Scaramuzza, “Low-latency automotive vision with event cameras,” *Sci. Robot.*, May 2024. 6