SKIP: EFFICIENT TRAJECTORY PREDICTION VIA SPARSE KEYFRAME INTERACTION PROPAGATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Trajectory predictors often recompute agent and map interactions at every historical frame, even though nearby frames can share similar spatial context. We propose Sparse Keyframe Interaction Propagation (SKIP), which reduces these updates while retaining the full motion history. SKIP executes the native spatial block at selected keyframes and forms an interaction residual by subtracting the block input from its output. At each skipped frame, it adds the residual of the nearest keyframe to the same agent's own feature. The receiving frame keeps its motion state; only the interaction increment is reused. SKIP introduces no extra parameters and leaves temporal processing, the decoder, and the training objective unchanged. We analyze its approximation error and computational cost, and evaluate it on QCNet and HiVT. On Argoverse 2, reducing QCNet's spatial updates from 50 to 17 per layer increases forward throughput by 24.5% and reduces peak GPU memory by 43.0% at batch size 4, while validation minFDE changes from 1.5103 to 1.5045 m. On Argoverse 1, reducing HiVT's local interaction updates from 20 to 7 increases forward throughput by 21.8% at batch size 32, with minFDE changing from 1.0304 to 1.0264 m. On these two backbones, spatial updates can be executed less frequently without discarding motion history or increasing validation minFDE.

1 INTRODUCTION

A vehicle's motion may depend strongly on a few surrounding agents. During lane following, for example, the vehicle ahead can constrain a target's speed, while traffic across a median has weaker direct influence. As these vehicles move, the leader can remain the main constraint over several observations (Figure 1a). A predictor needs the evolving motion history, but it may gain little from recomputing similar spatial context at every frame.

We examine this intuition using relative states across 24,988 Argoverse 2 validation scenes. We normalize each component's cross-frame change variance by its pooled state variance; Figure 1b plots the largest ratio for agent and map context separately. At a one-frame separation (0.1 s), this ratio is below 1% for all measured agent and map components. The mean Jaccard similarity of the agent-neighbor sets is also 0.964. Changes grow with separation: at 0.5 s, the largest ratio reaches 8.11%. These measurements motivate testing temporal reuse at short intervals; whether learned interaction features can be reused with comparable prediction accuracy remains an empirical question.

Graph-based predictors such as HiVT and QCNet compute spatial interactions across historical frames (Zhou et al., 2022; 2023). Existing sparse interaction methods select which neighbors exchange information (Shi et al., 2021; Rowe et al., 2023). We instead investigate how often those exchanges need to be recomputed. This distinction matters for efficiency: in our QCNet diagnostic, distance pruning removes more than 72% of two major edge sets but improves throughput by only 4.3%, since graph preparation and destination-node projections still run at every frame. Omitting a spatial update can avoid these repeated operations together.

We introduce Sparse Keyframe Interaction Propagation (SKIP), which reuses spatial context but keeps each frame's own motion feature (Figure 2). At selected keyframes, including the latest observation, SKIP executes the native spatial block and subtracts its input from its output to form an interaction residual. At each skipped frame, it adds the same agent's residual from the nearest keyframe to the frame's own input, retaining the motion feature while reusing the interaction residual. SKIP introduces no extra parameters and retains the original temporal processing, decoder, and

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

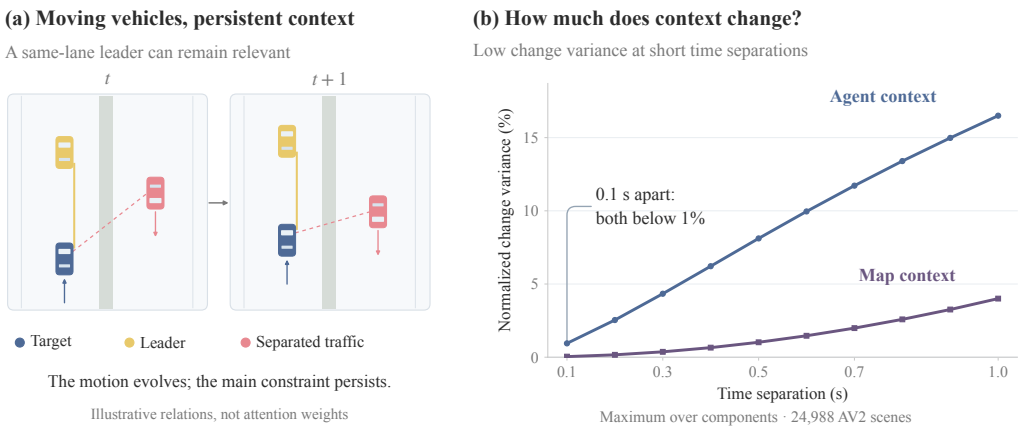

Figure 1: **From traffic intuition to measured temporal variation.** (a) A leader can remain relevant as the vehicles move; the links are illustrative. (b) Largest normalized change variance among the five agent-state components or three map-state components, measured across 24,988 AV2 validation scenes. Each component's increment variance is divided by its pooled state variance. Both curves start below 1% at 0.1 s. This measures the dispersion of changes, not their mean magnitude. Appendix A.12 defines the estimator and sampling.

training objective. Its schedule ranges from updating every frame to using the latest frame as a single interaction anchor. We analyze the resulting approximation error and computational cost.

We evaluate SKIP on QCNet with Argoverse 2 and HiVT with Argoverse 1. On the full Argoverse 2 validation set, stride-3 SKIP reduces spatial updates from 50 to 17 per QCNet layer. At batch size 4, throughput increases by 24.5%, latency falls by 19.7%, and peak allocated GPU memory falls by 43.0%, while minFDE changes from 1.5103 to 1.5045 m. On HiVT, reducing 20 local interaction updates to 7 increases batch-32 throughput by 21.8%, with minFDE changing from 1.0304 to 1.0264 m. Profiling identifies where the time is saved, and ablations separate residual reuse from sparse execution.

Our contributions are:

- A parameter-free encoder wrapper that reuses spatial interaction residuals across historical frames and keeps each frame's motion state.
- An analysis of approximation error, runtime, and peak memory as functions of the keyframe schedule, including sufficient conditions for a displacement bound and for a speedup.
- Accuracy and efficiency measurements on two backbones and both Argoverse datasets, with a QCNet diagnostic showing that skipping complete updates can save more runtime than removing a larger fraction of edges.

2 Related Work

Representations and multimodal trajectory prediction. VectorNet and LaneGCN encode vectorized trajectories and lane graphs (Gao et al., 2020; Liang et al., 2020); HiVT and QCNet combine temporal, actor–actor, and actor–map interactions (Zhou et al., 2022; 2023). Later decoders include ModeSeq, which models dependence among modes (Zhou et al., 2025); DONUT, a decoder-only autoregressive predictor (Knoche et al., 2025); FiM, which learns reward-guided intentions (Pei et al., 2025); and NDPNet, which couples near- and far-horizon heads (Hu et al., 2026). SKIP changes only the execution frequency of spatial blocks in an existing encoder; we evaluate it on QCNet and HiVT.

Efficient operators and conditional computation. Wayformer, ProphNet, ADAPT, and EMP reduce cost through attention factorization or compact architectures (Nayakanti et al., 2023; Wang et al., 2023; Aydemir et al., 2023; Prutsch et al., 2024); SmartRefine adapts decoder refinement to each scenario (Zhou et al., 2024). Trajectory Mamba uses selective state-space modeling (Huang et al.,

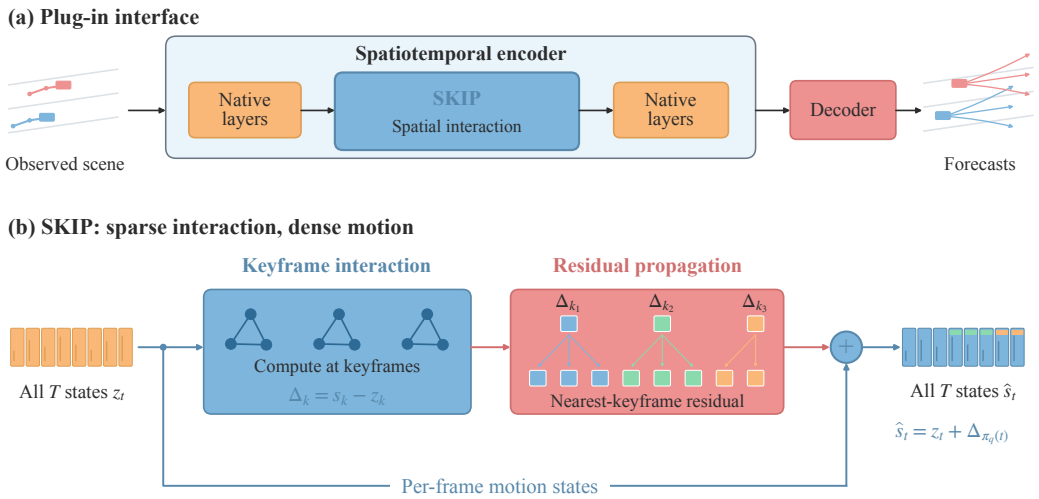

Figure 2: **SKIP computes interaction at keyframes and propagates the residual.** (a) SKIP wraps a frame-wise spatial block inside the existing encoder. (b) At a keyframe, subtract the block input from its output; add this residual to the same agent's own state at each assigned frame. One actor and one layer are shown: z_t is the block input, s_k its native keyframe output, and $\widehat{s}_t$ the SKIP output. The reference $\pi_q(t)$ is the nearest keyframe in Eq. (4). All historical states are retained.

2025), Future-Aware Interaction Network combines future-aware encoding with Mamba sequences (Li et al., 2025b), and FoSS combines Fourier- and time-domain state-space branches (Huang et al., 2026). SGCN learns sparse spatial and temporal graphs, and FJMP learns directed interaction graphs (Shi et al., 2021; Rowe et al., 2023). SkipNet, DynamicViT, and AdaFrame skip blocks, tokens, or frames in other tasks (Wang et al., 2018; Rao et al., 2021; Wu et al., 2019). SKIP keeps the native spatial operator and every historical state, and only reduces the execution frequency of that operator.

Context quality and observation robustness. Forecast-MAE and SEPT pretrain scene representations (Cheng et al., 2023; Lan et al., 2024); PerReg+ uses self-distillation, masked reconstruction, and adaptive prompting (Messaoud et al., 2025). SATP combines standard- and high-definition maps (Dong et al., 2025), and RAG-TP retrieves scene topology and motion priors (Wang et al., 2026). GALTraj generates rare scenarios (Park et al., 2025), Den-TP balances scenario density (Yang et al., 2026), and NATRA denoises pedestrian observations (Li et al., 2025a). These methods address data quality or representation learning. SKIP reduces repeated computation on the available context; we do not separately evaluate robustness to missing observations or rare scenes.

Temporal reuse and variable histories. Deep Feature Flow propagates video features using optical flow (Zhu et al., 2017); Zhao & Wildes (2019) predict temporal feature residuals. QCNet reuses features across observation windows (Zhou et al., 2023), and HPTR supports asynchronous context updates (Zhang et al., 2023). HPNet and RealMotion reuse earlier predictions (Tang et al., 2024; Song et al., 2024); ForeSight shares streaming query memory (Papais et al., 2025); SHARP carries context across short windows (Prutsch et al., 2026). PRF reconstructs missing history (Zhou et al., 2026). SKIP instead reuses spatial-block output-minus-input residuals within one observed window, retaining each receiving frame's own feature. Combining this with cross-call memory remains untested.

3 Sparse Keyframe Interaction Propagation

3.1 A shared interaction anchor for dense history

Notation. We use $j = 0, \ldots, L-1$ for encoder layers, $i = 1, \ldots, N$ for actors, and $t, u \in \{0, \ldots, T-1\}$ for historical times. A keyframe index is k, and the integer stride is $q \in \{1, \ldots, T\}$.

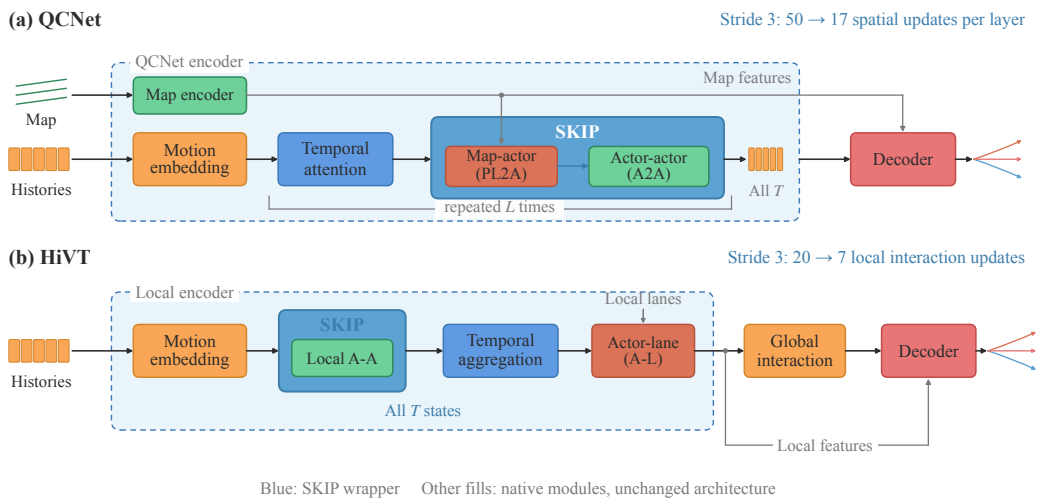

Figure 3: **SKIP insertion sites in QCNet and HiVT** (Zhou et al., 2023; 2022). In QCNet, SKIP wraps polygon-to-actor (PL2A) and actor-to-actor (A2A) interaction after temporal attention in each of L encoder layers. In HiVT, it wraps local actor–actor (A–A) interaction before temporal aggregation. Native modules keep their original computation; the blue box is the SKIP wrapper. At stride 3, the two spatial paths use 17 of 50 and 7 of 20 historical timestamps, respectively.

Layer indices always appear as subscripts. For feature representations, uppercase letters denote arrays and lowercase letters denote individual feature vectors. Thus $H_j \in \mathbb{R}^{T\times N\times d}$ is the layer input and $h_{j,i,t} \in \mathbb{R}^d$ is its actor-time vector, where d is the feature dimension. A hat marks a SKIP output.

A native layer computes

$$Z_j = F_j(H_j), \qquad S_j = \mathcal{S}_j(Z_j; \mathcal{C}), \qquad H_{j+1} = G_j(S_j). \tag{1}$$

The semicolon in $\mathcal{S}_j(Z_j; \mathcal{C})$ separates the variable features from fixed context. Here F_j and G_j contain the unchanged operations before and after spatial interaction $\mathcal{S}_j$; $\mathcal{C}$ contains fixed scene inputs, graphs, and masks. Figure 3 shows the two insertion sites. QCNet places temporal attention in F_j and wraps PL2A followed by A2A. HiVT wraps local A–A after center embedding, with temporal aggregation and A–L in G_j. In the remainder of this subsection, we fix one layer and suppress its index: $z_{i,t} = z_{j,i,t}$, $Z = Z_j$, and $S = S_j$. Appendix A.1 collects the conventions used in the analysis.

Latest-frame anchoring. The sparsest setting uses the latest observed frame $k = T - 1$ as its only interaction anchor. Evaluate the native spatial operator at k, form the actor-specific increment $b_i = s_{i,k} - z_{i,k} \in \mathbb{R}^d$, and broadcast it:

$$\widehat{s}_{i,t} = z_{i,t} + b_i = s_{i,k} + (z_{i,t} - z_{i,k}), \qquad 0 \le t < T. \tag{2}$$

For QCNet, $T = 50$ means the 50th observation (index 49) supplies one spatial update per encoder layer. All 50 historical states still undergo temporal processing. The anchor state has already received temporal context, and each layer recomputes its own increment. This is not prediction from a single observation.

For actor i, let $Z_i, \widehat{S}_i \in \mathbb{R}^{T\times d}$ stack the feature vectors as rows. With $\mathbf{1}_T \in \mathbb{R}^T$ the all-ones vector and $\top$ denoting transpose,

$$\widehat{S}_i = Z_i + \mathbf{1}_T b_i^\top, \qquad \widehat{s}_{i,t} - \widehat{s}_{i,u} = z_{i,t} - z_{i,u}. \tag{3}$$

The broadcast has rank at most one and preserves incoming temporal differences at this insertion site. The full representation need not be low rank, and later nonlinear operations need not preserve these properties.

Multiple anchors and current-state compensation. With multiple keyframes, each frame receives the residual of its nearest keyframe. For stride $1 \le q \le T$, use

$$\mathcal{K}_q = \{T - 1 - nq \mid n \in \mathbb{Z}_{\ge 0},\ T - 1 - nq \ge 0\}, \qquad \pi_q(t) = \arg\min_{k\in\mathcal{K}_q} |t - k|, \tag{4}$$

The counter n is a nonnegative integer; ties in $\pi_q(t)$ select the earlier keyframe. $q = T$ recovers the single anchor; $q = 1$ recovers dense execution. Write $\Delta_{i,t} := s_{i,t} - z_{i,t}$ for the native residual at any frame. With reference $k = \pi_q(t)$, the propagation rule is

$$\widehat{s}_{i,t} = \underbrace{s_{i,k}}_{\text{reference output}} + \underbrace{(z_{i,t} - z_{i,k})}_{\text{current-state compensation}} = z_{i,t} + \Delta_{i,k}. \tag{5}$$

SKIP gathers and adds increments without new parameters. References lie within the observed window and may be later than their receiving frames. For a valid reference, dense evaluation at the same input differs by

$$\xi_{i,t} := s_{i,t} - \widehat{s}_{i,t} = \Delta_{i,t} - \Delta_{i,k}. \tag{6}$$

Here $\xi_{i,t}$ is the local approximation error: SKIP substitutes the keyframe residual $\Delta_{i,k}$ for the frame-specific residual $\Delta_{i,t}$. When this difference is small, the spatial output is close to dense evaluation. When it is not, downstream layers may still leave the predicted trajectory essentially unchanged.

Missing observations. Equation (5) assumes a valid reference observation. In HiVT, an actor can be absent at its assigned keyframe; a skipped frame then keeps its own input without receiving a residual. Let $v_{i,k} \in \{0, 1\}$ equal one when actor i is observed at keyframe k, and zero otherwise. Collect the previously defined residuals $\Delta_{i,t}$ into an array Δ. For a fixed stride q, define the residual assigned to actor i at frame t by

$$(A_q\Delta)_{i,t} := \begin{cases} \mathbf{0}, & \text{HiVT},\ t \notin \mathcal{K}_q,\ v_{i,\pi_q(t)} = 0, \\ \Delta_{i,\pi_q(t)}, & \text{otherwise.} \end{cases} \tag{7}$$

Here $\pi_q(t)$ selects the source time, i stays fixed, and $\mathbf{0} \in \mathbb{R}^d$. Keyframes retain their native outputs; QCNet always uses the second branch. With Z collecting the inputs $z_{i,t}$ and $\widehat{S}$ the outputs $\widehat{s}_{i,t}$, the array form is

$$\widehat{S} = Z + A_q\Delta. \tag{8}$$

A_q is determined by the schedule and observation masks. Only keyframe residuals are evaluated. Appendix A.1 gives the general operator form used in the proofs.

Training and propagation controls. Training fixes the stride q and optimizes the backbone's weights with its original prediction loss; no feature-matching loss is added. The no-propagation control keeps keyframe outputs but adds no spatial residual at skipped frames. Inference ablations hold weights fixed, whereas retraining allows adaptation. Full definitions and the training objective appear in Appendices A.1 and A.4.

3.2 Anchor budget and structural cost

Update count and reference distance. For the schedule in Eq. (4), the number of spatial updates per layer and the largest distance to an assigned keyframe are

$$\begin{aligned} K &:= |\mathcal{K}_q| = \lceil T/q \rceil, \\ \rho(T, q) &:= \max_{0 \le t < T} |t - \pi_q(t)| = \max\{(T-1) \bmod q, \lfloor q/2 \rfloor\}. \end{aligned} \tag{9}$$

Proposition 1 (Minimum updates under a coverage constraint)**.** *Let $T \ge 1$ and $r \ge 0$ be integers. Any set of K keyframes that places each of the T historical frames within r steps of a keyframe must satisfy*

$$K \ge \left\lceil \frac{T}{2r+1} \right\rceil. \tag{10}$$

For $T \in \{20, 50\}$ and $r = 1$, the stride-3 schedule in Eq. (4) *attains this bound while including the latest observation.*

Thus one-frame coverage requires at least 17 updates for QCNet and 7 for HiVT. This optimum concerns reference distance, not prediction accuracy or runtime. Appendix A.1 proves the coverage formula and Proposition 1.

Saved spatial computation. Skipping a complete spatial update removes graph preparation, edge computation, and node projections together. Propagation costs $O(L(T-K)Nd)$ in addition to gathering. This explains why fewer updates can save more work than fewer edges, although scheduling overhead and unchanged computation still limit speedup. Appendix A.4 gives the full cost expression.

3.3 Approximation and prediction accuracy

Equation (6) gives the local error from residual reuse. To bound its accumulation, write $R_j(Z) = \mathcal{S}_j(Z;\mathcal{C}) - Z$ and $\eta_j(q) = \|(I - A_q)R_j(Z_j)\|_{\infty,2}$, evaluated on the dense path. Here $\|U\|_{\infty,2} = \max_{i,t} \|U_{i,t}\|_2$ includes padded slots; after temporal aggregation, the maximum is over actors. Let $e_j = \|H_j - \widehat{H}_j\|_{\infty,2}$.

Proposition 2 (Conditional representation and prediction error)**.** *Consider deterministic dense and SKIP execution with identical weights, scene inputs, masks, and initial features, so $e_0 = 0$. Assume exact arithmetic and that keyframe evaluation equals the restriction of the native spatial operator at the same input. Suppose F_j, R_j, and G_j are respectively α_j-, λ_j-, and β_j-Lipschitz in $\|\cdot\|_{\infty,2}$ on domains containing the paired inputs. With $a_j = \beta_j\alpha_j(1+\lambda_j)$,*

$$e_L \le B(q) := \sum_{j=0}^{L-1} \beta_j \eta_j(q) \prod_{\ell=j+1}^{L-1} a_\ell. \tag{11}$$

If the decoder is κ_D-Lipschitz into the maximum position norm over corresponding modes and valid future steps, minADE and minFDE, each minimized over the full fixed mode set, change by at most $\epsilon = \kappa_D B(q)$. A miss decision can change only if dense minFDE is within ϵ of the miss threshold.

For QCNet, $\eta_j(q)$ is bounded by residual variation within $\rho(T,q)$ frames; HiVT also needs a validity term. This fixed-weight bound does not establish accuracy parity after separate training. Appendix A.2 gives the proof; Appendix A.3 analyzes downstream prediction sensitivity.

4 Runtime and Memory Analysis

4.1 Latency and batch size

For fixed hardware, batch size, precision, and scene workload, let C_0 denote update-independent latency, c_s the mean cost of a spatial update, c_p the propagation cost per skipped frame, and h_q additional scheduling overhead. The additive model is

$$\tau_{\text{dense}} = C_0 + Tc_s, \tag{12}$$

$$\tau_q = C_0 + |\mathcal{K}_q|c_s + (T - |\mathcal{K}_q|)c_p + h_q. \tag{13}$$

The scheduler is beneficial under this model if and only if

$$(T - |\mathcal{K}_q|)(c_s - c_p) > h_q. \tag{14}$$

If savings grow with batch while h_q is approximately fixed, larger batches amortize scheduling overhead. We test this dependence on batch size using the same scenes at each setting.

The unchanged temporal and decoder work limits end-to-end speedup even at one anchor. Appendix A.4 gives the corresponding Amdahl expression (Amdahl, 1967).

4.2 When memory savings saturate

Peak memory is the largest allocation reached across execution phases. If the encoder high-water mark is $M_E(K) \approx m_0 + m_1 K$, $m_1 > 0$, and the other phases have an approximately fixed high-water mark M_F, then

$$\begin{aligned} M_{\text{peak}}(K) &= \max\{M_E(K), M_F\}, \\ K^*_{\text{mem}} &= (M_F - m_0)/m_1. \end{aligned} \tag{15}$$

The star in K^*_{mem} identifies the crossing of the two memory curves. An interior transition exists only if $K^*_{\text{mem}} \in [1, T]$. Below it, encoder allocation can decrease without lowering the overall peak. Each phase peak already includes all allocations live at that point; the peaks are not added together.

Choosing a schedule. Fewer keyframes reduce spatial computation, but can increase the error from residual reuse. Memory savings can also stop once another stage determines the peak. We therefore validate accuracy, runtime, and memory together when selecting a stride. Appendix A.5 combines the error and system bounds; Appendix A.8 reports the measured tradeoff.

5 Experimental Protocol

QCNet and Argoverse 2. Dense QCNet, distance pruning, and stride-3 SKIP share preprocessing, backbone, mixed-precision training, seed 2023, batch size 4, and 32 full-data epochs. We select each run's best validation-minFDE checkpoint and compute official focal-agent metrics on all 24,988 validation scenarios (Wilson et al., 2021). Efficiency uses the same hardware, data order, code path, and FP32 precision. Each main-comparison checkpoint processes all 6,247 batches in three fresh runs after warm-up. CUDA-synchronized timing measures only the model forward pass, excluding data loading, preprocessing, metric computation, and serialization. We report peak allocated GPU memory.

Additional full-data controls train at stride 3 without propagation and at stride 50 with propagation under the same 32-epoch budget. The latter has a separate single-run timing measurement. The main metrics minimize ADE and FDE independently over six modes. Auxiliary stage profiles, stride sweeps, and reduced-data seed checks use separate matched protocols (Appendix A.10); they are not pooled with the full-validation timing runs.

HiVT and Argoverse 1. On Argoverse 1 (Chang et al., 2019), dense HiVT-64, SKIP strides 2 and 3, and stride-3 SKIP without propagation are trained for 64 epochs on the complete training split with seed 2022. Accuracy uses all 39,472 validation scenes. Efficiency uses the same 6,400-scene set at batch sizes $1, 2, 4, 8, 16, 32$, 30 warm-up batches, and three fresh runs on one RTX 4090 D.

6 Results

6.1 QCNet on Argoverse 2

We compare dense QCNet, distance pruning, and stride-3 SKIP on the full Argoverse 2 validation split (Table 1). SKIP changes official minADE from 0.7331 to 0.7339, minFDE from 1.5103 to 1.5045, and MR from 0.2181 to 0.2142. Forward throughput increases from 80.05 ± 0.07 to 99.69 ± 0.07 scenarios/s, and peak allocation falls from 2510.5 to 1430.7 MB. Distance pruning changes minADE and minFDE by only a few millimetres relative to dense QCNet, but improves throughput by only 4.3% with no reduction in peak memory.

Table 1: QCNet on the full Argoverse 2 validation split. Timing is batch-4 FP32 forward over all scenarios, with mean $\pm$ sample standard deviation across three runs.

Model	minADE	minFDE	MR	Scenarios/s	Gain	Peak MB
Dense QCNet	**0.7331**	1.5103	0.2181	80.05 ± 0.07	n/a	2510.5
Distance pruning	0.7349	1.5119	0.2173	83.46 ± 0.07	$+4.3\%$	2510.5
SKIP, stride 3	0.7339	**1.5045**	**0.2142**	$\mathbf{99.69 \pm 0.07}$	$\mathbf{+24.5\%}$	**1430.7**

Across 97,424 focal and scored actors, SKIP achieves world-aligned displacement errors close to those of dense QCNet, while distance pruning achieves the lowest minADE and minFDE (Appendix A.7).

Figure 4 compares the computational savings from SKIP and distance pruning. SKIP keeps about 34% of destination tokens and of the PL2A and A2A edges, because it omits two-thirds of the spatial updates. Distance pruning keeps every destination and only 26.2% and 27.6% of those edges. Encoder time then falls from 28.88 ms/batch to 17.02 under SKIP (41.1%), compared with 3.3% for pruning, since omitted spatial updates also avoid graph preparation and destination-node computation.

On a fixed scene set, latency is close to linear in the update count,

$$\tau(K) = (8.783 \pm 0.035) + (0.07393 \pm 0.00122)K, \tag{16}$$

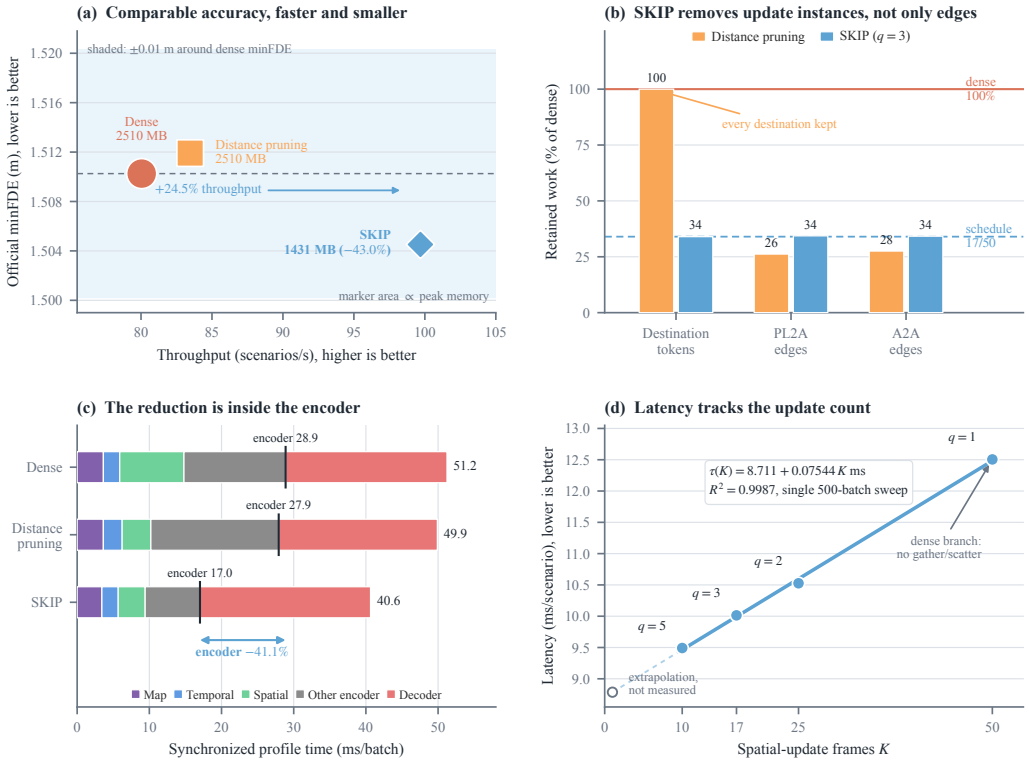

Figure 4: QCNet mechanism and runtime evidence. (a) Full-validation quality and throughput; marker area is proportional to peak allocated memory. Shading marks a ± 0.01 m reference band, not a confidence interval. (b) Distance pruning retains all destinations; SKIP retains about 34% of complete updates. (c) Encoder stage totals from a separate 200-batch profile. (d) Latency fit from a single 500-batch stride sweep; the open $K = 1$ point is extrapolated.

with $R^2 = 0.9966 \pm 0.0017$ over three runs (ms/scenario). This is consistent with the model in Section 4. The slope reflects both spatial-update and propagation costs; the $q = 1$ point uses the native dense branch. Forecast error is measured on the official split, not from this fit.

For the stride-3 checkpoint, prediction error varies nonmonotonically with execution stride: it increases at strides 1 and 2 but remains close to the training-stride result at stride 5 (Appendix A.8).

6.2 Batch size and memory

We next vary the batch size and check which phase sets peak memory. At QCNet batch 1, SKIP reduces mean latency by 7.6% and peak allocation by 63.1% on a 500-scene prefix. Distance pruning is 5.4% slower than dense QCNet, indicating that reducing the number of edges alone does not ensure a runtime improvement.

On HiVT, we evaluate batch sizes from 1 to 32. The speedup is $1.028\times$ at batch 1 (bootstrap 95% interval $[1.008, 1.047]$) and $1.218\times$ at batch 32 ($[1.203, 1.244]$). SKIP is faster at all six sizes. The larger gains at larger batch sizes are consistent with amortized scheduling overhead. These confidence intervals apply to the tested implementation and hardware.

QCNet peak memory stops falling near 26 retained updates, after which other phases set the high-water mark. HiVT is already in that regime, so the memory reduction is small. After SKIP, the QCNet decoder still takes 58.1% of the profiled forward time (Appendix A.11).

6.3 HiVT TRANSFER AND PROPAGATION ABLATIONS

Table 2 reports the same stride-3 wrapper on HiVT-64 and Argoverse 1. minFDE changes from 1.0304 to 1.0264 m, and batch-32 throughput increases by 21.8%.

Table 2: HiVT-64 transfer on the full Argoverse 1 validation split. Efficiency uses the same 6,400 scenes at batch 32 with three fresh runs. No propagation is a full retraining control.

Model	Updates	minADE	minFDE	MR	Scenarios/s	Gain	Peak MB
Dense HiVT	20	0.6861	1.0304	0.1033	1672.2 ± 5.4	n/a	629.3
SKIP, stride 3	7	**0.6838**	**1.0264**	**0.1007**	$\mathbf{2036.6 \pm 42.8}$	$\mathbf{+21.8\%}$	**623.4**
SKIP, no propagation	7	0.6868	1.0307	0.1024	n/a	n/a	n/a

Applying stride 3 to the released dense checkpoint, without retraining, raises minFDE by 4.11%. Training at stride 3 recovers baseline accuracy; the model trained at stride 2 achieves minFDE of 1.0287 m, between dense HiVT and stride-3 SKIP. Disabling residual propagation at inference on the trained SKIP model increases minFDE from 1.0264 to 1.0351. Retraining without propagation reaches 1.0307, 0.03% above dense HiVT and 0.41% above propagated SKIP. The encoder can adapt to keyframe-only interaction, but the residual still gives the lowest minFDE in this single-seed comparison.

Disabling propagation has a larger effect on QCNet: turning propagation off for the stride-3 checkpoint raises minFDE from 1.5045 to 1.8986. Retraining without propagation on the full training set reduces minFDE to 1.5375, still above 1.5045 (Appendix A.9). Over three reduced-data seeds, dense QCNet obtains 1.9679 ± 0.0413 and SKIP 1.9701 ± 0.0384; the mean paired difference (SKIP minus dense) is 0.0022 ± 0.0334 m, with individual differences changing sign across seeds.

Training from scratch at $q = 50$ for 32 epochs gives official minADE/minFDE/MR of 0.7465/1.5586/0.2255, 3.2% higher minFDE than dense QCNet, with 110.25 scenarios/s and 1433.9 MB at batch 4. The single-anchor setting provides higher throughput at the cost of increased prediction error. Fixed-weight $q = 50$ knockouts raise minFDE from 1.5175 to 1.9471 on the dense checkpoint and from 1.6408 to 2.2967 on the stride-3 checkpoint (Appendix A.9).

On 1,500 scenes evaluated with a dense-trained QCNet checkpoint, residual differences increase with temporal lag and network depth. Across scenes, final-state feature gaps correlate weakly with the absolute change in minFDE. These measurements support evaluating stride choices using prediction accuracy rather than feature similarity alone (Appendix A.6).

7 DISCUSSION AND LIMITATIONS

The keyframe budget trades computation against the effect of residual reuse on prediction. Dense temporal processing and the decoder limit speedup, and other execution phases can limit peak-memory savings.

Timing covers the forward pass on prepared batches. Full-data accuracy uses one seed per configuration, both datasets belong to the Argoverse family, and timing uses one GPU class. Scene-adaptive schedules, other accelerators, compiled kernels, and downstream planning effects remain open. Stride 3 has the most complete accuracy and efficiency evaluation. Single-anchor training incurs a 3.2% minFDE increase, and its timing has only one full-validation run.

8 CONCLUSION

SKIP reuses keyframe interaction residuals while preserving each historical frame's motion state. On QCNet, validation minFDE changes from 1.5103 m to 1.5045 m, throughput increases by 24.5%, and peak memory falls by 43.0%. On HiVT, minFDE changes from 1.0304 m to 1.0264 m and throughput increases by 21.8%. Under stated Lipschitz assumptions the analysis bounds approximation error, and the tested stride-3 schedules do not increase validation minFDE on these two backbones.

AI USE STATEMENT

Generative AI tools assisted with literature discovery, language polishing, code inspection, and checking of mathematical proofs. We designed and ran the experiments, organized the data, carried out the theoretical derivations and proofs, and set the assumptions and conditions. We take responsibility for the final content.

REPRODUCIBILITY STATEMENT

The update rule, schedule, cost model, and conditional error analysis are specified in Sections 3 and 4. Section 5 separates accuracy evaluation from timing and profiling and reports precision, batch, repetitions, and scene counts. Appendix A.1 defines the norms and assumptions used in the proof; Appendices A.6 and A.8 report residual measurements and the execution-stride scan. The supplementary archive contains the LaTeX sources, vector figures, figure-generation files, the SKIP-enabled QCNet and HiVT source files used in the experiments, analysis scripts, official evaluator outputs, and selected raw measurement records. It does not contain checkpoints, datasets, or a complete third-party dependency environment.

REFERENCES

Gene M. Amdahl. Validity of the single processor approach to achieving large scale computing capabilities. In *Proceedings of the April 18–20, 1967, Spring Joint Computer Conference*, pp. 483–485, 1967. doi: 10.1145/1465482.1465560.

Görkay Aydemir, Adil Kaan Akan, and Fatma Güney. ADAPT: Efficient multi-agent trajectory prediction with adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 8295–8305, 2023.

Ming-Fang Chang, John Lambert, Patsorn Sangkloy, Jagjeet Singh, Slawomir Bak, Andrew Hartnett, De Wang, Peter Carr, Simon Lucey, Deva Ramanan, and James Hays. Argoverse: 3D tracking and forecasting with rich maps. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8748–8757, 2019.

Jie Cheng, Xiaodong Mei, and Ming Liu. Forecast-MAE: Self-supervised pre-training for motion forecasting with masked autoencoders. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 8679–8689, 2023.

Zhiwei Dong, Ran Ding, Wei Li, Peng Zhang, Guobin Tang, and Jia Guo. Leveraging SD map to augment HD map-based trajectory prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 17219–17228, 2025.

Jiyang Gao, Chen Sun, Hang Zhao, Yi Shen, Dragomir Anguelov, Congcong Li, and Cordelia Schmid. VectorNet: Encoding HD maps and agent dynamics from vectorized representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11525–11533, 2020.

Hua Hu, Zikang Zhou, Qian Zhou, Zihao Wen, Junjie Hu, Xinhong Chen, Zhengmin Jiang, Yung-Hui Li, and Jianping Wang. Perceiving the near, reasoning the distant: Coherent long-horizon trajectory prediction for autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 24875–24884, 2026.

Yizhou Huang, Yihua Cheng, and Kezhi Wang. Trajectory Mamba: Efficient attention-Mamba forecasting model based on selective SSM. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12058–12067, 2025.

Yizhou Huang, Genze Jiang, Yihua Cheng, and Kezhi Wang. FoSS: Modeling long-range dependencies and multimodal uncertainty in trajectory prediction via Fourier-state space integration. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3750–3760, 2026.

Markus Knoche, Daan de Geus, and Bastian Leibe. DONUT: A decoder-only model for trajectory prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 28903–28912, 2025.

Zhiqian Lan, Yuxuan Jiang, Yao Mu, Chen Chen, and Shengbo Eben Li. SEPT: Towards efficient scene representation learning for motion prediction. In *International Conference on Learning Representations*, 2024.

Rongqing Li, Changsheng Li, Ruilin Lv, Yuhang Li, Yang Gao, Xiaolu Zhang, and Jun Zhou. NATRA: Noise-agnostic framework for trajectory prediction with noisy observations. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 27872–27884, 2025a.

Shijie Li, Chunyu Liu, Xun Xu, Si Yong Yeo, and Xulei Yang. Future-aware interaction network for motion forecasting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 7505–7515, 2025b.

Ming Liang, Bin Yang, Rui Hu, Yun Chen, Renjie Liao, Song Feng, and Raquel Urtasun. Learning lane graph representations for motion forecasting. In *European Conference on Computer Vision*, pp. 541–556, 2020.

Kaouther Messaoud, Matthieu Cord, and Alexandre Alahi. Towards generalizable trajectory prediction using dual-level representation learning and adaptive prompting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 27564–27574, 2025.

Nigamaa Nayakanti, Rami Al-Rfou, Aurick Zhou, Kratarth Goel, Khaled S. Refaat, and Benjamin Sapp. Wayformer: Motion forecasting via simple & efficient attention networks. In *IEEE International Conference on Robotics and Automation*, pp. 2980–2987, 2023. doi: 10.1109/ICRA48891.2023.10160609.

Sandro Papais, Letian Wang, Brian Cheong, and Steven L. Waslander. ForeSight: Multi-view streaming joint object detection and trajectory forecasting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 25474–25484, 2025.

Daehee Park, Monu Surana, Pranav Desai, Ashish Mehta, Reuben MV John, and Kuk-Jin Yoon. Generative active learning for long-tail trajectory prediction via controllable diffusion model. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 27839–27850, 2025.

Muleilan Pei, Shaoshuai Shi, Xuesong Chen, Xu Liu, and Shaojie Shen. Foresight in motion: Reinforcing trajectory prediction with reward heuristics. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 28303–28312, 2025.

Alexander Prutsch, Horst Bischof, and Horst Possegger. Efficient motion prediction: A lightweight & accurate trajectory prediction model with fast training and inference speed. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 9411–9417, 2024. doi: 10.1109/IROS58592.2024.10802425.

Alexander Prutsch, Christian Fruhwirth-Reisinger, David Schinagl, and Horst Possegger. SHARP: Short-window streaming for accurate and robust prediction in motion forecasting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 32103–32112, 2026.

Yongming Rao, Wenliang Zhao, Benlin Liu, Jiwen Lu, Jie Zhou, and Cho-Jui Hsieh. DynamicViT: Efficient vision transformers with dynamic token sparsification. In *Advances in Neural Information Processing Systems*, volume 34, pp. 13937–13949, 2021.

Luke Rowe, Martin Ethier, Eli-Henry Dykhne, and Krzysztof Czarnecki. FJMP: Factorized joint multi-agent motion prediction over learned directed acyclic interaction graphs. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13745–13755, 2023.

Liushuai Shi, Le Wang, Chengjiang Long, Sanping Zhou, Mo Zhou, Zhenxing Niu, and Gang Hua. SGCN: Sparse graph convolution network for pedestrian trajectory prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8994–9003, 2021.

Nan Song, Bozhou Zhang, Xiatian Zhu, and Li Zhang. Motion forecasting in continuous driving. In *Advances in Neural Information Processing Systems*, volume 37, pp. 78147–78168, 2024.

Xiaolong Tang, Meina Kan, Shiguang Shan, Zhilong Ji, Jinfeng Bai, and Xilin Chen. HPNet: Dynamic trajectory forecasting with historical prediction attention. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15261–15270, 2024.

Xin Wang, Fisher Yu, Zi-Yi Dou, Trevor Darrell, and Joseph E. Gonzalez. SkipNet: Learning dynamic routing in convolutional networks. In *Proceedings of the European Conference on Computer Vision*, pp. 409–424, 2018.

Xishun Wang, Tong Su, Fang Da, and Xiaodong Yang. ProphNet: Efficient agent-centric motion forecasting with anchor-informed proposals. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 21995–22003, 2023.

Ziyi Wang, Yang Zhang, Guijian Tang, Chao Zhang, Shibo Zhang, Xueqiong Li, and Shaowu Yang. RAG-TP: A general framework for vehicle trajectory prediction via retrieval-augmented generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 24865–24874, 2026.

Benjamin Wilson, William Qi, Tanmay Agarwal, John Lambert, Jagjeet Singh, Siddhesh Khandelwal, Bowen Pan, Ratnesh Kumar, Andrew Hartnett, Jhony Kaesemodel Pontes, Deva Ramanan, Peter Carr, and James Hays. Argoverse 2: Next generation datasets for self-driving perception and forecasting. In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1, 2021.

Zuxuan Wu, Caiming Xiong, Chih-Yao Ma, Richard Socher, and Larry S. Davis. AdaFrame: Adaptive frame selection for fast video recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1278–1287, 2019.

Ruining Yang, Yi Xu, Yun Fu, and Lili Su. Den-TP: A density-balanced data curation and evaluation framework for trajectory prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 10632–10641, 2026.

Zhejun Zhang, Alexander Liniger, Christos Sakaridis, Fisher Yu, and Luc Van Gool. Real-time motion prediction via heterogeneous polyline transformer with relative pose encoding. In *Advances in Neural Information Processing Systems*, volume 36, pp. 57481–57499, 2023.

He Zhao and Richard P. Wildes. Spatiotemporal feature residual propagation for action prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 7003–7012, 2019.

Hao Zhou, Lu Qi, Xiangtai Li, Jie Zhang, Yi Liu, Xu Yang, Mingyu Fan, and Fei Luo. Recover to predict: Progressive retrospective learning for variable-length trajectory prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 17863–17873, 2026.

Yang Zhou, Hao Shao, Letian Wang, Steven L. Waslander, Hongsheng Li, and Yu Liu. SmartRefine: A scenario-adaptive refinement framework for efficient motion prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 15281–15290, 2024.

Zikang Zhou, Luyao Ye, Jianping Wang, Kui Wu, and Kejie Lu. HiVT: Hierarchical vector transformer for multi-agent motion prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8823–8833, 2022.

Zikang Zhou, Jianping Wang, Yung-Hui Li, and Yu-Kai Huang. Query-centric trajectory prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 17863–17873, 2023.

Zikang Zhou, Hengjian Zhou, Haibo Hu, Zihao Wen, Jianping Wang, Yung-Hui Li, and Yu-Kai Huang. ModeSeq: Taming sparse multimodal motion prediction with sequential mode modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 1612–1621, 2025.

Xizhou Zhu, Yuwen Xiong, Jifeng Dai, Lu Yuan, and Yichen Wei. Deep feature flow for video recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2349–2358, 2017.

A Additional Analysis and Results

A.1 Schedule geometry and residual projection

Notation conventions. Table 3 lists the indices used below. Layer indices stay in subscripts in the method and the proofs; $j = L$ is the final encoder boundary, or the fully replaced mixed execution. Parentheses hold function arguments: $\eta_j(q)$ is the error at layer j for stride q. In $(A_qU)_{i,t}$ the operator acts first, then the subscript picks an actor-time vector. Upright superscripts are labels. $A_q^2 = A_q \circ A_q$ is composition and $b_i^\top$ is a transpose.

Table 3: Indices and annotations used in the method and analysis.

Notation	Meaning
$j, \ell \in \{0, \dots, L-1\}$	Encoder layer; ℓ is a second layer index in products
$i \in \{1, \dots, N\}$	Actor index
$t, u \in \{0, \dots, T-1\}$	Historical times; k is a selected historical time
$q \in \{1, \dots, T\}$	Integer execution stride
n	Nonnegative schedule counter or explicitly indexed data sample
$m \in \{1, \dots, M\}$	Prediction mode in the displacement-metric analysis
H_j, Z_j, S_j	Layer input, pre-interaction array, spatial output
$z_{j,i,t}$	A d-dimensional actor-time vector in Z_j
$\widehat{H}_j, \widehat{Z}_j$	SKIP counterparts of the dense arrays H_j, Z_j
P_q, A_q, M_q	Nearest-reference copy, gated copy, keyframe-only mask
$v_{i,t}, g_{i,t}$	Observation validity and propagation gate, each in $\{0, 1\}$
$\mathrm{np}, \mathrm{mix}$	No-propagation and mixed-execution labels
$\mathrm{src}, \mathrm{dst}$	Source-node and destination-node labels
$\mathbf{1}_T,\ \mathbf{1}\{\cdot\}$	All-ones vector and indicator function, respectively
$\|\cdot\|_{\infty,2}$	Maximum Euclidean norm over actor-time vectors
$\|\cdot\|_{\mathrm{pos}}$	Maximum Euclidean position gap over modes and future times

Validity and propagation controls. The main-text array Δ collects the spatial residuals. For the analysis, write

$$R(Z) := \mathcal{S}(Z; \mathcal{C}) - Z, \qquad \Delta = R(Z). \tag{17}$$

R is the residual map at one layer; R_j is that map at layer j. It is a function of its input; Δ is the value at Z. For an actor-time array U, extend the main-text copy by

$$(P_qU)_{i,t} := U_{i,\pi_q(t)}, \qquad (A_qU)_{i,t} := g_{i,t}U_{i,\pi_q(t)}. \tag{18}$$

The actor index i is unchanged; only time is replaced by the nearest keyframe. The gate $g_{i,t}$ recovers the two branches of Eq. (7) when $U = \Delta$. Let $v_{i,t} \in \{0, 1\}$ mark a valid observation of actor i at time t. The implementation uses

$$g_{i,t} := \begin{cases} 1, & \text{QCNet}, \\ 1, & \text{HiVT and } t \in \mathcal{K}_q, \\ v_{i,\pi_q(t)}, & \text{HiVT and } t \notin \mathcal{K}_q. \end{cases} \tag{19}$$

HiVT keeps native keyframe outputs and zeros residuals copied from invalid reference actors. The gate is fixed by the observations and the schedule; it is not learned. In $(A_qU)_{i,t}$ the operator acts on the array before an actor-time vector is selected. Array shape is preserved.

Removing propagation. Define an operator that keeps keyframe vectors and zeros all others:

$$(M_qU)_{i,t} := \begin{cases} U_{i,t}, & t \in \mathcal{K}_q, \\ \mathbf{0}, & t \notin \mathcal{K}_q, \end{cases} \qquad S^{\mathrm{np}} = Z + M_q\Delta. \tag{20}$$

Here $\mathbf{0} \in \mathbb{R}^d$ is the zero vector and np means no propagation: a skipped frame gets no spatial residual. The removed contribution is

$$\widehat{S} - S^{\mathrm{np}} = (A_q - M_q)\Delta. \tag{21}$$

At $q = T$ in QCNet this strips the shared residual b_i from every non-anchor frame. An inference ablation freezes the weights; retraining without propagation lets the model adapt.

Residual error and temporal variation. Fix weights, scene inputs, masks, and deterministic evaluation. Use $\|Z\|_{\infty,2} = \max_{i,t} \|Z_{i,t}\|_2$, including padded slots. Let $R_j(Z) = \mathcal{S}_j(Z; \mathcal{C}) - Z$ and $\Delta_j = R_j(Z_j)$ on the dense execution path; $\Delta_{j,i,t}$ is its actor-time vector. For an integer temporal radius $r \geq 0$, define

$$\eta_j(q) = \|(I - A_q)\Delta_j\|_{\infty,2}, \qquad \omega_j(r) = \max_i \max_{|t-u| \leq r} \|\Delta_{j,i,t} - \Delta_{j,i,u}\|_2. \tag{22}$$

$\eta_j(q)$ is the largest residual approximation error at layer j; $\omega_j(r)$ is the largest increment difference within r frames. For QCNet, $\eta_j(q) \leq \omega_j(\rho(T, q))$; HiVT needs an extra validity term. Under the Lipschitz assumptions of Proposition 2, the final feature gap is at most $B(q)$ in Eq. (11). For a κ_D-Lipschitz predictor, corresponding predicted positions then differ by at most $\kappa_D B(q)$. The same bound holds at $q = T$, though the radius argument is then weak. A usable single-anchor forecast does not mean the residuals were constant.

Schedule coverage. Take integers $T \geq 1$, $1 \leq q \leq T$, and zero-based frames. Write $T-1 = nq+r$ with $n \in \mathbb{Z}_{\geq 0}$ and $0 \leq r < q$. In increasing order the keyframes are $r, r+q, \ldots, r+nq = T-1$, so there are $n+1 = \lceil T/q \rceil$ of them. The opening segment has maximum source distance r; between consecutive keyframes the distance to the nearer endpoint is at most $\lfloor q/2 \rfloor$. A single keyframe means $q = T$, and the opening segment already gives $T-1$. That is Eq. (9), including $T = q = 1$. Ties go to the earlier keyframe. The radius does not depend on the tie rule.

Proof of Proposition 1. Each keyframe covers at most $2r+1$ integer frames within radius r. The union of the covered sets therefore has at most $K(2r+1)$ elements, regardless of overlap or sequence boundaries. Covering all T frames requires $T \leq K(2r+1)$, which gives Eq. (10). For $T = 20$ and $T = 50$, the stride-3 schedules contain 7 and 17 keyframes, respectively. Equation (9) gives $\rho(T, 3) = 1$ in both cases. These schedules include $T-1$ and attain the lower bound for $r = 1$.

Residual projection. For a tensor Z with aligned actor-time slots define

$$\|Z\|_{\infty,2} = \max_{i,t} \|Z_{i,t}\|_2, \qquad (P_q Z)_{i,t} = Z_{i,\pi_q(t)}. \tag{23}$$

Because a keyframe maps to itself, $P_q^2 = P_q$. The gated operator A_q is defined in Eq. (7); its gate in Eq. (19) is fixed and equals one at every keyframe. Hence

$$(A_q^2 Z)_{i,t} = g_{i,t} g_{i,\pi_q(t)} Z_{i,\pi_q(t)} = (A_q Z)_{i,t}, \qquad \|A_q Z - A_q W\|_{\infty,2} \leq \|Z - W\|_{\infty,2}. \tag{24}$$

Both operators are linear, idempotent, and nonexpansive, and generally not orthogonal projections. Fixed points of P_q are constant inside each actor's reference cell. For a fixed point of A_q, destinations with $g_{i,t} = 1$ equal their source values, whereas destinations with $g_{i,t} = 0$ are zero. SKIP is exact at a stage whenever the native increment lies in that fixed-point space, even if the incoming states have changed.

For QCNet, $A_q = P_q$. HiVT keeps the native keyframe output, but zeros an invalid keyframe actor's increment before reusing it at skipped frames. To separate reference distance from this validity rule, define

$$\chi_j(q) = \|(P_q - A_q)\Delta_j\|_{\infty,2}, \qquad \eta_j(q) \leq \omega_j(\rho(T, q)) + \chi_j(q). \tag{25}$$

Splitting $I - A_q = (I - P_q) + (P_q - A_q)$ gives the inequality. At a destination with $g_{i,t} = 0$, the reused increment is zero, so the local error is $\Delta_{j,i,t}$. A short reference distance does not control that case through the ungated temporal modulus. χ_j vanishes for QCNet and whenever the reference increments associated with destinations where $g_{i,t} = 0$ are already zero.

Under the Frobenius norm, the copy operator can have norm $\sqrt{\max_k n_k}$, where n_k is the number of destinations assigned to source k. Repeated copies do not increase the operator norm under the maximum actor-time norm used here. Padded slots are included because the implementation can propagate their features; downstream masks stay part of the fixed computation.

A.2 Representation and forecast perturbation

We prove Proposition 2 by bounding the error at each layer, then relating the final feature gap to displacement metrics.

Scope of the assumptions. Compare deterministic dense and SKIP evaluation of the same checkpoint on the same scene. Positions, map inputs, actor identities, validity masks, and learned weights are fixed. Graphs may differ across historical frames, but each retained frame must use the same spatial operator as native evaluation at that frame. That restriction holds for frame-local interaction when gathering keeps graph neighbors and relation features. The argument excludes stochastic dropout and assumes exact arithmetic; finite precision or graph-backend mismatch needs a separate numerical check. No Lipschitz assumption on the graph as a function of actor position is required across scenes.

We use the arrays H_j, Z_j and maps F_j, G_j from the main text, with the residual map R_j defined above. A hat is SKIP with the same weights and inputs; unmarked arrays are dense. For QCNet, F_j includes temporal attention; for HiVT, temporal aggregation can sit in G_j. Set

$$Z_j = F_j(H_j), \qquad \widehat{Z}_j = F_j(\widehat{H}_j), \tag{26}$$

$$H_{j+1} = G_j(Z_j + R_j(Z_j)), \qquad \widehat{H}_{j+1} = G_j(\widehat{Z}_j + A_q R_j(\widehat{Z}_j)). \tag{27}$$

The residual acts on the full actor array, neighbors included. When a native operation changes array shape, $\|\cdot\|_{\infty,2}$ is the maximum over output feature vectors; after temporal aggregation that is a maximum over actors. The Lipschitz inequalities for F_j, R_j, G_j are assumed on domains that contain the paired inputs used below. Set $e_j = \|H_j - \widehat{H}_j\|_{\infty,2}$ and $\Delta_j = R_j(Z_j)$.

One-stage decomposition. Add and subtract $A_q R_j(Z_j)$ to obtain

$$\begin{aligned} R_j(Z_j) - A_q R_j(\widehat{Z}_j) = (I - A_q) R_j(Z_j) \\ + A_q\big(R_j(Z_j) - R_j(\widehat{Z}_j)\big). \end{aligned} \tag{28}$$

The first term has norm $\eta_j(q)$, the local residual error along the dense execution path. By Eq. (24), the norm of the second term is at most $\lambda_j \|Z_j - \widehat{Z}_j\|_{\infty,2}$. Including the identity path and applying G_j gives

$$\begin{aligned} e_{j+1} &\le \beta_j\big[(1+\lambda_j)\|Z_j - \widehat{Z}_j\|_{\infty,2} + \eta_j(q)\big] \\ &\le a_j e_j + \beta_j \eta_j(q), \qquad a_j = \beta_j \alpha_j (1+\lambda_j). \end{aligned} \tag{29}$$

Induction from $e_0 = 0$ yields Eq. (11); an empty product is one. For ungated reuse each source lies within $\rho(T, q)$, so $\eta_j(q) \le \omega_j(\rho(T, q))$. With validity gating, use Eq. (25). If every local residual error vanishes, every representation gap vanishes. For $q = 1$, $A_q = I$ and the check is automatic. Exactness at a later keyframe for the same input does not make the full dense and SKIP trajectories equal: temporal mixing can carry earlier skipped-frame errors into that keyframe.

From representation to official displacement metrics. Let $Y = D(H_L)$ and $\widehat{Y} = D(\widehat{H}_L)$ contain M corresponding modes and T_f valid future steps. Here $m = 1, \dots, M$ is a mode index and $u = 1, \dots, T_\mathrm{f}$ is a future-time index. Use

$$\|Y - \widehat{Y}\|_{\mathrm{pos}} = \max_{m,u} \|Y_{m,u} - \widehat{Y}_{m,u}\|_2 \le \kappa_D e_L \le \kappa_D B(q). \tag{30}$$

An actor maximum can be added for multi-actor outputs. By the reverse triangle inequality, the displacement error at each position moves by at most $\epsilon = \kappa_D B(q)$. Averaging over valid steps keeps the bound for each mode's ADE, as does taking the last valid step for FDE. For any two real lists $(a_m)_{m=1}^M$ and $(b_m)_{m=1}^M$,

$$\Big|\min_m a_m - \min_m b_m\Big| \le \max_m |a_m - b_m|. \tag{31}$$

MinADE and minFDE stay inside the same bound even if the minimizing mode switches. The two metrics are minimized independently. The argument uses the full fixed mode set; a discontinuous top-k probability filter would need a separate argument.

For evaluation scene n, let f_n and $\widehat{f}_n$ be dense and SKIP minFDE, and let τ_{miss} be the miss threshold. Write $\epsilon_n = \kappa_{D,n} B_n(q)$ for the scene-specific displacement bound. Then

$$|\mathbf{1}\{\widehat{f}_n > \tau_{\text{miss}}\} - \mathbf{1}\{f_n > \tau_{\text{miss}}\}| \le \mathbf{1}\{|f_n - \tau_{\text{miss}}| \le \epsilon_n\}. \tag{32}$$

Consequently, for N_s scenes,

$$|\widehat{\text{MR}} - \text{MR}| \le \frac{1}{N_s} \sum_{n=1}^{N_s} \mathbf{1}\{|f_n - \tau_{\text{miss}}| \le \epsilon_n\}. \tag{33}$$

No bound on probability calibration or Brier scores follows unless predicted probabilities are controlled as well.

A.3 Effect of residual reuse on the final prediction

Changes in the predicted trajectory. Let Y_j^{mix} be the prediction with the first j layers on SKIP and the rest on dense interaction. The superscript mix labels that mixed run; the subscript $j \in \{0, \dots, L\}$ counts replaced layers. So Y_0^{mix} is dense and Y_L^{mix} is SKIP. The change from replacing layer j is

$$\delta_j(q) = \|Y_{j+1}^{\text{mix}} - Y_j^{\text{mix}}\|_{\text{pos}}. \tag{34}$$

The triangle inequality gives

$$\|Y_L^{\text{mix}} - Y_0^{\text{mix}}\|_{\text{pos}} \le \varepsilon_{\text{task}}(q), \qquad \varepsilon_{\text{task}}(q) = \sum_{j=0}^{L-1} \delta_j(q). \tag{35}$$

Here $\|Y - \widetilde{Y}\|_{\text{pos}} = \max_{m,u} \|Y_{m,u} - \widetilde{Y}_{m,u}\|_2$, with m indexing matching modes and u future steps. δ_j is a change in predicted positions; η_j is a feature reconstruction error. The next derivation relates the two through the untouched downstream network. We do not measure these response terms or assume they are small.

Both displacement bounds also bound changes in minADE and minFDE. A miss can flip only when the dense error sits within the bound of the miss threshold. Weights are held fixed. Similar validation metrics for separately trained models imply neither similar features nor matched predictions, so schedule-matched training and inference knockouts answer different questions.

Fix the checkpoint, scene, masks, and mode order. As above, Y_j^{mix} uses SKIP in layers $0, \dots, j-1$ and dense interaction in layers $j, \dots, L-1$. Comparing Y_j^{mix} and Y_{j+1}^{mix} changes only layer j: the prefix and the remaining dense layers agree.

Let Z_j^{mix} be their common input to the spatial operator. Define its dense and SKIP outputs, respectively, by

$$U_j = Z_j^{\text{mix}} + R_j(Z_j^{\text{mix}}), \qquad V_j = Z_j^{\text{mix}} + A_q R_j(Z_j^{\text{mix}}). \tag{36}$$

So $U_j - V_j = (I - A_q) R_j(Z_j^{\text{mix}})$ is the residual error at this common input. Let Φ_j map the spatial output of layer j to predicted positions through G_j, the remaining dense layers, and the decoder. Then

$$Y_j^{\text{mix}} = \Phi_j(U_j), \qquad Y_{j+1}^{\text{mix}} = \Phi_j(V_j), \qquad \delta_j(q) = \|\Phi_j(V_j) - \Phi_j(U_j)\|_{\text{pos}}. \tag{37}$$

Summing the differences gives the exact identity

$$Y_L^{\text{mix}} - Y_0^{\text{mix}} = \sum_{j=0}^{L-1} \left(Y_{j+1}^{\text{mix}} - Y_j^{\text{mix}}\right). \tag{38}$$

Taking norms proves Eq. (35). No differentiability is used. The input Z_j^{mix} comes from the preceding SKIP layers. It is not the dense-path input Z_j from the feature-error proof, so the residuals are evaluated separately.

Directional sensitivity. The real variable $\zeta \in [0,1]$ parametrizes the line segment between U_j and V_j; it is not a layer or time index. If Φ_j is continuously differentiable along this segment, then

$$\begin{aligned} \Phi_j(V_j) - \Phi_j(U_j) &= \int_0^1 J_{\Phi_j}(U_j + \zeta(V_j - U_j))\,(V_j - U_j)\,d\zeta, \\ \delta_j(q) &\le \int_0^1 \left\| J_{\Phi_j}(U_j + \zeta(V_j - U_j))\,(V_j - U_j) \right\|_{\mathrm{pos}}\,d\zeta. \end{aligned} \tag{39}$$

$J_{\Phi_j}(W)$ is the Jacobian of Φ_j at W, with arrays treated as vectors. It acts on the actual perturbation $V_j - U_j$. The same formula holds along an absolutely continuous segment when the chain rule holds almost everywhere. We do not measure these Jacobians; the formula only explains how feature error and forecast error can come apart.

For an affine downstream map $\Phi_j(W) = BW + c$, the response is exactly $\|B(V_j - U_j)\|_{\mathrm{pos}}$. A feature change in the null space of B does not move the forecast. Conversely, a temporally constant interaction increment can be reconstructed exactly by propagation, while dropping it still moves the forecast. Insensitivity to increment variation can therefore coexist with reliance on the shared increment. These are examples, not claims about QCNet's decoder.

For one fixed layer and actor, the single-anchor update $\widehat{S}_i - Z_i = \mathbf{1}_T b_i^\top$ has rank at most one. With K anchors the added matrix has rank at most $\min(K, d)$. That statement is about the added matrix at the spatial insertion site, not about the full feature array or its image under later nonlinear layers.

The metric argument applies to both $\varepsilon_{\mathrm{task}}$ and $\kappa_D B$. Close average minADE, minFDE, or MR across independently trained models does not imply a small mode-aligned displacement bound: errors can cancel across scenes, and the models can pick different trajectories. Aggregate metrics alone do not certify small feature errors or small layer responses.

A.4 Training objective and structural cost

Learning with an anchor budget. Let (X_n, Y_n), $n = 1, \ldots, N_{\mathrm{tr}}$, be training pairs: X_n holds the observed motion and scene context, Y_n the ground-truth future. With encoder $E_{\theta,q}$, decoder D_θ, and weights θ, training solves

$$\begin{aligned} &\widehat{Y}_n(\theta, q) := D_\theta(E_{\theta,q}(X_n))\,, \\ &\min_\theta \quad \frac{1}{N_{\mathrm{tr}}} \sum_{n=1}^{N_{\mathrm{tr}}} \mathcal{L}_{\mathrm{native}}\Big(\widehat{Y}_n(\theta, q), Y_n\Big)\,. \end{aligned} \tag{40}$$

N_{tr} is the number of training examples, $\widehat{Y}_n$ the decoder prediction, and $\mathcal{L}_{\mathrm{native}}$ the backbone's original prediction loss. The stride q is fixed; only θ is optimized. There is no feature-matching loss. Changing the stride only at inference freezes the weights and tests schedule transfer.

The anchor count is fixed by the stride:

$$K = |\mathcal{K}_q| = \lceil T/q \rceil\,. \tag{41}$$

Residuals and temporal states can adapt together. At $q = T$ the encoder learns with one interaction anchor per spatial block and still keeps every historical state. Without a dense-feature matching loss, independently trained dense and SKIP encoders need not have similar intermediates. Changing q only at inference instead freezes θ and tests schedule transfer. Neither intervention predicts the other.

Structural cost and speed ceiling. For an executed-frame set $\mathcal{I} \subseteq \{0, \ldots, T-1\}$, let $E_{j,t}$ count edges and $V_{j,t}^{\mathrm{src}}, V_{j,t}^{\mathrm{dst}}$ count source and destination nodes. The upright superscripts identify node roles. A conservative cost model is

$$\begin{aligned} C_{\mathrm{sp}}(\mathcal{I}) = O\Big(&\sum_{t \in \mathcal{I}} C_{\mathrm{graph}}(t) \\ &+ \sum_{j=0}^{L-1} \sum_{t \in \mathcal{I}} \left[dE_{j,t} + d^2(E_{j,t} + V_{j,t}^{\mathrm{src}} + V_{j,t}^{\mathrm{dst}}) \right] \Big). \end{aligned} \tag{42}$$

The first term includes graph and base-relation preparation performed once per executed frame. With attention width proportional to d, the second includes message aggregation, source/destination projections, and QCNet's per-layer relation projections on edges ($O(d^2E)$). Propagation costs $O(L(T-|\mathcal{K}_q|)Nd)$ in addition to gathering. Update sparsity can remove graph, edge, and node work together; edge pruning mainly reduces $E_{j,t}$. The latency analysis below adds the unchanged temporal and decoder paths.

With the fraction of dense runtime spent on spatial updates $f = Tc_s/\tau_{\text{dense}}$, update ratio $\mu_q = |\mathcal{K}_q|/T$, propagation cost ratio $\gamma = c_p/c_s$, and normalized overhead $o = h_q/\tau_{\text{dense}}$, an Amdahl-style expression (Amdahl, 1967) is

$$S(q) = \big[1 - f(1-\mu_q)(1-\gamma) + o\big]^{-1}. \tag{43}$$

Here f includes only the spatial-update work SKIP can remove. At least one update remains, so the ideal finite-history ceiling is $[1 - f(1-1/T)]^{-1}$. At $q = T$ the model gives $\tau_T = C_0 + c_s + (T-1)c_p + h_T$. Removing 98% of QCNet spatial frame evaluations is not a $50\times$ end-to-end speedup.

A.5 Joint conditions for bounded error and speedup

For scene n, let $B_n(q)$ and $\varepsilon_{\text{task},n}(q)$ be the bounds in Eqs. (11) and (35), evaluated on that scene. Let $K_q = |\mathcal{K}_q|$. Under the stated feature regularity and additive runtime assumptions, schedules satisfying

$$\mathcal{Q}_\varepsilon = \Big\{ q \in \{1,\dots,T\} : \max_n \min\{\kappa_{D,n} B_n(q), \varepsilon_{\text{task},n}(q)\} \le \varepsilon, \quad (T-K_q)(c_s - c_p) > h_q \Big\} \tag{44}$$

satisfy the per-scene displacement tolerance and reduce runtime under the model. The set can be empty. We do not decide which tested strides belong to it: the experiments measure neither worst-case Lipschitz constants nor all mixed-execution response terms. The inequalities are sufficient conditions, not an empirical certificate. Residual percentiles cannot stand in for the first condition.

A sufficient radius budget follows without assuming differentiability in time. Telescoping adjacent dense residual differences gives, for historical indices $s, t \in \{0,\dots,T-1\}$ and integer $r \ge 0$,

$$\begin{aligned} \|\Delta_{j,i,t} - \Delta_{j,i,s}\|_2 &\le \sum_{u=\min(s,t)}^{\max(s,t)-1} \|\Delta_{j,i,u+1} - \Delta_{j,i,u}\|_2, \\ \omega_j(r) &\le r\,\omega_j(1). \end{aligned} \tag{45}$$

Let $v_j = \omega_j(1)$ and use common Lipschitz constants over the candidate schedules, with $a_j \le a$. Then

$$\kappa_D B(q) \le \kappa_D \sum_{j=0}^{L-1} \beta_j \big[\rho(T,q) v_j + \chi_j(q)\big] a^{L-1-j}. \tag{46}$$

The constants v_j are worst-case adjacent differences, not the reported percentiles; they can be large at abrupt changes. For ungated reuse ($\chi_j = 0$), if the multiplier of $\rho(T,q)$ in Eq. (46) is $C > 0$, then $\rho(T,q) \le \lfloor \varepsilon/C \rfloor$ suffices for the fixed-scene tolerance. That is a reason to keep reference distance small, conditional on quantities we have not measured. Keyframe sets for different strides are generally not nested, so neither $\eta_j(q)$ nor measured accuracy must be monotone in q. The bound allows either sign of metric change. Retraining changes weights, which this frozen-weight argument does not cover.

Latency and memory implications. Subtracting the runtime models gives Eq. (14); dividing by τ_{dense} and inverting yields Eq. (43). For fixed c_s, c_p, h, write $\tau(K) = A + bK$ with $A = C_0 + Tc_p + h$ and $b = c_s - c_p$. If $A > 0$ and $b > 0$, throughput $v(K) = 1/(A+bK)$ has derivative $-b/(A+bK)^2$. Throughput rises as updates are removed; the positive intercept still limits the gain. Across strides, $K_q = \lceil T/q \rceil$ is a staircase and h_q may change, so the model is not a smooth hardware-independent speed law.

For memory, solving $m_0 + m_1 K = M_F$ gives the crossing in Eq. (15). Above it the modeled peak has slope m_1; below it the slope is zero. Each phase peak includes every live allocation at that point. Numbers from separate profiling setups cannot be added. A roughly constant total peak does not, by itself, say whether parameters, workspace, or downstream activations set the floor.

What the measurements cover. Structure profiles count the skipped spatial updates; repeated timing measures the runtime effect. Residual tables describe the approximation error; no-propagation controls isolate reuse. Full-validation metrics then score the forecast. The accuracy–efficiency claim uses all of these, not any one of them.

A.6 Residual measurements and the scope of the bound

Table 4 reports 95th percentiles on 1,500 scenes from a dense-trained QCNet checkpoint. Lag differences compare the same actor's dense residual at times t and $t + r$. Propagated-pair differences compare a skipped frame with its actual nearest source. Both endpoints must be valid. A pair is "transition" when the actor-neighbor set changes, and "stable" otherwise. The sample is a valid-pair subset of the differences that define $\eta_j(q)$; it excludes invalid-to-valid transitions and padded slots. Pooled percentiles are neither per-scene maxima nor confidence bounds.

Table 4: Residual-difference 95th percentiles in feature-space units. Transition means a changed actor-neighbor set with both endpoints valid. Layer indices are zero-based; values are not displacement errors in meters.

Pair type	Lag / stride	L0 stable	L0 transition	L1 stable	L1 transition
Lag	$r = 1$	5.14	5.09	19.10	19.54
Lag	$r = 5$	8.28	8.26	28.93	29.74
Propagated	$q = 2$	5.14	5.12	19.20	19.98
Propagated	$q = 3$	5.14	5.10	19.16	19.74
Propagated	$q = 5$	5.83	6.04	21.47	23.81

Strides 2 and 3 both have $\rho = 1$ and similar propagated-pair quantiles; stride 5 has $\rho = 4$ and larger quantiles. Layer 1 varies much more than layer 0. These measurements characterize residual reconstruction over short temporal separations, but neither determine the largest usable stride nor rule out single-anchor execution. The measured final-state gap has scene-wise correlation magnitude below 0.03 with absolute minFDE change at the tested strides. This checkpoint is also not the SKIP-trained checkpoint in the next scan, so the residual table cannot be used to predict that scan's accuracy.

The feature-error bound is conservative. The last observed step is always a keyframe, so its layer-0 dense-SKIP gap is nearly zero. At the next layer the gap is large, yet its scene-wise correlation with absolute minFDE change still has magnitude below 0.03 for strides 2, 3, and 5. Intermediate feature distance is a poor error predictor on its own. The response decomposition separates that gap from downstream forecast sensitivity; neither low sensitivity nor single-anchor training quality follows from the correlation.

A.7 Complementary world-aligned evaluation

Table 5 reports official world-aligned metrics over 97,424 focal and scored AV2 actors. Stride-3 SKIP stays close to dense QCNet; the added single-anchor and no-propagation training controls have higher errors. The same evaluator is used throughout.

Table 5: World-aligned metrics over focal and scored Argoverse 2 actors.

Model	minADE	minFDE	MR	B-minFDE
Dense	0.7773	1.8289	0.2562	2.4217
Distance pruning	**0.7699**	**1.8058**	0.2572	**2.3958**
SKIP, stride 3	0.7762	1.8135	**0.2540**	2.3973
SKIP, stride 3, no propagation	0.8012	1.8830	0.2646	2.4732
SKIP, stride 50	0.7901	1.8620	0.2603	2.4548

A.8 POST-TRAINING EXECUTION SCHEDULES

Changing the execution stride after training measures schedule mismatch. The stride-3 checkpoint is worse at strides 1 and 2; stride 5 stays close and is faster. The accuracy trend is not monotone. Figure 5 plots full-validation minFDE and three-run throughput, including the completed measurements at strides 4 and 8. Stride 3 is the coarsest one-frame-coverage schedule on both histories, not a claimed global optimum. The main comparison matches training and evaluation schedules.

We first freeze the stride-3 QCNet checkpoint and vary only its execution stride. Figure 5 combines two separate evaluations of this checkpoint: full-validation accuracy at $q \in \{1, 2, 3, 4, 5, 8\}$, and a confirmatory batch-4 FP32 timing scan on the same 6,400 scenes at $q \in \{1, 2, 3, 4, 5, 8\}$ with three fresh process runs. Table 6 lists coverage and the measured values. The stride-limit evaluation supplies the added stride-4 and stride-8 minFDE/MR values; its legacy ADE field is excluded. Accuracy has one frozen-checkpoint evaluation per reported stride; timing error bars are sample standard deviations and do not measure training-seed uncertainty.

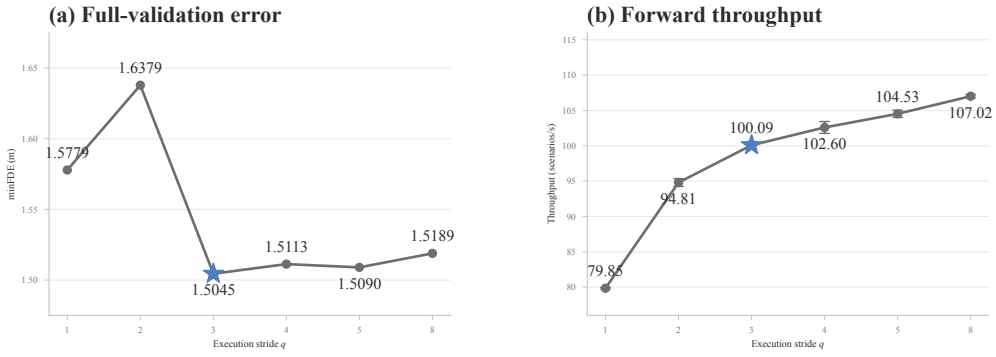

Figure 5: Execution-stride scan with one QCNet checkpoint trained at $q = 3$. (a) minFDE on all 24,988 AV2 validation scenes, including stride-4 and stride-8 measurements. (b) Forward throughput on a fixed 6,400-scene set, batch 4, FP32, mean $\pm$ sample standard deviation over three runs. Lines connect measured settings as visual guides. The starred setting matches training. Dense execution at $q = 1$ uses this same SKIP-trained checkpoint, not the separately trained baseline.

Table 6: Fixed stride-3 QCNet checkpoint. Accuracy uses full validation; throughput uses three runs on 6,400 fixed scenes.

Eval. q	Updates K	Radius ρ	minFDE	MR	Scenarios/s
1	50	0	1.5779	0.2301	79.85 ± 0.13
2	25	1	1.6379	0.2412	94.81 ± 0.56
3*	17	1	1.5045	0.2142	100.09 ± 0.51
4	13	2	1.5113	0.2167	102.60 ± 0.84
5	10	4	1.5090	0.2141	104.53 ± 0.52
8	7	4	1.5189	0.2177	107.02 ± 0.23

*Training-matched execution. Timing uncertainty is sample SD, not a training-seed interval.

Measured throughput rises from 79.85 scenarios/s at stride 1 to 107.02 at stride 8, but quality is not ordered by that gain. Restoring dense execution gives minFDE 1.5779, against 1.5045 at the training stride. Stride 2 is worse still, 1.6379. Changing the execution schedule changes the computation applied to the learned weights; extra interaction evaluations do not undo the mismatch. The pattern is consistent with adaptation to the training schedule. It does not identify the internal cause.

Strides 4, 5, and 8 give minFDE 1.5113, 1.5090, and 1.5189. Stride 5 gives 104.53 scenarios/s, against 100.09 at stride 3, so it is another usable operating point. A deployment stride still needs an explicit error tolerance and a check on held-out data. These are execution changes, not six separately retrained models.

Table 7 adds the complementary dense-trained checkpoint at strides 2 and 3. Its single-run 2,000-scene throughput values use a different timing harness and are not averaged with Figure 5. The main baseline comparison uses full-validation timing and stays separate from both scans.

Table 7: Complementary dense-trained QCNet transfer. Accuracy uses full validation; throughput uses a separate single-run 500-batch harness.

Training	Eval. q	minFDE	MR	scen./s
Dense	2	1.5111	0.2175	94.66
Dense	3	1.5108	0.2186	99.50

A.9 Single-anchor training and propagation controls

Full-data training. Table 8 adds two completed runs to the 32-epoch, seed-2023 comparison: stride 3 without propagation (best epoch 30) and stride 50 with propagation (best epoch 31). Each run starts from scratch. The latter uses only the latest observation for spatial updates and still keeps all 50 historical motion states. The official evaluator gives minFDE 1.5586, 0.0483 m (3.2%) above dense QCNet; MR rises by 0.0074. Full-data stride-3 training without propagation gives 1.5375 minFDE, 0.0330 m above propagated SKIP. At stride 3, the minFDE increase without propagation is smaller after retraining than when propagation is disabled only at inference.

Table 8: Full-data QCNet training controls: 32 epochs, seed 2023, best validation-minFDE checkpoint. Official focal-agent metrics cover all 24,988 AV2 validation scenes. Training and evaluation use the same stride and propagation setting.

Configuration	q	Prop.	minADE	minFDE	MR	B-minFDE
Dense	1	native	0.7331	1.5103	0.2181	2.1148
SKIP	3	on	0.7339	1.5045	0.2142	2.0965
SKIP, no propagation	3	off	0.7478	1.5375	0.2257	2.1461
SKIP, single anchor	50	on	0.7465	1.5586	0.2255	2.1708

A separate batch-4 forward benchmark of the stride-50 checkpoint covers all 6,247 validation batches: mean 9.0702 ms/scenario, 110.25 scenarios/s, and 1433.9 MB peak allocated memory. Batch-latency median and 99th percentile are 35.57 and 54.61 ms. This is one synchronized timing run on the FP32 benchmark path; it is not pooled with the three-run main comparison or the CUDA-event stride scan. Allocated memory includes the benchmark's model loading and warm-up. One run gives no estimate of timing variability and is not used to update the latency fit.

Fixed-weight inference interventions. We evaluate the full-data dense checkpoint (epoch 30) and the full-data stride-3 checkpoint (epoch 31) at execution stride 50, keeping all motion states. For each checkpoint, propagation is turned on and off at inference with no extra training. Each evaluation uses all 6,247 validation batches at batch size 4, covering 24,988 AV2 scenarios. Reference evaluations use each checkpoint's native execution stride through the same evaluator. These measurements use the legacy ADE convention, so Table 9 reports only minFDE and MR, whose definitions match the main evaluation. They are not results from a model trained at stride 50.

At stride 50, turning propagation off raises minFDE by 28.3% and 40.0% for the dense and stride-3 checkpoints. With propagation on, the dense checkpoint's minFDE is 0.48% above its native-stride result. Reliance on propagation is checkpoint-dependent. These numbers are not retraining results and have no confidence intervals. The dense checkpoint run at stride 50 also has lower minFDE than the stride-50-trained model, so matching training and execution schedules does not guarantee the best quality.

Table 9: Single-anchor inference intervention on two fixed QCNet checkpoints. q_{train} and q_{eval} distinguish training and execution. All rows retain the full motion history.

Checkpoint	q_{train}	q_{eval}	Propagation	minFDE	MR
Dense	1	1	native	1.5103	0.2181
Dense	1	50	on	1.5175	0.2194
Dense	1	50	off	1.9471	0.3390
SKIP	3	3	on	1.5045	0.2142
SKIP	3	50	on	1.6408	0.2388
SKIP	3	50	off	2.2967	0.3645

Reduced-data retraining. Table 10 reports completed 16-epoch runs with 10% training data, 40% validation data, and seed 2023. Each configuration keeps all 50 motion states. At stride 50, training without propagation gives minFDE 2.0768 and MR 0.3115, against 1.9870 and 0.3041 with propagation. Propagation lowers minFDE by 0.0898 m under this matched budget. Relative to the same-seed dense run (2.0004), propagated stride 50 differs by -0.0134 m; one seed does not establish parity. Strides 8, 12, and 25 further show that retrained accuracy need not be monotone in stride. Only completed runs are included; each large-stride configuration has one completed seed.

Table 10: Completed reduced-data QCNet retraining: 10% training data, 40% validation data, 16 epochs, seed 2023. Each row uses its best validation-minFDE checkpoint. These repository-evaluator metrics are separate from the official full-validation results.

Train/eval. q	Updates K	Propagation	minFDE	MR
1	50	on	2.0004	0.3090
2	25	on	1.9250	0.2974
3	17	on	1.9652	0.3039
3	17	off	2.0445	0.3167
5	10	on	2.0123	0.3089
8	7	on	2.0444	0.3096
12	5	on	1.9218	0.2970
25	2	on	1.9553	0.2984
50	1	on	1.9870	0.3041
50	1	off	2.0768	0.3115

A.10 Auxiliary evaluation protocols

Checkpoints are selected by validation minFDE, and reported accuracy is measured on the validation split rather than a held-out test split. For HiVT, training and accuracy evaluation use the serial path; timing compares the parallel implementations of both models. These paths are reported separately.

The original stride sweep measures 12.505, 10.526, 10.014, and 9.492 ms/scenario at $K = 50, 25, 17, 10$. The confirmatory three-run sweep adds $K = 13$ and 7 on a fixed scene set and supports the latency fit in the main text.

The main evaluator minimizes ADE and FDE independently over six output modes. We also compute world-aligned metrics over 97,424 focal and scored actors. Earlier development logs used the ADE of the FDE-selected mode under a minADE label, so those fields are not mixed with official QCNet minADE. Legacy controls contribute minFDE and MR only, whose definitions agree. Accuracy, timing repetitions, structure profiling, and training-seed pairs are kept as separate evidence.

Structure and stage profiles use a separate matched 200-batch harness. The main-text latency plot uses one 500-batch run. A confirmatory sweep evaluates $q \in \{1, 2, 3, 4, 5, 8\}$ on the same 6,400 scenes in three fresh runs. A reduced-data check trains matched dense and SKIP models for 16 epochs with seeds 2023, 1, and 42. Frozen-checkpoint inference noise is kept separate from training-seed variation.

A.11 MEMORY PHASES AND THE REMAINING BOTTLENECK

QCNet follows the phase model in Eq. (15). Two measurements of encoder memory give $M_E(K) = 129.3 + 38.57K$ MB. The measured allocation level in a subsequent execution phase intersects this model near $K = 25.9$. A separate stride harness reaches 2053.6 MB at $K = 50$ and about 1125 MB at $K = 25, 17, 10$. Cutting updates further does not move the running peak. HiVT is already in that fixed-allocation regime: its batch-32 peak changes from 629.3 to 623.4 MB, although live allocation after the decoder falls from 23.6 to 20.8 MB. Lowering update frequency can still improve runtime when peak memory changes little.

The profiler also locates the next bottleneck. Dense QCNet spends 56.4% of its instrumented encoder-plus-decoder time in the encoder. After SKIP the encoder share drops and the untouched decoder occupies 58.1% of the measured forward. SKIP is an encoder optimization with a predictable Amdahl limit. Further speedups have to cut the remaining costs, including the decoder.

A.12 RELATIVE-STATE STATISTICS FOR THE MOTIVATION

Sampling. Figure 1b uses all 24,988 AV2 validation scenes, one focal target per scene, and 50 historical observations at 10 Hz. We keep agents and map polygons within 50 m at source time t. Agent increments need valid target and neighbor observations at both endpoints; the neighbor may leave the radius. Map increments need a valid target at both endpoints. Eligible target–neighbor–time samples are pooled, so scenes with more neighbors weigh more.

Estimator. For a target–neighbor pair, r_t denotes its relative state in the target's heading-aligned frame at time t, with the pair index suppressed. Agent components are longitudinal and lateral relative position, longitudinal and lateral relative velocity, and relative heading. Map components are relative position and orientation. Let $u \in \{1, \dots, 10\}$ be the separation in frames and c index a state component. Define

$$\delta r_t(u) = r_{t+u} - r_t, \qquad \widehat{\Sigma}(u) = \mathrm{Cov}_{\mathrm{emp}}[\delta r_t(u)], \qquad \nu_c(u) = \frac{\widehat{\Sigma}_{cc}(u)}{\mathrm{Var}_{\mathrm{emp}}[r_{c,t}]}. \tag{47}$$

Angular increments are wrapped to $[-\pi, \pi)$. The diagonal entry $\widehat{\Sigma}_{cc}(u)$ is the variance of changes in component c. Its denominator pools valid in-radius states across historical times. Empirical variances divide by the sample count. Figure 1b plots $100 \max_c \nu_c(u)$ separately over the five agent components and the three map components. Each curve is a component maximum, not an average or a confidence interval. These are raw-input statistics. The numerator measures dispersion around the mean increment, not its magnitude. Pooled state variance includes differences between scenes and neighbors, so a small ratio does not establish within-track correlation, learned-residual smoothness, or prediction accuracy.

The plotting coordinates distributed with the manuscript were recovered from an earlier vector figure. They reproduce the plotted curves and are not the underlying per-scene measurements.

Neighbor persistence. For valid agents within 50 m, write the neighbor set as $\mathcal{N}_t$. Its Jaccard similarity is

$$J_t(u) = \frac{|\mathcal{N}_t \cap \mathcal{N}_{t+u}|}{|\mathcal{N}_t \cup \mathcal{N}_{t+u}|}, \tag{48}$$

where vertical bars denote set cardinality. Averaging over target-time samples with valid target endpoints and a nonempty union gives 0.964 at $u = 1$ and 0.852 at $u = 5$. Learned-residual measurements and propagation ablations are separate evidence about SKIP itself.