EdgeVTP: Exploration of Latency-efficient Trajectory Prediction for Edge-based Embedded Vision Applications

Supplementary Material

1. Ablation Study and Operating-Point Selection

We ablate EdgeVTP along three design knobs that directly control the accuracy-latency behavior on NGSIM: (i) the interaction radius r (meters), which determines the spatial extent of neighbor interactions; (ii) a hard top-K neighbor cap, which bounds the number of edges per agent and yields predictable message-passing cost; and (iii) whether residual separation is enabled (Residual = Y/N), which increases capacity with an additional compute and memory overhead. Unless stated otherwise, all settings share the same training protocol, Transformer decoder configuration, and one-shot Bézier head, and we report both model-only and end-to-end (E2E) latency, where E2E includes graph construction and trajectory reconstruction.

Table 2 reports the complete sweep, indexed by serial ID. As expected, increasing K and r tends to increase interaction cost by admitting more neighbors, while enabling residual separation improves error rate at the cost of higher model-only and E2E runtime. Rather than selecting a single configuration based purely on ADE/FDE, we identify three representative operating points that span the practical accuracy-latency spectrum and use them consistently throughout the paper (including Results Section in main paper).

From the sweep, we select three operating points: a latency-focused setting (EdgeVTP$_{\text{Lat}}$), an accuracy-focused setting (EdgeVTP$_{\text{Error}}$), and a balanced knee-point setting (EdgeVTP$_{\text{TF}}$). We refer to the knee point as the configuration that achieves a strong accuracy gain over the latency-focused setting with only a modest additional latency, while avoiding the diminishing returns of more expensive configurations. Table 1 summarizes these selections and Table 2 highlights them in color (blue/yellow/green).

Trend summary. Across both residual and non-residual variants, K primarily controls latency by bounding the per-agent neighbor count, while r controls the interaction *scope* and can improve error rate when longer-range interactions are informative. Residual separation consistently improves error but increases model-only latency, which in turn raises E2E latency.

Table 2 provides the complete grid. From this sweep, we select three representative operating points for the main paper: (i) the lowest-latency configuration (EdgeVTP$_{\text{Lat}}$), (ii) the best error rates configuration (EdgeVTP$_{\text{Error}}$), and (iii) a knee-point trade-off configuration (EdgeVTP$_{\text{TF}}$) that balances sub-5ms inference with strong prediction quality. The selected configurations are summarized in Table 1. For convenience, the three operating points used in the main paper are summarized in Table 1. We note a performance degradation in IDs 5 and 6; we attribute this to increased social noise from distant neighbors in these specific configurations, suggesting a threshold where wider context becomes counter-productive for this architecture.

1.1. Latency Breakdown: Model Inference vs. Pipeline Overhead

A central motivation of EdgeVTP is that, on edge hardware, end-to-end latency is not dominated by learned-parameter inference alone. To quantify this, Table 2 decomposes the E2E latency into model-only inference and pipeline overhead, where the latter includes graph construction (Edge Builder) and Bézier curve evaluation for trajectory reconstruction. Across the three selected operating points, pipeline overhead accounts for 25-35% of total E2E runtime. Notably, this overhead is largely independent of whether residual separation is enabled (compare IDs 3 and 12: overhead is $\sim$1.1 ms in both cases), because it is driven by scene-dependent neighbor search and analytic curve evaluation rather than learned-parameter forward passes. When the interaction radius increases from 20 m to 40 m, the overhead grows further (e.g., ID 7: 1.63 ms, 43.9% of E2E), confirming that graph construction cost scales with the candidate neighbor set and can become the dominant latency component in larger-radius configurations. These findings support the architectural design choices in EdgeVTP: bounding interaction complexity via radius gating and a hard top-K cap directly reduces the fastest-growing component of end-to-end latency, an effect that post-hoc model compression (e.g., pruning or quantization of learned weights) would not address.

Table 1. Selected operating points from the ablation sweep (NGSIM).

Choice	ID	r	K	Residual	ADE	E2E (ms)
Best latency	3	20	16	N	2.13	3.17
Best trade-off	12	20	16	Y	1.89	4.30
Best Error	15	30	16	Y	1.85	4.58

Table 2. Ablation sweep on NGSIM (indexed by serial ID). Errors are in meters; latency is in milliseconds. Blue: best latency. Yellow: best trade-off. Green: best ADE.

ID	r	K	Residual	ADE	FDE	RMSE (m)					AVG	Params	E2E	Model-only
	(m)		(Y/N)	(m)	(m)	1s	2s	3s	4s	5s	(m)	(K)	(ms)	(ms)
1	20	8	N	3.24	6.62	1.15	2.36	3.67	5.09	6.62	3.78	134.5	3.45	2.09
2	20	12	N	2.25	5.17	0.64	1.51	2.59	3.85	5.29	2.78	134.5	3.35	2.16
3	**20**	**16**	**N**	**2.13**	**4.93**	**0.59**	**1.40**	**2.42**	**3.63**	**5.01**	**2.61**	**134.5**	**3.17**	**2.08**
4	30	8	N	2.68	5.92	0.85	1.88	3.09	4.49	6.03	3.27	134.5	3.69	2.12
5	30	12	N	2.15	4.99	2.14	4.33	6.59	8.93	11.38	6.67	134.5	3.63	2.11
6	30	16	N	4.58	5.61	1.74	3.54	5.42	7.39	9.46	5.51	134.5	3.50	2.13
7	40	8	N	2.13	4.96	0.60	1.40	2.42	3.65	5.07	2.63	134.5	3.71	2.08
8	40	12	N	2.30	5.24	0.70	1.59	2.67	3.94	5.39	2.86	134.5	3.71	2.08
9	40	16	N	2.77	6.07	0.89	1.96	3.22	4.65	6.23	3.39	134.5	3.71	2.12
10	20	8	Y	2.04	4.85	0.52	1.29	2.28	3.49	4.88	2.49	145.9	4.57	3.18
11	20	12	Y	1.89	4.37	0.56	1.27	2.16	3.25	4.54	2.36	145.9	4.33	3.14
12	**20**	**16**	**Y**	**1.89**	**4.37**	**0.56**	**1.26**	**2.15**	**3.24**	**4.52**	**2.35**	**145.9**	**4.30**	**3.20**
13	30	8	Y	1.88	4.38	0.54	1.24	2.13	3.22	4.51	2.33	145.9	4.76	3.16
14	30	12	Y	1.89	4.35	0.59	1.31	2.21	3.30	4.58	2.40	145.9	4.63	3.12
15	**30**	**16**	**Y**	**1.85**	**4.25**	**0.60**	**1.31**	**2.19**	**3.25**	**4.51**	**2.37**	**145.9**	**4.58**	**3.21**
16	40	8	Y	1.91	4.42	0.55	1.26	2.16	3.26	4.55	2.36	145.9	4.81	3.18
17	40	12	Y	1.89	4.38	0.56	1.27	2.16	3.26	4.54	2.36	145.9	4.84	3.16

Table 3. TCN-based temporal encoder variants on NGSIM. Errors are in meters; latency is in milliseconds. Missing entries are denoted by –.

ID	r	K	Residual	L	H	Smooth	ADE	FDE	RMSE (m)					AVG	Params	E2E
	(m)		(Y/N)			(Y/N)	(m)	(m)	1s	2s	3s	4s	5s	(m)	(K)	(ms)
1	35	–	N	2	2	N	1.86	4.23	0.58	1.28	2.15	3.21	4.46	2.336	145.9	–
2	35	8	Y	2	2	N	1.89	4.38	0.55	1.25	2.14	3.23	4.51	2.34	145.9	4.20
3	35	12	Y	2	2	N	1.86	4.28	0.55	1.24	2.11	3.17	4.43	2.30	145.9	3.00
4	35	16	Y	2	2	N	1.86	4.27	0.57	1.28	2.16	3.23	4.49	2.35	145.9	3.00
5	35	12	Y	8	4	N	1.84	4.20	0.56	1.23	2.08	3.12	4.34	2.27	147.9	6.30
6	35	12	Y	4	4	N	1.86	4.29	0.55	1.25	2.12	3.18	4.44	2.31	146.6	4.50
7	35	12	Y	4	2	N	1.86	4.31	0.56	1.25	2.13	3.20	4.47	2.32	146.6	4.40
8	35	12	Y	8	2	N	1.98	4.55	0.64	1.36	2.28	3.43	4.78	2.50	147.9	5.00
9	45	12	Y	8	4	Y	1.82	4.26	0.51	1.21	2.10	3.18	4.44	2.29	147.9	6.30
10	35	12	Y	8	4	Y	1.87	4.30	0.56	1.26	2.14	3.21	4.47	2.33	147.9	6.40
11	35	12	Y	2	2	Y	1.87	4.29	0.57	1.27	2.15	3.22	4.47	2.34	145.9	3.60
12	45	12	Y	2	2	Y	1.88	4.32	0.56	1.27	2.15	3.23	4.49	2.34	145.9	3.50

2. Effect of TCN Temporal Encoder

We additionally evaluated a non-causal TCN temporal encoder as a drop-in replacement for the FC temporal projection to test whether temporal convolutions improve the error-latency frontier. We keep interaction graph construction and one-shot Bézier decoding fixed, and vary decoder capacity (layers/heads), neighbor cap K, residual separation, and optional smoothing. Table 3 reports the evaluated configurations.

3. Extended Jetson Power-Mode Profiling

To further characterize deployability under realistic edge constraints, we report extended batch=1 latency across additional Jetson power modes and CPU core configurations. Table 5 profiles Jetson Xavier NX under multiple 10W/15W/20W settings with different active CPU cores, including a desktop configuration. Table 4 reports Jetson Nano under 10W and 5W modes. These measurements fol-

Table 4. Jetson Nano latency (batch=1), reported in ms (lower is better).

Model	10W	5W
STA-LSTM [1]	51.82	102.85
VT-Former$_{LH}$ [2]	1034.26	1669.51
EdgeVTP$_{\text{Lat}}$	27.87	48.46
EdgeVTP$_{\text{TF}}$	38.27	66.46
EdgeVTP$_{\text{Error}}$	37.36	65.52

Table 5. Jetson Xavier NX latency (batch=1), reported in ms across power modes and CPU core configurations (lower is better).

	10W			15W			20W		
Model	**2-core**	**4-core**	**Desktop**	**2-core**	**4-core**	**6-core**	**2-core**	**4-core**	**6-core**
STA-LSTM [1]	29.49	35.37	23.08	27.79	30.92	30.52	23.33	29.82	30.64
VT-Former$_{LH}$ [2]	416.66	506.78	328.65	340.34	429.26	425.49	334.40	433.84	400.95
EdgeVTP$_{\text{Lat}}$	14.06	16.28	10.85	11.49	13.68	13.73	11.85	14.09	15.66
EdgeVTP$_{\text{TF}}$	19.10	22.23	15.34	15.94	18.76	19.01	16.21	19.81	20.22
EdgeVTP$_{\text{Error}}$	19.07	21.99	14.89	15.88	18.81	18.90	16.23	19.44	19.69

low the same latency protocol used in the main paper.

Across all Xavier NX configurations, the EdgeVTP operating points remain consistently low-latency, while the VT-Former baseline is substantially slower (hundreds of milliseconds). The latency-focused EdgeVTP variant achieves the lowest runtime across power modes, and the balanced/error-focused variants incur only a modest increase. On Jetson Nano, EdgeVTP remains within tens of milliseconds under both 10W and 5W, whereas VT-Former requires seconds per inference.

References

[1] Lei Lin, Weizi Li, Huikun Bi, and Lingqiao Qin. Vehicle trajectory prediction using lstms with spatial-temporal attention mechanisms. *IEEE Intelligent Transportation Systems Magazine*, 2021. 2, 3

[2] Armin Danesh Pazho, Ghazal Alinezhad Noghre, Vinit Katariya, and Hamed Tabkhi. Vt-former: An exploratory study on vehicle trajectory prediction for highway surveillance through graph isomorphism and transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2024. 2, 3