

Supplementary Material for SOFI: Multi-Scale Deformable Transformer for Camera Calibration with Enhanced Line Queries

Sebastian Janampa Student
sebasjr1966@unm.edu

Marios Pattichis
pattichi@unm.edu

Department of Electrical & Computer
Engineering
The University of New Mexico
New Mexico, USA

1 Line Queries: Content and Geometric Information

Queries are crucial in the transformer decoder module because they interact with the enhanced feature maps to produce a task-specific output. For camera parameters, transformer-based models have two types of queries: camera parameter queries $\mathbf{q}_{\text{camera}}$ and line queries \mathbf{q}_{line} . The camera parameters queries produce three outputs: the zenith vanishing point, the field of view, and the horizon line. The line queries classify if a line passes through the zenith vanishing point or the horizon line.

We revisit the query formulation and how they are processed in the decoder in Fig. 1. CTRL-C [9] and MSCC [9] initialize $\mathbf{q}_{\text{camera}}$ as in DETR [10]. However, \mathbf{q}_{line} is initialized oppositely, as shown in Fig. 1a. This bad initialization for the line queries does not allow the line geometric information to propagate through the decoder layers. Additionally, their line query formulation does not allow the deformable attention mechanism [10] that promotes the cross-scale interaction.

First, we allow the propagation of the line geometric information as shown in Fig. 1b. We also added the line content information for a better understanding of the line. This simple and effective modification significantly boosts the performance of CTRL-C, as shown in Tab. ???. Moreover, this new query initialization allows us to adapt deformable-DETR [10] for camera calibration tasks. Notice that with the zero vector initialization from Fig. 1a the reference all the line queries would share the same reference points, while ours allows different reference points for each line query.

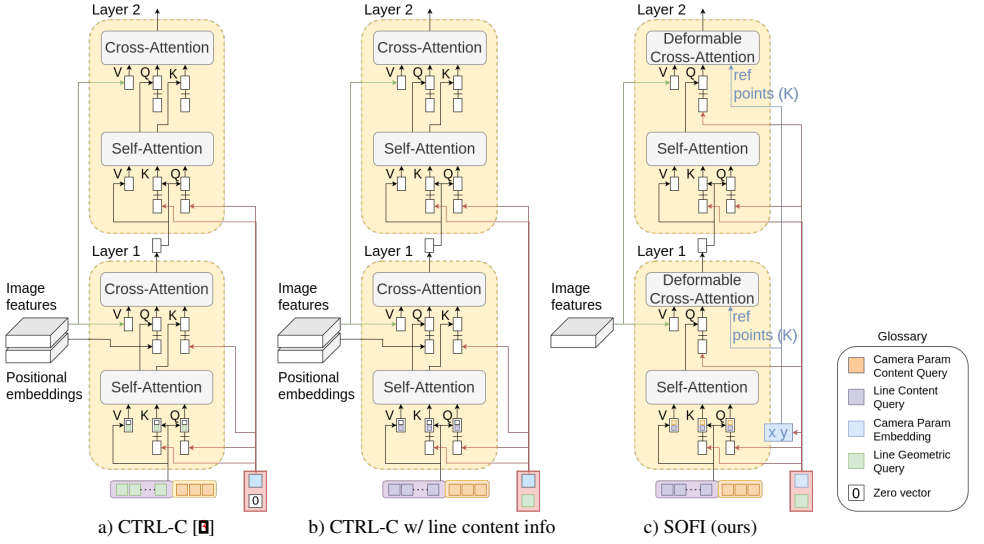


Figure 1: Decoder module. Query definition in different transformer-based model for camera calibration.

2 Loss function

2.1 Zenith Vanishing Point

For the zenith vanishing point (zvp), given a ground-truth zvp \mathbf{z} and a predicted zvp $\hat{\mathbf{z}}$, we define the loss between the two instances as:

$$\mathcal{L}_{\text{zvp}} = 1 - \left| \frac{\mathbf{z}^T \hat{\mathbf{z}}}{\|\mathbf{z}\| \|\hat{\mathbf{z}}\|} \right| \quad (1)$$

where $\|\cdot\|$ represents the euclidean norm function.

2.2 Horizon Line

Given a ground-truth horizon line \mathbf{hl} , we compute the left \mathbf{b}_l and right \mathbf{b}_r boundaries by intersecting \mathbf{hl} to the image. We repeat the same procedure with the predicted horizon line $\hat{\mathbf{hl}}$ to compute $\hat{\mathbf{b}}_l$ and $\hat{\mathbf{b}}_r$. Then, we compute the loss for the horizon line as

$$\mathcal{L}_{\text{hl}} = \max(\|\mathbf{b}_l - \hat{\mathbf{b}}_l\|_1, \|\mathbf{b}_r - \hat{\mathbf{b}}_r\|_1). \quad (2)$$

2.3 Field of View

We define the loss function as:

$$\mathcal{L}_{\text{fov}} = |f - \hat{f}| \quad (3)$$

where f and \hat{f} correspond to the ground-truth and predicted field of view.

2.4 Line Classification

Following the procedure in [9], we produce pseudo ground truth for line classification. Given a line segment \mathbf{l} and a vanishing point \mathbf{v} , the line intersects the vanishing point if $d(\mathbf{l}, \mathbf{v}) < \delta$, where $\delta = \sin(2^\circ)$, and $d(\cdot)$ is:

$$d(\mathbf{l}, \mathbf{v}) = \left| \frac{\mathbf{l}^T \mathbf{v}}{\|\mathbf{l}\| \|\mathbf{v}\|} \right|. \quad (4)$$

The Google Street dataset [10] provides a zenith vanishing point \mathbf{v}_z and two horizontal vanishing points \mathbf{v}_{hl1} and \mathbf{v}_{hl2} . We group the lines depending on whether they pass through the zenith vanishing point or the horizon line with the following conditions:

$$c(\mathbf{l}) = \begin{cases} 2 & \text{if } d(\mathbf{l}, \mathbf{v}_z) < \delta, \\ 1 & \text{if } d(\mathbf{l}, \mathbf{v}_{hl1}) < \delta \text{ or } d(\mathbf{l}, \mathbf{v}_{hl2}) < \delta, \\ 0 & \text{otherwise.} \end{cases}$$

We also group the lines based on whether they pass through any vanishing point. We mathematically describe the process as follows:

$$q(\mathbf{l}) = \begin{cases} 1 & \text{if } c(\mathbf{l}) = \{1, 2\}, \\ 0 & \text{otherwise.} \end{cases}$$

3 More Ablation Studies

3.1 Effects of the New Loss Function

CTRL-C [9] and MSCC [9] define the loss function as

$$\mathcal{L} = \mathcal{L}_{zvp} + \mathcal{L}_{hl} + \mathcal{L}_{fov} + \mathcal{L}_{verL} + \mathcal{L}_{horL} \quad (5)$$

where all losses are weighted equally. However, they do not share the same importance. We follow DETR models [9, 10, 11] and give more importance to the main task (\mathcal{L}_{zvp} , \mathcal{L}_{hl} , and \mathcal{L}_{fov}). Our new loss function is

$$\mathcal{L} = 5\mathcal{L}_{zvp} + 5\mathcal{L}_{hl} + 5\mathcal{L}_{fov} + \mathcal{L}_{verL} + \mathcal{L}_{horL}. \quad (6)$$

To validate our idea, we run CTRL-C and MSCC using the loss from equation 6 and report the results in Table 1. Training using eq. 6 allows CTRL-C to have similar or better results than MSCC. In terms of the AUC, CTRL-C[†] has slightly lower results. We believe this happens because detecting the horizon line has a stronger connection to the line classification task than the other task do.

3.2 CTRL-C with Line Content Information

We conduct a study to validate the importance of combining line geometric and line content information for better scene and line understanding. We present the results in Table 2. Adding the line content information for the Google Street View dataset [10] only slightly

Model	Up (°) ↓	Pitch (°) ↓	Roll (°) ↓	FoV (°) ↓	AUC(↑)
CTRL-C [8]	1.80	1.58	0.66	3.59	87.29
MSCC [9]	1.72	1.50	0.62	3.21	/
CTRL-C [†]	1.71	1.52	0.57	3.38	87.16

Table 1: Results on Google Street View Dataset [8]. †: means the model was trained using eq. 6.

Model	Up (°) ↓	Pitch (°) ↓	Roll (°) ↓	FoV (°) ↓	AUC(↑)
Google Street View [8]					
CTRL-C	1.80	1.58	0.66	3.59	87.29
+ eq. 6	1.71	1.52	0.57	3.38	87.16
+ line content	1.71	1.51	0.59	3.12	87.72
Holicity [9]					
CTRL-C	2.83	2.29	1.44	11.50	67.78
+ eq. 6	2.66	2.26	1.09	12.41	72.31
+ line content	2.55	2.13	1.14	11.46	77.57

Table 2: Ablation study of the different components added to CTRL-C [8].

increases the model’s accuracy. This may be due the fact that the dataset is relatively easy and the errors are already relatively low. On the other hand, we use the Holicity [9] dataset for testing where there are significant improvements after adding each component. The most noticeable improvement is the AUC of the horizon line, which increases by almost 10 points compared to CTRL-C.

References

- [1] Google street view images api. URL <https://developers.google.com/maps/>.
- [2] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European conference on computer vision*, pages 213–229. Springer, 2020.
- [3] Jinwoo Lee, Hyunsung Go, Hyunjoon Lee, Sunghyun Cho, Minhyuk Sung, and Junho Kim. Ctrl-c: Camera calibration transformer with line-classification. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 16228–16237, 2021.
- [4] Shilong Liu, Feng Li, Hao Zhang, Xiao Yang, Xianbiao Qi, Hang Su, Jun Zhu, and Lei Zhang. DAB-DETR: Dynamic anchor boxes are better queries for DETR. In *International Conference on Learning Representations*, 2022. URL <https://openreview.net/forum?id=oMI9PjOb9Jl>.
- [5] Xu Song, Hao Kang, Atsunori Moteki, Genta Suzuki, Yoshie Kobayashi, and Zhiming Tan. Mscs: Multi-scale transformers for camera calibration. In *Proceedings of*

-
- the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 3262–3271, January 2024.
- [6] Yichao Zhou, Jingwei Huang, Xili Dai, Linjie Luo, Zhili Chen, and Yi Ma. HoliCity: A city-scale data platform for learning holistic 3D structures. 2020. *arXiv:2008.03286 [cs.CV]*.
- [7] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. *arXiv preprint arXiv:2010.04159*, 2020.