Dense Flow from Adaptive Correspondence

Cheng-Lin Hsieh **Stella X. Yu**
University of Michigan
{clhsieh,stellayu}@umich.edu

Abstract

Dense optical flow estimates where every pixel moves between two images. It must be precise and robust. Both properties depend on two choices: which units to match, and what evidence to match them with. Small units localize precisely but carry weak evidence. Large units carry richer evidence but can mix motions across boundaries. Recent work has strengthened the evidence with better features, cost volumes, attention, and generative priors. The unit, however, is still a fixed grid point. We propose DACE: *Dense flow from Adaptive CorrespondencE*, which learns the units while keeping the evidence dense. A differentiable assignment groups pixels into tokens, builds their descriptors from dense image features, and maps token motion back to pixels as dense flow. Tokens can be large where motion is coherent and small near boundaries and fine detail. Grouping and matching are learned jointly from flow supervision: grouping makes matching easier, and matching shapes grouping. DACE achieves the lowest zero-shot Sintel Final and KITTI endpoint errors among published methods, with fewer MACs than most. Controlled comparisons with fixed grids show that the gain comes from the adaptive units, which make DACE both more precise near motion boundaries and more robust where no match is visible, such as at occluded or out-of-frame pixels.



Input images Learned tokens Predicted dense flow

Figure 1: **Dense flow from adaptive correspondence.** DACE replaces fixed grid points with learned matching units. White contours show how pixels are grouped into tokens: groups follow coherent regions and stay sharp near boundaries. Token motion is mapped back to pixels, so the predicted flow keeps sharp boundaries and thin structures. Color shows flow direction and magnitude.

1 Introduction

Visual correspondence asks which elements in different images refer to the same scene content. It underlies optical flow [1, 2], point tracking [3], and multimodal alignment [4]. The shared question is how to make correspondence both precise and robust across views and appearance change. We study this question through two-frame optical flow, which estimates where every pixel moves in the

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

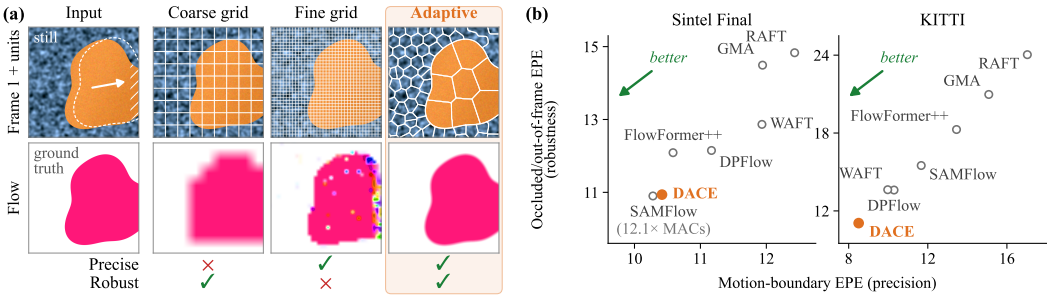


Figure 2: **Adaptive correspondence combines precision and robustness.** (a) Why adaptive units help, in an idealized example where only the matching units differ. The striped part of the object has no visible match. (b) Zero-shot results of competitive methods with public checkpoints trained on FlyingChairs and FlyingThings3D, the same training data as DACE.

next frame. Flow is widely used, from action recognition [5] and autonomous driving [6] to action prediction in video world models [7]. It is also a cleaner testbed than long-range point tracking [8–10], which often relies on semantics. For flow, precision means localizing motion exactly at boundaries and thin structures. Robustness means estimating motion reliably where evidence is weak or missing, such as at textureless, occluded, or out-of-frame pixels.

Precision and robustness are hard to achieve together, because both depend on two coupled choices: which units to match, and what evidence to match them with. Small units, such as keypoints or grid cells, support precise localization, but often carry weak evidence. Large units, such as image regions or object parts, carry richer appearance, shape, and motion context. However, their boundaries are harder to delineate, and their interior may not move as one. *What helps localization may hurt matching; what helps matching may hurt localization.*

Progress in optical flow has mostly strengthened the evidence. PWC-Net uses pyramidal cost volumes to handle large displacements [11], and RAFT builds all-pairs correlations and recurrently refines a flow field on a grid [12]. Transformer-based methods improve global matching and context aggregation [13, 14]. WAFT achieves high accuracy by iteratively warping features, without explicit cost volumes [15]. Diffusion-based methods add learned priors for ambiguous regions: some treat flow estimation as denoising [16, 17], and others extract correspondence features from pretrained diffusion models [18, 19]. Yet the unit of correspondence has stayed the same: points on a regular grid. The grid is a natural choice, because the desired output is itself a dense, pixel-accurate flow field. A grid point may attend globally, be refined recurrently, or be guided by a generative prior, but it is still the unit whose motion is estimated. The question is not whether a grid point has enough context, but whether it should be the fixed unit of correspondence in the first place.

This fixed-unit recipe is so familiar that it is easy to miss how odd it is. The goal is correspondence everywhere, yet motion is estimated only at fixed points. A single point is ambiguous in textureless regions and suffers from the aperture problem along edges. A larger piece can be more informative, as long as it does not span two motions. A sleeve or a car body may contain no single distinctive point, yet each moves as a coherent whole. Once such a piece is matched, assigning motion inside it becomes easier, even for parts that become occluded or leave the frame. Such a piece should also transfer better to new datasets. A change in texture or lighting can mislead a single point, but a whole piece is matched using evidence pooled over many pixels.

Our answer is that the matching unit should adapt. Dense output need not rely on dense grid points. Sometimes the right unit is local and point-like, anchored by a clear visual cue. Sometimes it is broader and irregular, held together by common motion, as in the Gestalt principle of common fate [20, 21]. We illustrate why adaptive units help in Figure 2a, with an idealized example where only the units differ. Large grid cells pool enough evidence to estimate the motion of a faintly textured object, even where it has no visible match. But a cell straddling its boundary carries one motion for both object and background, so the boundary blurs. Small grid cells keep the boundary but err where the texture is faint and fail where no match is visible. Adaptive units, as many as the large cells, follow the object, so they keep the boundary and still match reliably.

In practice, the object is not given. Appearance alone is an unreliable guide to common motion: a white car can look like the bright road it drives on. Motion would be the right cue, but it is known only after matching. The unit should thus come from a learned prior of what tends to move together. Pixels should be grouped sharply at boundaries but can be shared among units inside coherent regions.

We propose DACE: *Dense flow from Adaptive CorrespondencE*, which learns its matching units together with their motion while keeping the evidence dense (Figure 1). DACE groups pixels into tokens, builds their descriptors from dense image features, matches the tokens between frames, and maps token motion back to pixels as dense flow. Grouping and matching share features and are learned jointly from flow supervision, so common fate works in both directions. A good grouping makes motion easier to estimate, and learning to estimate motion teaches the model which pixels should share a token and which should stay separate.

Following the standard protocol, we train DACE on the synthetic FlyingChairs and FlyingThings3D datasets [22, 23] and evaluate zero-shot on Sintel and KITTI [24, 25]. Zero-shot transfer matters because dense motion labels are scarce in real scenes. DACE achieves the lowest Sintel Final and KITTI endpoint errors among published methods, with fewer MACs than most. Figure 2b compares precision near motion boundaries and robustness on occluded and out-of-frame pixels with competitive prior methods trained on the same data: DACE has the lowest errors in both regions on KITTI, and is close to the far more expensive SAMFlow on Sintel Final.

Our main contributions are:

- We show that dense optical flow need not be estimated at fixed grid points: its matching units can adapt to what tends to move together, while the image evidence stays dense.
- We introduce a differentiable pixel-to-token assignment that forms tokens from dense features and maps their motion back to pixels, so grouping and matching are learned jointly.
- We achieve the best zero-shot accuracy on Sintel Final and KITTI among published methods. Controlled comparisons with fixed grids, including one with four times as many units, show that the gain comes from the adaptive units, which improve precision and robustness.

2 Related Work

Correspondence evidence on fixed grids. Deep optical flow has steadily strengthened matching evidence. FlowNet2 stacks warping networks [26], and coarse-to-fine pyramids combine evidence across scales [11, 27]. RAFT [12] and its recurrent variants [28] refine flow with correlation lookups, and DPFlow adapts its pyramid to the input resolution [29]. Transformers support global matching [13] and cost-volume encoding [14, 30]. Other methods use geometrically pretrained features [31], diffusion priors [16, 17], or warping without cost volumes [15]. GMA [32] targets occluded and out-of-frame pixels by propagating motion among grid points with similar context features. This acts as an implicit grouping, learned only from flow supervision. Yet in all of these methods, the units whose motion is estimated are grid points, each tied to the same block of pixels no matter what that block contains. DACE instead groups pixels into tokens according to what the image contains, and estimates motion for each token. A token can thus follow a coherent region and end at a boundary.

Learned superpixels. Superpixels partition an image into compact regions that follow image boundaries [33, 34]. SSN [35] and SpixelFCN [36] make this partition differentiable by softly assigning each pixel to nearby superpixels, so per-pixel features or labels can be pooled into regions and mapped back to pixels. Their goal is a good partition of a single image. They can also be trained to preserve labels such as semantic classes or ground-truth flow, but they do not estimate flow. When SpixelFCN is applied to stereo, superpixels downsample the input images, so matching starts from region-averaged colors. DACE adopts the same local soft assignment for a different purpose. Its tokens, with descriptors built from learned features, are the units matched between frames. Their grouping is learned jointly with the matching, without an appearance objective.

High-resolution evidence and correspondence units. High-resolution features resolve fine structure and motion boundaries. WAFT [15] shows that matching at higher resolution improves flow accuracy. Halving the grid spacing, however, quadruples the number of units whose motion must be estimated and refined. We argue that the benefit comes largely from resolving boundaries, which

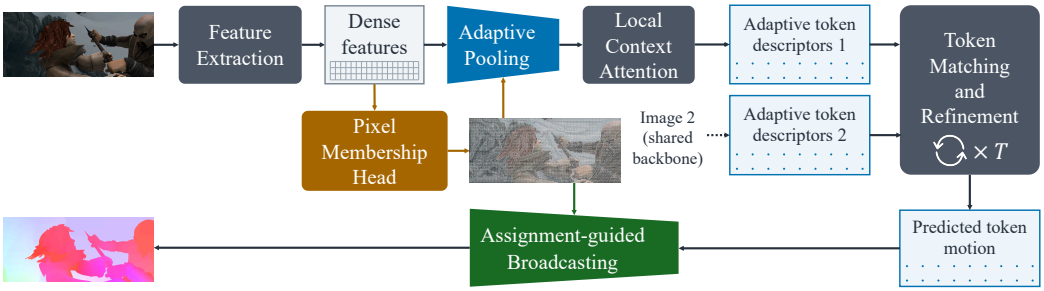


Figure 3: **DACE groups pixels into tokens, matches the tokens, and maps their motion back to pixels.** Shared image features feed the membership head and token description. The learned assignment pools features into tokens, which gather local context before matching and iterative motion refinement. The Image 1 assignment broadcasts token motion back to its pixels. Image 2 follows the same feature, membership, and description pipeline with shared weights.

grouping can exploit without a finer grid. DACE therefore uses high-resolution features to place token boundaries and build token descriptors, while estimating motion on fewer adaptive tokens.

3 Method

Overview. DACE learns *what to group* together with *how it moves*. Given Image 1, I_1, and Image 2, I_2, each of size $H \times W$, DACE estimates every pixel's displacement $x \mapsto x + \hat{f}(x)$, with $\hat{f} \in \mathbb{R}^{2\times H\times W}$. Instead of estimating motion at fixed grid points, DACE groups pixels into tokens, matches the tokens between the two images, and maps their motion back to pixels.

Figure 3 summarizes the pipeline. A shared backbone extracts dense features from each image. A membership head predicts how strongly each pixel belongs to each nearby token. Pooling features over each token's pixels, followed by local attention, produces token descriptors. Matching and refinement then estimate each token's motion. Image 1's memberships broadcast token motion back to pixels, so the output stays dense although motion is estimated only per token. The same memberships also define the training losses (Section 3.3). Because every step is differentiable, grouping and correspondence are learned end to end from shared features.

3.1 Learning the correspondence unit

Pixel-to-token assignment. We predict each pixel's soft memberships over nearby tokens from the dense features (Figure 4(a)). Tokens have fixed anchors q on a lattice spaced S image pixels apart. The lattice fixes how many tokens there are and roughly where they sit; the image determines which pixels each token covers. Pixel x can belong to the tokens at the 3×3 anchors around its cell, $\mathcal{N}(x) = \{q_i(x)\}_{i=1}^{9}$, with local offsets $\delta_i \in \{-1, 0, 1\}^2$ in token-lattice units. A token therefore draws pixels only from its own and neighboring cells.

The membership head predicts five parameters per pixel: a mean $\mu(x) \in \mathbb{R}^2$ and a lower-triangular factor $L(x)$ with positive diagonal, which defines the inverse covariance $G(x) = L(x)L(x)^\top$. A Gaussian with this mean and inverse covariance scores each candidate anchor, and a softmax converts the scores to memberships:

$$
\begin{aligned}
a_i(x) &= -\tfrac{1}{2}(\delta_i - \mu(x))^\top G(x)(\delta_i - \mu(x)), \\
A_i(x) &= \frac{\exp a_i(x)}{\sum_{j=1}^{9} \exp a_j(x)}.
\end{aligned}
\tag{1}
$$

Here $A_i(x)$ is the probability that pixel x belongs to token $q_i(x)$, with $\sum_{i=1}^{9} A_i(x) = 1$. The mean shifts membership toward some anchors, and the inverse covariance sets how concentrated it is and along which direction. A concentrated Gaussian puts a pixel on a single token, as needed at a motion boundary; a broad one spreads it over several tokens, as inside a coherent region.

Pooling and broadcasting. The assignment $A = \{A_i\}_{i=1}^{9}$ defines two operators: P_A pools pixel values into tokens, and B_A broadcasts token values to pixels (Figure 4(b,c)). For a dense map z,

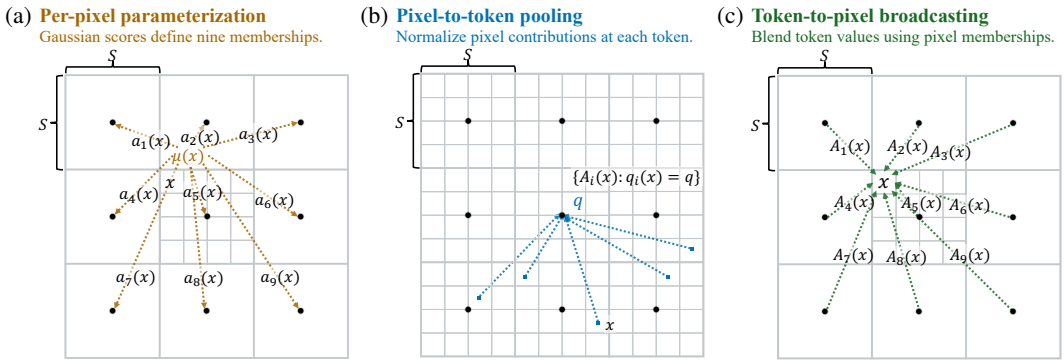


Figure 4: **One assignment connects membership prediction, pooling, and broadcasting.** (a) Gaussian scores $a_i(x)$ for pixel x are normalized into memberships $A_i(x)$ over its nine neighboring anchors. (b) Incoming memberships define a weighted mean of pixel values at token q. (c) Each pixel blends neighboring token values using the same memberships. Black dots denote fixed anchors, spaced S pixels apart; only a few pooling links are drawn for clarity. All operations are differentiable.

$(P_A z)(q)$ is the weighted mean of all pixels that have q among their candidates. The condition $q_i(x) = q$ selects these links; their local index i can differ across pixels:

$$(P_A z)(q) = \frac{\sum_{x,i:\, q_i(x)=q} A_i(x) z(x)}{\sum_{x,i:\, q_i(x)=q} A_i(x) + \epsilon}. \tag{2}$$

Here ϵ stabilizes the denominator. Broadcasting blends the nine neighboring values of a token map u using each pixel's memberships:

$$(B_A u)(x) = \sum_{i=1}^{9} A_i(x) u(q_i(x)). \tag{3}$$

Both operators are differentiable, so grouping can be learned jointly with correspondence.

3.2 Describing and matching adaptive tokens

Token descriptors. Unlike standard pooling or patch merging, which combine features over fixed spatial groups [37], DACE pools features over the pixels assigned to each token, so each descriptor summarizes an image-dependent group of pixels. Each pooled feature $\bar{T}(q) = (P_A F)(q)$ then queries the dense feature map F through local attention, with features in a window $\mathcal{W}(q)$ as keys and values:

$$T(q) = \mathrm{LocalAttn}\big(\bar{T}(q), F|_{\mathcal{W}(q)}, F|_{\mathcal{W}(q)}\big). \tag{4}$$

The grouping defines what each token is; attention adds context from nearby dense features that helps recognize it in the other image, without adding tokens whose motion must be estimated.

Matching and motion inference. We match the two images' token maps T_1, T_2 by iterative warping and refinement, following the recurrent update of WAFT [15]. At step t, $d_t(q)$ is the displacement in *token-lattice units* and h_t is the recurrent state. Starting from $d_0 = 0$, warping bilinearly samples T_2 at $q + d_t(q)$, and the update module Ψ predicts a motion correction:

$$(\Delta d_t, h_{t+1}) = \Psi\big(T_1, \mathrm{warp}(T_2, d_t), h_t, d_t\big), \qquad d_{t+1} = d_t + \Delta d_t. \tag{5}$$

The assignment is predicted once per image and reused across updates. Image 1's memberships broadcast token motion to pixels, with S converting it to image-pixel units:

$$\hat{f}_t(x) = \big(B_A(S d_t)\big)(x). \tag{6}$$

Refinement combines matching evidence and spatial context across tokens, and broadcasting passes each token's motion to all of its pixels. A pixel that becomes occluded or leaves the frame can thus receive motion estimated from visible parts of the same structure.

3.3 Learning grouping and correspondence jointly

Three losses, all computed from ground-truth flow, train the model jointly. The token loss supervises each token's motion. Two further losses act on the assignment directly: one uses the predicted token motions to teach each pixel which neighboring token moves like it, and the other uses ground-truth flow alone to keep each token within a single motion. All three shape the grouping, so it is learned from motion rather than from an appearance objective or segmentation labels.

Token motion supervision. For each token, the model predicts motion $Sd(q)$ in image pixels and an uncertainty, following the probabilistic loss of SEA-RAFT [28]. The likelihood $p_q(\cdot;\theta_q)$ is a mixture of Laplace distributions centered on the predicted motion; their scales and mixture weights express the uncertainty, and θ_q collects all these parameters. The target is ground-truth flow f^* pooled over the token's pixels. With $\ell_{\mathrm{NLL}}(p,u)$ denoting the negative log-likelihood of target u under prediction p, the token loss is

$$\bar{u}^* = P_{\mathrm{sg}(A)} f^*, \qquad \ell_{\mathrm{token}}(q) = \ell_{\mathrm{NLL}}(p_q(\cdot;\theta_q), \bar{u}^*(q)) \,. \tag{7}$$

Here sg stops gradients through the pooled target $\bar{u}^*$, which is recomputed as the assignment changes. Without it, the assignment could lower the loss by moving the target toward the prediction. Gradients through matching, refinement, and descriptors still reach the backbone and membership head, so this loss also teaches the grouping which pixels provide useful matching evidence.

Assignment supervision. This loss teaches each pixel to favor the neighboring token whose motion explains the pixel's own flow. At an object boundary, a pixel's flow may match a token on the object but not one on the background. We mix the token likelihoods with weights $R(x,q)$, the pixel's memberships renormalized over the neighboring tokens used in this loss. Holding each token's motion and uncertainty fixed, we evaluate the pixel's flow $f^*(x)$ under this mixture:

$$\ell_{\mathrm{assign}}(x) = \ell_{\mathrm{NLL}}\left(\sum_{q \in \mathcal{N}(x)} R(x,q)\, p_q(\cdot;\mathrm{sg}(\theta_q)), f^*(x) \right). \tag{8}$$

Because the loss compares each token's prediction with the pixel's flow instead of averaging the predictions first, it rewards putting weight on the token that moves with the pixel rather than on a blend of motions. This encourages sharp grouping at motion boundaries. Detaching θ_q sends this loss only to the membership head and backbone; token motion is supervised by the token loss.

Flow-consistency supervision. Following the reconstruction loss of SSN [35], we pool ground-truth flow into tokens, broadcast it back, and compare the reconstruction f_A^* with the original flow using a Charbonnier penalty ℓ_{rec}:

$$u_A^* = P_A f^*, \qquad f_A^* = B_A u_A^*, \qquad \ell_{\mathrm{flow}}(x) = \ell_{\mathrm{rec}}(f_A^*(x), f^*(x)) \,. \tag{9}$$

Gradients reach the membership head and backbone through both operators. A token spanning two motions reconstructs neither well, so this loss keeps each token within a single motion.

Combined objective. We apply the token and assignment losses at every refinement step and write their averages at step t as $\mathcal{L}_{\mathrm{token},t}$ and $\mathcal{L}_{\mathrm{assign},t}$. Since the assignment is shared across steps, the flow-consistency loss $\mathcal{L}_{\mathrm{flow}}$ is added once. With T refinement updates, iteration decay γ, and weight λ_{flow}, the combined objective is

$$\mathcal{L} = \sum_{t=1}^{T} \gamma^{T-t} \big(\mathcal{L}_{\mathrm{token},t} + \mathcal{L}_{\mathrm{assign},t} \big) + \lambda_{\mathrm{flow}} \mathcal{L}_{\mathrm{flow}}. \tag{10}$$

Each loss directly supervises motion or the assignment; all three shape the shared features.

4 Experiments

We compare DACE with prior methods, then examine where it reduces error, whether adaptive tokens drive the gains, which design choices matter, and how accuracy trades off against computation.

Experimental setup. We build DACE on WAFT [15], sharing its DINOv3 ViT-S backbone, feature warping, and transformer-based recurrent update, with five updates in training and inference. We use this backbone throughout the main paper and report Depth Anything V2 [38] results in Appendix A. To isolate the benefit of adaptive tokens, we compare DACE with fixed-grid models. For an $H \times W$ image, all models extract features at $(H/2) \times (W/2)$, and DACE estimates motion for tokens on an $(H/4) \times (W/4)$ lattice. WAFT estimates motion on a fixed grid at the feature resolution. WAFT-H/4 differs only in estimating motion on a fixed $(H/4) \times (W/4)$ grid; we write $H/2$ and $H/4$ for these resolutions. We also test WAFT-H/4 with the local attention that builds DACE's token descriptors (Eq. (4)), so each grid point draws context from the dense features. The fixed-grid models use convex upsampling [12], whereas DACE broadcasts token motion to pixels through the memberships (Eq. (6)). WAFT results use its released checkpoint. We train both WAFT-H/4 variants with WAFT's code and loss, and all models we train follow WAFT's training data and optimization recipe.

We train on FlyingChairs and then FlyingThings3D, and evaluate zero-shot on the training splits of Sintel and KITTI. For in-distribution analyses, we also evaluate on the FlyingThings3D test split (Things). Things and Sintel each have Clean and Final renderings, abbreviated C and F in tables.

Metrics. Endpoint error (EPE) is the distance in pixels between predicted and ground-truth flow, averaged within each image and then over images. KITTI also reports Fl, the percentage of pixels whose error exceeds both 3 pixels and 5% of the true motion. For regional analyses, we pool EPE over all pixels in a subset, such as pixels whose target is occluded or out of frame (Appendix B.1). On Things we use image bounds only, so this subset contains only out-of-frame pixels. MACs count multiply–accumulate operations for one 540×960 image pair.

Table 1: **DACE achieves the lowest Sintel Final and KITTI EPE with fewer MACs than most methods.** Zero-shot results on the Sintel and KITTI training splits after training on FlyingChairs and FlyingThings3D. Bold: lowest error in each column. –: not reported.

Method	Sintel		KITTI		MACs
	Clean	Final	EPE	Fl (%)	(G)
FlowNet2 [26]	2.02	3.14	10.10	30.4	230
PWC-Net [11]	2.55	3.93	10.40	33.7	101
RAFT [12]	1.43	2.71	5.04	17.4	938
GMA [32]	1.30	2.74	4.69	17.1	1352
SKFlow [39]	1.22	2.46	4.27	15.5	751
DIP [40]	1.30	2.82	4.29	13.7	3068
CRAFT [41]	1.27	2.79	4.88	17.5	2274
GMFlowNet [42]	1.14	2.71	4.24	15.4	1094
FlowFormer [14]	1.01	2.40	4.09	14.7	1715
EMD-L [43]	0.88	2.55	4.12	13.5	1755
FlowFormer++ [30]	0.90	2.30	3.93	14.1	1713
MatchFlow(G) [31]	1.03	2.45	4.08	15.6	1669
AnyFlow [44]	1.10	2.52	3.76	12.4	–
RPKNet [45]	1.12	2.45	–	13.0	137
CCMR+ [46]	0.98	2.36	–	12.9	12653
SEA-RAFT(L) [28]	1.19	4.11	3.62	12.9	655
SAMFlow [47]	0.87	2.11	3.44	12.3	9717
FlowDiffuser [17]	**0.86**	2.19	3.61	11.8	2466
DPFlow [29]	1.02	2.26	3.37	**11.1**	414
WAFT [15]	1.28	2.56	3.49	12.9	732
DACE	1.03	**2.10**	**3.11**	11.9	796

4.1 Zero-shot transfer

DACE achieves the lowest EPE in realistic settings. Training on synthetic data and testing on unseen datasets measures transfer across data distributions. This is the most relevant indicator for real-world use, where new domains rarely have flow labels. In Table 1, DACE has the lowest EPE on Sintel Final, which adds realistic effects such as motion blur, and on KITTI's real driving scenes. DACE also needs fewer MACs than most methods. The two methods closest to it on Sintel Final, SAMFlow and FlowDiffuser, need 3 to 12 times as many.

Table 2: **DACE lowers EPE where no match is visible and near motion boundaries.** EPE is pooled over the pixels of each region. Region (c) contains pixels within two pixels of a ground-truth motion boundary and overlaps (a) and (b). Share: the region's percentage of valid pixels.

	(a) Visible		(b) Occluded/out-of-frame			(c) Motion boundaries		
Dataset	WAFT	DACE	Share (%)	WAFT	DACE	Share (%)	WAFT	DACE
Things C	2.14	**1.37**	6.4	23.03	**16.61**	9.3	13.40	**9.24**
Things F	2.09	**1.40**	6.4	21.96	**15.89**	9.3	13.14	**9.48**
Sintel C	0.65	**0.49**	8.4	7.95	**6.77**	6.2	7.30	**6.63**
Sintel F	1.59	**1.28**	8.4	12.86	**10.93**	6.2	11.92	**10.45**
KITTI	**2.01**	2.05	13.9	13.65	**11.05**	13.6	9.97	**8.51**

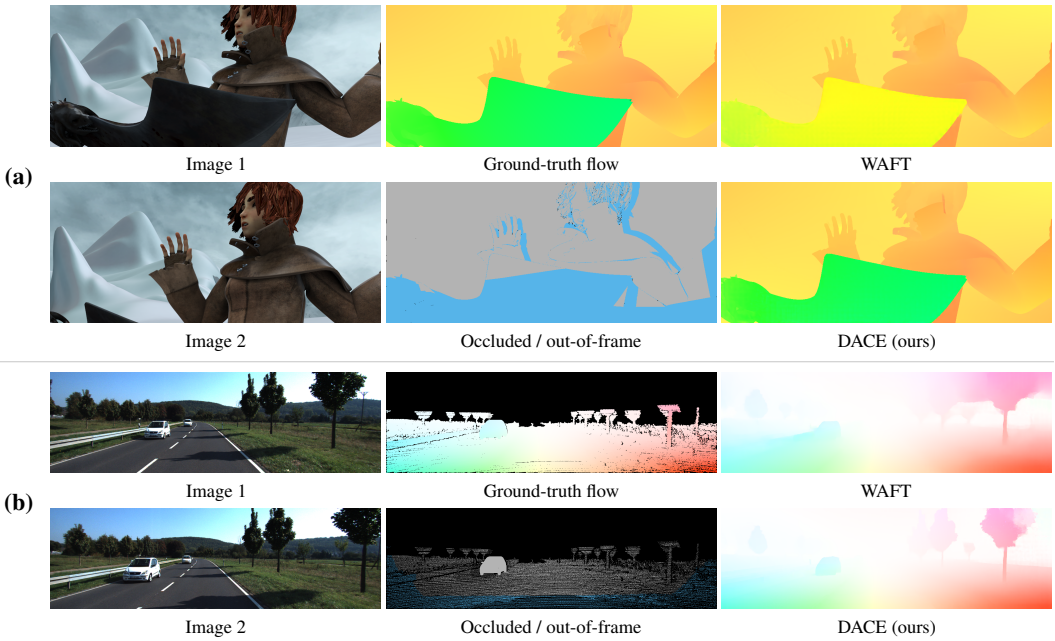


Figure 5: **DACE recovers motion where no match is visible and preserves fine structure.** (a) Sintel and (b) KITTI examples. Both models are evaluated zero-shot after training only on FlyingChairs and FlyingThings3D. In occluded/out-of-frame maps, blue marks these pixels, gray other valid pixels, and black pixels excluded from evaluation. Ground truth and both predictions share one color mapping per pair: hue shows direction and intensity shows magnitude.

4.2 Where DACE reduces error

Occluded and out-of-frame pixels concentrate flow error. Table 2 compares WAFT and DACE on three regions: visible pixels, occluded or out-of-frame pixels, and pixels near motion boundaries. The second region makes up 6.4–13.9% of valid pixels, yet causes 42.3–52.2% of WAFT's EPE (Appendix B.1). Such pixels are more common in dynamic scenes and under large camera motion, as in KITTI's driving videos. With no visible match, their motion must be inferred from context.

DACE is more robust where no match is visible. A token covers many pixels, so its motion can be estimated from those that stay visible and shared with those that do not. In Table 2, DACE lowers EPE on occluded and out-of-frame pixels on every dataset. On Things and Sintel, these reductions are six to nine times those on visible pixels. On KITTI, DACE's visible-pixel EPE stays close to WAFT's (2.05 vs. 2.01), so DACE's KITTI gain comes from occluded and out-of-frame pixels.

DACE is more precise near motion boundaries. Precision matters most at motion boundaries, where a small misplacement gives pixels the motion of the wrong surface. We define boundary EPE as the EPE within two pixels of ground-truth motion boundaries, found by thresholding the local flow change at 2 pixels. In Table 2, these pixels make up 6.2–13.6% of valid pixels, and DACE lowers their EPE on every dataset, although WAFT estimates motion for four times as many units. DACE's

advantage also holds at thresholds of 3 and 4 pixels and against WAFT-H/4 with local attention (Appendix B.2). DACE's tokens can end at a boundary instead of mixing the motions on both sides.

Qualitative comparison. In Figure 5(a), the blue mask marks where the cape has no visible match. DACE recovers its motion and outline, which WAFT largely misses. In (b), DACE predicts coherent motion where the road leaves the frame and keeps sharper outlines of the vehicle and trees. Appendix C shows four additional examples with corresponding token and assignment-entropy maps.

4.3 Adaptive tokens versus a fixed grid

To isolate the effect of adaptivity, we compare DACE with both WAFT-H/4 variants, which estimate motion on a fixed grid at DACE's token resolution.

Table 3: **Adaptive tokens outperform a fixed grid at the same resolution and gain more from context.** Attention gain is EPE without minus with local attention.

(a) Overall accuracy

		Things		Sintel		KITTI	
System	Attn.	C	F	C	F	EPE	Fl (%)
WAFT-H/4	–	2.93	2.72	1.08	2.49	3.96	13.3
WAFT-H/4	✓	2.73	2.62	1.07	2.57	3.75	12.6
DACE	–	2.56	2.48	1.12	2.14	3.54	13.0
DACE	✓	**2.39**	**2.38**	**1.03**	**2.10**	**3.11**	**11.9**

(b) Occluded/out-of-frame EPE

		Things		Sintel		KITTI
System	Attn.	C	F	C	F	EPE
WAFT-H/4	–	19.65	17.79	7.15	12.91	14.87
WAFT-H/4	✓	18.98	17.64	7.19	12.90	14.09
Attention gain		0.67	0.15	−0.04	0.01	0.78
DACE	–	17.49	16.55	6.89	11.40	12.96
DACE	✓	16.61	15.89	6.77	10.93	11.05
Attention gain		**0.88**	**0.66**	**0.12**	**0.47**	**1.91**

Adaptive tokens outperform a fixed grid at the same resolution. In Table 3(a), DACE improves every metric over both WAFT-H/4 variants. Without attention in either model, DACE is still better on five of six metrics. Adding attention to the fixed grid helps on Things and KITTI but does not close the gap, so the gain is not explained by added context alone.

Context helps adaptive tokens more where no match is visible. In Table 3(b), local attention lowers occluded/out-of-frame EPE more for DACE than for the fixed grid on every dataset.

4.4 Design choices

Table 4 changes one design choice of DACE at a time.

Table 4: **Every alternative is less accurate than DACE on most metrics.** Each row changes one design choice from DACE and keeps all other settings fixed.

					Things		Sintel		KITTI	
Variant	Membership	Local attn.	Flow consist.	Stop-gradient	C	F	C	F	EPE	Fl (%)
DACE	Gaussian	✓	✓	✓	**2.39**	**2.38**	1.03	**2.10**	**3.11**	**11.9**
Direct logits	Direct	✓	✓	✓	2.58	2.55	1.02	2.24	3.16	12.1
No local attention	Gaussian	–	✓	✓	2.56	2.48	1.12	2.14	3.54	13.0
No flow consistency	Gaussian	✓	–	✓	2.68	2.61	1.05	2.17	3.14	12.4
No stop-gradient	Gaussian	✓	✓	–	2.49	2.46	**1.00**	2.18	3.64	12.3

Gaussian memberships outperform direct logits. The direct-logit variant predicts the scores $a_i(x)$ directly instead of from a Gaussian (Eq. (1)). The Gaussian improves Things, Sintel Final, and KITTI, so a Gaussian shape helps the memberships, which still adapt to the image.

Local attention improves every metric. The grouping defines each token, and attention adds nearby context that helps recognize it in the other image, especially on KITTI (3.54 to 3.11 EPE).

The flow-consistency loss improves every metric. Without this loss (Eq. (9)), the grouping is trained only through the token and assignment losses. Asking each token directly to stay within one motion complements what the grouping learns from matching.

The stop-gradient in the assignment loss helps. Removing the stop-gradient on θ_q in Eq. (8) lets the assignment loss also update token motion and uncertainty. This hurts Things, Sintel Final, and KITTI, and helps only Sintel Clean. Token motion is better learned from the token loss alone, leaving the assignment loss to train the membership head and backbone.

4.5 Accuracy and computational cost

With a quarter as many units as WAFT, DACE can afford more computation per unit.

Table 5: **Fewer units leave room for richer processing.** Costs are GMACs for one 540×960 image pair; errors are EPE except KITTI Fl. †: profiled only, not trained.

			Cost (GMACs)			Things		Sintel		KITTI	
System	Resolution	Updates	Attention	Refinement	Total	C	F	C	F	EPE	Fl (%)
WAFT-64 [15]	$H/2$	5	–	425.8	732.4	3.54	3.44	1.28	2.56	3.49	12.9
WAFT-128 + attn.†	$H/2$	5	57.0	943.9	1313.2	–	–	–	–	–	–
DACE-64	$H/4$	5	20.6	**293.8**	**636.3**	2.55	2.48	**1.03**	2.24	3.52	12.5
DACE-128	$H/4$	4	20.6	363.1	706.1	2.44	2.43	**1.03**	2.15	3.18	12.2
DACE-128 (main)	$H/4$	5	20.6	453.9	796.8	**2.39**	**2.38**	**1.03**	**2.10**	**3.11**	**11.9**

Model variants. Suffixes in Table 5 give the channel width of the recurrent state h_t and its refinement layers. WAFT-64 is the released WAFT, and DACE-128 is our main model. DACE-64 halves the state width. WAFT-128 widens WAFT's state and adds attention of the same width.

Fewer units leave room for richer processing. Widening WAFT more than doubles its refinement cost (425.8 to 943.9 GMACs). On DACE's coarser lattice, local attention of the same width costs about a third as much (20.6 vs. 57.0 GMACs). DACE-64 costs 13% less than WAFT and is more accurate on five of six metrics, all but KITTI EPE. Run with four updates instead of five, DACE-128 is also cheaper than WAFT and more accurate on all six (Appendix D).

5 Conclusion

DACE learns the units of correspondence for dense optical flow. A differentiable assignment groups pixels into tokens and maps their motion back to pixels, so grouping and matching are learned jointly. Motion is estimated on coarser tokens, while image evidence stays dense. DACE achieves the lowest zero-shot Sintel Final and KITTI EPE among published methods, with fewer MACs than most. Even against a fixed grid with four times as many units, DACE lowers EPE on every dataset, is more robust where no match is visible, and is more precise near motion boundaries. DACE also improves every metric over a fixed grid at its own resolution, which isolates the benefit of adaptivity. Learning the units to match complements learning the evidence used to match them.

6 Limitations

DACE's zero-shot advantage does not yet carry over to target-specific fine-tuning. Fine-tuned with WAFT's recipe, it matches WAFT on Sintel Final but trails it on Sintel Clean, KITTI, and Spring (Appendix E). Improving how DACE adapts to a target domain remains future work.

References

[1] Berthold K. P. Horn and Brian G. Schunck. Determining optical flow. *Artificial Intelligence*, 17 (1):185–203, August 1981. ISSN 0004-3702. doi: 10.1016/0004-3702(81)90024-2.

[2] Bruce D Lucas and Takeo Kanade. An Iterative Image Registration Technique with an Application to Stereo Vision. In *IJCAI'81: 7th International Joint Conference on Artificial Intelligence*, volume 2, pages 674–679, Vancouver, Canada, August 1981.

[3] Carl Doersch, Ankush Gupta, Larisa Markeeva, Adrià Recasens, Lucas Smaira, Yusuf Aytar, João Carreira, Andrew Zisserman, and Yi Yang. Tap-vid: A benchmark for tracking any point in a video. In *Advances in Neural Information Processing Systems*, volume 35, pages 13610–13626, 2022.

[4] Moab Arar, Yiftach Ginger, Dov Danon, Amit H. Bermano, and Daniel Cohen-Or. Unsupervised multi-modal image registration via geometry preserving image-to-image translation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020.

[5] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos, 2014. URL `https://arxiv.org/abs/1406.2199`.

[6] Moritz Menze and Andreas Geiger. Object Scene Flow for Autonomous Vehicles. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3061–3070, 2015.

[7] Yixiang Chen, Peiyan Li, Yuan Xu, Qisen Ma, et al. FlowWAM: Optical flow as a unified action representation for world action models, 2026. URL `https://arxiv.org/abs/2607.13017`.

[8] Adam W. Harley, Zhaoyuan Fang, and Katerina Fragkiadaki. Particle video revisited: Tracking through occlusions using point trajectories. In *Computer Vision – ECCV 2022*, pages 59–75. Springer, 2022.

[9] Carl Doersch, Yi Yang, Mel Vecerik, Dilara Gokay, Ankush Gupta, Yusuf Aytar, João Carreira, and Andrew Zisserman. TAPIR: Tracking any point with per-frame initialization and temporal refinement. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10061–10072, 2023.

[10] Nikita Karaev, Ignacio Rocco, Benjamin Graham, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. Cotracker: It is better to track together. In *Computer Vision – ECCV 2024*, pages 18–35. Springer, 2024.

[11] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. PWC-Net: CNNs for Optical Flow Using Pyramid, Warping, and Cost Volume. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8934–8943, 2018.

[12] Zachary Teed and Jia Deng. RAFT: Recurrent All-Pairs Field Transforms for Optical Flow. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, pages 402–419, Cham, 2020. Springer International Publishing. ISBN 978-3-030-58536-5. doi: 10.1007/978-3-030-58536-5_24.

[13] Haofei Xu, Jing Zhang, Jianfei Cai, Hamid Rezatofighi, and Dacheng Tao. GMFlow: Learning Optical Flow via Global Matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8121–8130, 2022.

[14] Zhaoyang Huang, Xiaoyu Shi, Chao Zhang, Qiang Wang, Ka Chun Cheung, Hongwei Qin, Jifeng Dai, and Hongsheng Li. FlowFormer: A Transformer Architecture for Optical Flow. In Shai Avidan, Gabriel Brostow, Moustapha Cissé, Giovanni Maria Farinella, and Tal Hassner, editors, *Computer Vision – ECCV 2022*, pages 668–685, Cham, 2022. Springer Nature Switzerland. ISBN 978-3-031-19790-1. doi: 10.1007/978-3-031-19790-1_40.

[15] Yihan Wang and Jia Deng. WAFT: Warping-Alone Field Transforms for Optical Flow, October 2025.

[16] Saurabh Saxena, Charles Herrmann, Junhwa Hur, Abhishek Kar, Mohammad Norouzi, Deqing Sun, and David J. Fleet. The surprising effectiveness of diffusion models for optical flow and monocular depth estimation. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=jDIlzSU8wJ.

[17] Ao Luo, Xin Li, Fan Yang, Jiangyu Liu, Haoqiang Fan, and Shuaicheng Liu. FlowDiffuser: Advancing Optical Flow Estimation with Diffusion Models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 19167–19176, 2024.

[18] Luming Tang, Menglin Jia, Qianqian Wang, Cheng Perng Phoo, and Bharath Hariharan. Emergent correspondence from image diffusion. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=ypOiXjdfnU.

[19] Junyi Zhang, Charles Herrmann, Junhwa Hur, Luisa Polania Cabrera, Varun Jampani, Deqing Sun, and Ming-Hsuan Yang. A tale of two features: Stable diffusion complements dino for zero-shot semantic correspondence. *arXiv preprint arXiv:2305.15347*, 2023.

[20] Max Wertheimer. Laws of organization in perceptual forms. In *A Source Book of Gestalt Psychology*, pages 71–88. Kegan Paul, Trench, Trubner & Company, London, England, 1938. doi: 10.1037/11496-005.

[21] Alan L. Yuille and Norberto M. Grzywacz. A computational theory for the perception of coherent visual motion. *Nature*, 333(6168):71–74, May 1988. ISSN 1476-4687. doi: 10.1038/333071a0.

[22] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Hausser, Caner Hazirbas, Vladimir Golkov, Patrick van der Smagt, Daniel Cremers, and Thomas Brox. FlowNet: Learning Optical Flow With Convolutional Networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2758–2766, 2015.

[23] Nikolaus Mayer, Eddy Ilg, Philip Hausser, Philipp Fischer, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox. A Large Dataset to Train Convolutional Networks for Disparity, Optical Flow, and Scene Flow Estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4040–4048, 2016.

[24] Daniel J. Butler, Jonas Wulff, Garrett B. Stanley, and Michael J. Black. A Naturalistic Open Source Movie for Optical Flow Evaluation. In Andrew Fitzgibbon, Svetlana Lazebnik, Pietro Perona, Yoichi Sato, and Cordelia Schmid, editors, *Computer Vision – ECCV 2012*, pages 611–625, Berlin, Heidelberg, 2012. Springer. ISBN 978-3-642-33783-3. doi: 10.1007/978-3-642-33783-3_44.

[25] A Geiger, P Lenz, C Stiller, and R Urtasun. Vision meets robotics: The KITTI dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, September 2013. ISSN 0278-3649. doi: 10.1177/0278364913491297.

[26] Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox. FlowNet 2.0: Evolution of Optical Flow Estimation With Deep Networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2462–2470, 2017.

[27] Anurag Ranjan and Michael J. Black. Optical Flow Estimation Using a Spatial Pyramid Network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4161–4170, 2017.

[28] Yihan Wang, Lahav Lipson, and Jia Deng. SEA-RAFT: Simple, Efficient, Accurate RAFT for Optical Flow. In Aleš Leonardis, Elisa Ricci, Stefan Roth, Olga Russakovsky, Torsten Sattler, and Gül Varol, editors, *Computer Vision – ECCV 2024*, pages 36–54, Cham, 2025. Springer Nature Switzerland. ISBN 978-3-031-72667-5. doi: 10.1007/978-3-031-72667-5_3.

[29] Henrique Morimitsu, Xiaobin Zhu, Roberto M. Cesar, Xiangyang Ji, and Xu-Cheng Yin. DPFlow: Adaptive Optical Flow Estimation with a Dual-Pyramid Framework. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17810–17820, 2025.

[30] Xiaoyu Shi, Zhaoyang Huang, Dasong Li, Manyuan Zhang, Ka Chun Cheung, Simon See, Hongwei Qin, Jifeng Dai, and Hongsheng Li. FlowFormer++: Masked Cost Volume Autoencoding for Pretraining Optical Flow Estimation. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1599–1610, Vancouver, BC, Canada, June 2023. IEEE. ISBN 979-8-3503-0129-8. doi: 10.1109/CVPR52729.2023.00160.

[31] Qiaole Dong, Chenjie Cao, and Yanwei Fu. Rethinking Optical Flow from Geometric Matching Consistent Perspective. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1337–1347, Vancouver, BC, Canada, June 2023. IEEE. ISBN 979-8-3503-0129-8. doi: 10.1109/CVPR52729.2023.00135.

[32] Shihao Jiang, Dylan Campbell, Yao Lu, Hongdong Li, and Richard Hartley. Learning To Estimate Hidden Motions With Global Motion Aggregation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9772–9781, 2021.

[33] Radhakrishna Achanta, Appu Shaji, Kevin Smith, Aurelien Lucchi, Pascal Fua, and Sabine Süsstrunk. SLIC Superpixels Compared to State-of-the-Art Superpixel Methods. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(11):2274–2282, November 2012. ISSN 1939-3539. doi: 10.1109/TPAMI.2012.120.

[34] Michael Van den Bergh, Xavier Boix, Gemma Roig, Benjamin de Capitani, and Luc Van Gool. SEEDS: Superpixels Extracted via Energy-Driven Sampling. In Andrew Fitzgibbon, Svetlana Lazebnik, Pietro Perona, Yoichi Sato, and Cordelia Schmid, editors, *Computer Vision – ECCV 2012*, pages 13–26, Berlin, Heidelberg, 2012. Springer. ISBN 978-3-642-33786-4. doi: 10.1007/978-3-642-33786-4_2.

[35] Varun Jampani, Deqing Sun, Ming-Yu Liu, Ming-Hsuan Yang, and Jan Kautz. Superpixel Sampling Networks, July 2018.

[36] Fengting Yang, Qian Sun, Hailin Jin, and Zihan Zhou. Superpixel Segmentation With Fully Convolutional Networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13964–13973, 2020.

[37] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin Transformer: Hierarchical Vision Transformer Using Shifted Windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10012–10022, 2021.

[38] Lihe Yang, Bingyi Kang, Zilong Huang, Zhen Zhao, Xiaogang Xu, Jiashi Feng, and Hengshuang Zhao. Depth Anything V2. *Advances in Neural Information Processing Systems*, 37:21875–21911, December 2024. doi: 10.52202/079017-0688.

[39] Shangkun Sun, Yuanqi Chen, Yu Zhu, Guodong Guo, and Ge Li. SKFlow: Learning Optical Flow with Super Kernels. In *Advances in Neural Information Processing Systems*, 2022.

[40] Zihua Zheng, Ni Nie, Zhi Ling, Pengfei Xiong, Jiangyu Liu, Hao Wang, and Jiankun Li. DIP: Deep Inverse Patchmatch for High-Resolution Optical Flow. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8925–8934, 2022.

[41] Xiuchao Sui, Shaohua Li, Xue Geng, Yan Wu, Xinxing Xu, Yong Liu, Rick Goh, and Hongyuan Zhu. CRAFT: Cross-Attentional Flow Transformer for Robust Optical Flow. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17602–17611, 2022.

[42] Shiyu Zhao, Long Zhao, Zhixing Zhang, Enyu Zhou, and Dimitris Metaxas. Global Matching with Overlapping Attention for Optical Flow Estimation. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 17571–17580, New Orleans, LA, USA, June 2022. IEEE. ISBN 978-1-6654-6946-3. doi: 10.1109/CVPR52688.2022.01707.

[43] Changxing Deng, Ao Luo, Haibin Huang, Shaodan Ma, Jiangyu Liu, and Shuaicheng Liu. Explicit Motion Disentangling for Efficient Optical Flow Estimation. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9487–9496, Paris, France, October 2023. IEEE. ISBN 979-8-3503-0718-4. doi: 10.1109/ICCV51070.2023.00873.

[44] Hyunyoung Jung, Zhuo Hui, Lei Luo, Haitao Yang, Feng Liu, Sungjoo Yoo, Rakesh Ranjan, and Denis Demandolx. AnyFlow: Arbitrary Scale Optical Flow With Implicit Neural Representation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5455–5465, 2023.

[45] Henrique Morimitsu, Xiaobin Zhu, Xiangyang Ji, and Xu-Cheng Yin. Recurrent Partial Kernel Network for Efficient Optical Flow Estimation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(5):4278–4286, March 2024. ISSN 2374-3468. doi: 10.1609/aaai.v38i5.28224.

[46] Azin Jahedi, Maximilian Luz, Marc Rivinius, and Andrés Bruhn. CCMR: High Resolution Optical Flow Estimation via Coarse-To-Fine Context-Guided Motion Reasoning. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 6899–6908, 2024.

[47] Shili Zhou, Ruian He, Weimin Tan, and Bo Yan. SAMFlow: Eliminating Any Fragmentation in Optical Flow with Segment Anything Model. *Proceedings of the AAAI Conference on Artificial Intelligence*, 38(7):7695–7703, March 2024. ISSN 2374-3468. doi: 10.1609/aaai.v38i7.28603.

[48] MPI Sintel. Optical flow evaluation metrics. `https://sintel.is.tuebingen.mpg.de/results`. Accessed September 29, 2026.

Appendix overview. Appendix A repeats the zero-shot comparison with a second backbone. Appendix B defines the occluded/out-of-frame and motion-boundary regions used in Section 4.2 and extends their analysis to more thresholds and large displacements. Appendix C adds qualitative examples, and Appendix D reports accuracy and cost per number of updates. Appendix E reports fine-tuned benchmark results, and Appendix F tests how well DACE's tokens can represent motion.

Common setting. Unless stated otherwise, results use the DINOv3 models of Section 4: DACE, the released WAFT, and WAFT-H/4 with local attention, all trained on FlyingChairs and FlyingThings3D. We evaluate them on the Things test split and, zero-shot, on the Sintel and KITTI training splits.

A Zero-Shot Transfer with Another Backbone

Setup. Table 6 compares DACE and WAFT with two backbones: DINOv3, used throughout the main paper, and Depth Anything V2 (DAv2) [38]. WAFT results are from its paper [15]. All models are trained on FlyingChairs and FlyingThings3D and evaluated zero-shot.

DACE's advantage holds with both backbones. With each backbone, DACE lowers EPE relative to WAFT on both Sintel renderings and on KITTI.

Table 6: **DACE lowers EPE with both backbones.** Zero-shot results on the Sintel and KITTI training splits after training on FlyingChairs and FlyingThings3D. Bold: better of each pair.

		Sintel		KITTI	
Backbone	Method	Clean	Final	EPE	Fl (%)
DINOv3	WAFT [15]	1.28	2.56	3.49	12.9
	DACE	**1.03**	**2.10**	**3.11**	**11.9**
DAv2	WAFT [15]	1.01	2.49	3.28	**10.9**
	DACE	**0.97**	**2.30**	**2.93**	**10.9**

B Occlusion, Motion Boundaries, and Large Displacements

We examine three properties of a pixel: whether its target is visible in Image 2, whether its neighbors move differently, and how far it moves. Each region's EPE is pooled over its pixels.

B.1 Occluded and out-of-frame pixels

Definition. A pixel is out of frame if its ground-truth endpoint $x + f^*(x)$ falls outside Image 2. On Sintel and KITTI, a pixel also belongs to this region if the dataset's ground truth marks it as occluded in Image 2. On Things we use image bounds only, so its region contains only out-of-frame pixels. All regions are restricted to valid pixels. On Sintel, pixels in the dataset's invalid masks are left out of both the visible and the occluded/out-of-frame regions but still count as valid, so the two regions need not cover all valid pixels. Region shares always divide by all valid pixels.

Share of EPE. EPE in the main tables averages errors within each image first, so a share of it must weight each image's errors the same way. For N image pairs, let V_i be the valid pixels of pair i, $O_i \subseteq V_i$ its occluded/out-of-frame pixels, and $e_i(x) = \|\hat{f}_i(x) - f_i^*(x)\|_2$ the endpoint error. With E the EPE over the N pairs, the share of EPE from this region is

$$B_O = \frac{1}{NE} \sum_{i=1}^{N} \frac{\sum_{x \in O_i} e_i(x)}{|V_i|}. \tag{11}$$

For WAFT, $100B_O$ is 43.0% and 42.7% on Things Clean and Final, 52.2% and 42.3% on Sintel Clean and Final, and 51.9% on KITTI. These values give the 42.3–52.2% range in Section 4.2.

B.2 Motion boundaries

Definition. A unit that spans a motion boundary mixes evidence from two differently moving surfaces, which can blur the predicted flow. To test whether DACE keeps such motions apart, we measure EPE near ground-truth motion boundaries. The boundary strength of pixel x is

$$s(x) = \|\Delta_1 f^*(x)\|_2 + \|\Delta_2 f^*(x)\|_2, \tag{12}$$

where Δ_1 and Δ_2 take the flow difference to the next pixel along each image axis; each difference requires valid labels at both pixels. For each threshold of 2, 3, or 4 pixels, pixels whose s exceeds it are boundary pixels, and the region holds the valid pixels in a 5×5 window centered on any of them. Table 2 uses the 2-pixel threshold, and Table 7 reports all three.

DACE is more precise near boundaries at every threshold. Table 7 compares DACE with WAFT, which estimates motion for four times as many units, and with WAFT-H/4 with local attention, which uses a fixed grid at DACE's resolution. DACE has the lowest boundary EPE on every dataset at all three thresholds. DACE is thus more precise near boundaries than a fixed grid with four times as many units, and its advantage holds when the grid points also draw context through local attention.

Table 7: **DACE has the lowest boundary EPE at every threshold.** EPE pooled over valid pixels within two pixels of a ground-truth motion boundary. The threshold sets how large a local flow change must be to count as a boundary. WAFT estimates motion for four times as many units as DACE, and WAFT-H/4 with attention for the same number.

	Things		Sintel		KITTI
System	Clean	Final	Clean	Final	EPE
(a) Boundary threshold: 2 pixels					
WAFT	13.40	13.14	7.30	11.92	9.97
WAFT-H/4 + attention	10.77	10.65	6.97	11.40	11.75
DACE	**9.24**	**9.48**	**6.63**	**10.45**	**8.51**
(b) Boundary threshold: 3 pixels					
WAFT	12.50	12.37	7.38	12.52	12.84
WAFT-H/4 + attention	10.08	10.07	7.02	11.90	15.30
DACE	**8.56**	**8.77**	**6.64**	**11.03**	**10.65**
(c) Boundary threshold: 4 pixels					
WAFT	12.29	12.26	7.75	13.38	15.75
WAFT-H/4 + attention	10.02	9.99	7.34	12.73	18.87
DACE	**8.40**	**8.63**	**6.85**	**11.78**	**12.76**

B.3 Occluded/out-of-frame pixels with large displacements

Setup. A large displacement makes matching harder, because the target lies farther away. We test whether DACE's advantage on occluded/out-of-frame pixels holds when these pixels also move far. Following Sintel's $s40+$ speed category [48], we keep the occluded/out-of-frame pixels whose ground-truth displacement exceeds 40 pixels and apply this cutoff on every dataset.

DACE's advantage holds at large displacements. In Table 8, DACE has lower EPE on these pixels than both WAFT and WAFT-H/4 with local attention on every dataset. DACE's robustness where no match is visible thus also extends to large displacements.

C Qualitative Comparisons

Setup. Figure 6 adds two Sintel Clean pairs and two KITTI pairs from the training splits to Figure 5, using the same zero-shot checkpoints of DACE and the released DINOv3 WAFT. Each method's label gives its EPE on the pair. The token and entropy panels visualize DACE's Image 1 memberships. Ground truth and both predictions share one fixed color mapping within each comparison, so the same flow vector is rendered identically across panels. All flow maps use the standard Middlebury

Table 8: **DACE improves occluded/out-of-frame flow even at large displacements.** EPE pooled over pixels whose target is occluded or out of frame and whose ground-truth displacement exceeds 40 pixels. Pixel share: this subset's percentage of valid pixels.

	Things		Sintel		KITTI
System	Clean	Final	Clean	Final	EPE
WAFT	33.03	31.41	19.19	29.00	15.34
WAFT-H/4 + attention	27.19	25.25	17.25	27.48	15.81
DACE	**23.77**	**22.73**	**16.16**	**24.08**	**12.36**
Pixel share (%)	4.4	4.4	2.7	2.7	12.2

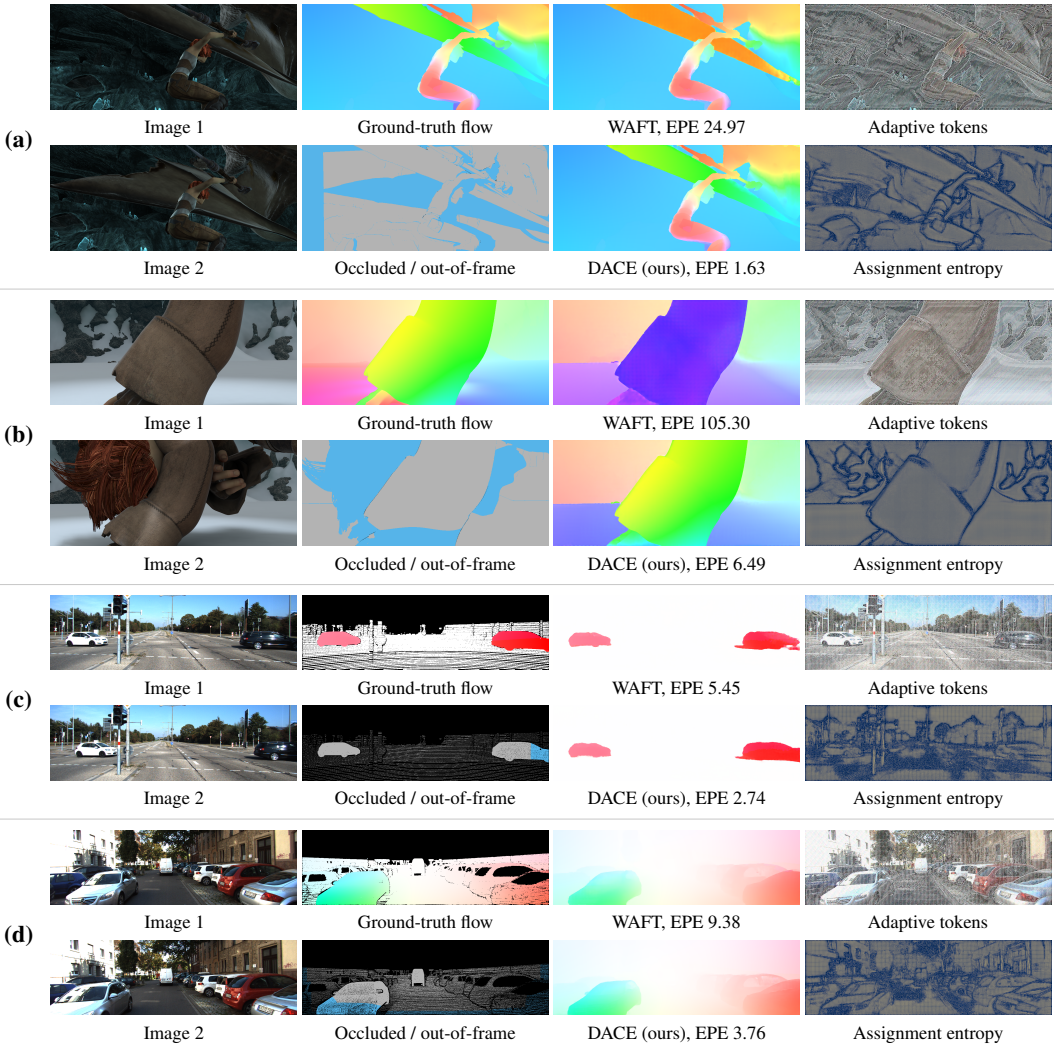


Figure 6: **DACE preserves fine motion detail in additional examples.** (a, b) Sintel Clean and (c, d) KITTI examples using the same zero-shot models as Figure 5. DACE preserves the thin limbs crossing the dragon's wing (a), motion variation across the cloth (b), the vehicle leaving the right edge (c), and car silhouettes near image boundaries (d). Each pair includes its Image 1 token map and assignment entropy, where yellow marks pixels shared among tokens.

color wheel. On Sintel, a flow of magnitude m is drawn at wheel radius $\log(1+m)/6$; on KITTI, the radius grows linearly up to the largest valid ground-truth magnitude.

Reading the figure. In flow maps, hue encodes motion direction and color intensity encodes displacement magnitude; near-white indicates little motion. In the occluded/out-of-frame maps, blue

Table 9: **Four-update DACE outperforms five-update WAFT at lower cost.** Both models are trained with five updates and run here with one to five. DACE-64 halves the recurrent state width.

		Things		Sintel		KITTI		
System	Updates	Clean	Final	Clean	Final	EPE	Fl (%)	MACs (G)
WAFT	1	6.70	6.44	2.20	3.50	8.47	37.1	391.8
DACE	1	5.01	4.84	1.54	2.67	8.30	36.9	433.8
WAFT	2	4.27	4.15	1.49	2.88	4.67	18.5	477.0
DACE	2	2.91	2.84	1.10	2.29	4.09	17.1	524.5
WAFT	3	3.78	3.66	1.34	2.70	3.81	14.0	562.1
DACE	3	2.55	2.49	1.06	2.22	3.38	13.2	615.3
WAFT	4	3.61	3.51	1.29	2.56	3.56	13.1	647.3
DACE	4	2.44	2.43	1.03	2.15	3.18	12.2	706.1
WAFT	5	3.54	3.44	1.28	2.56	3.49	12.9	732.4
DACE	5	2.39	2.38	1.03	2.10	3.11	11.9	796.8
DACE-64	5	2.55	2.48	1.03	2.24	3.52	12.5	636.3

marks these pixels, gray other valid pixels, and black pixels excluded from evaluation. Compare each method's flow with ground truth at the blue pixels. Black in ground-truth flow denotes missing labels, and sparse KITTI labels are enlarged for display only. White contours outline the regions formed by assigning each pixel to its strongest token; DACE itself uses soft memberships. Assignment entropy $H(A)/\log 9 \in [0, 1]$ is dark blue where a pixel belongs mostly to one token and yellow where its membership is spread across tokens. It measures assignment ambiguity, not motion uncertainty.

D Accuracy–Compute Trade-offs

Setup. Table 9 runs the trained DACE and WAFT checkpoints with one to five updates at inference. Both were trained with five updates, and their weights stay fixed across the sweep. DACE-64, which halves the width of the recurrent state (Section 4.5), is evaluated with five updates. MACs are for one 540×960 image pair with model-specific padding. They include feature extraction, token description, all updates, and the upsampling or broadcasting to full resolution.

DACE is more accurate at every number of updates. At each update count, DACE improves all six metrics over WAFT. With four updates, DACE is more accurate than five-update WAFT on every metric at lower cost (706.1 vs. 732.4 GMACs). With three, it is more accurate on five of six metrics, all but KITTI Fl, with 16% fewer MACs. DACE-64 is more accurate than five-update WAFT on five of six metrics, all but KITTI EPE, while costing 13% less in total.

E Target-Specific Fine-Tuning

Table 10 compares DACE, fine-tuned with WAFT's recipe and submitted to the benchmark servers, with WAFT's published DINOv3 results [15]. Scores come from the hidden test sets of Sintel, KITTI 2015, and Spring under each benchmark's own protocol. Unlike in the rest of the paper, these models are also trained on data from the target benchmarks.

Sintel reports EPE. KITTI reports Fl over all labeled pixels (Fl-all) and over non-occluded pixels (Fl-noc). Spring provides four ground-truth values per image pixel and measures each error against the closest one. Its 1px metric is the percentage of pixels with error above one pixel, and WAUC summarizes accuracy across error thresholds. Section 6 discusses these results.

F How Accurately Tokens Represent Motion

Setup. We test how accurately each model's units could represent dense motion if their motion were known. We pool ground-truth flow into DACE's tokens and broadcast it back through the same

Table 10: **After fine-tuning, DACE matches WAFT on Sintel Final but trails it elsewhere.** Benchmark test results for DACE and DINOv3 WAFT. Lower is better except WAUC (↑).

	Sintel		KITTI 2015		Spring			
System	Clean	Final	Fl-all (%)	Fl-noc (%)	1px (%)	EPE	Fl (%)	WAUC ↑
WAFT	0.94	2.02	3.56	2.13	3.18	0.33	1.25	95.05
DACE	1.06	2.02	3.67	2.35	3.43	0.35	1.30	94.86

Table 11: **DACE's tokens represent motion more accurately than a fixed grid.** EPE after pooling ground-truth flow into each model's units and mapping it back to pixels, without retraining. Lower values mean the units can represent the true motion more accurately.

	Things		Sintel		KITTI
System	Clean	Final	Clean	Final	EPE
WAFT-H/4 + attention	0.50	0.53	0.21	0.39	1.52
DACE	**0.19**	**0.21**	**0.09**	**0.19**	**0.39**

memberships (Eqs. (2) and (3)). For WAFT-H/4 with local attention, we average ground-truth flow within each fixed cell and upsample it with its predicted convex weights. Memberships and convex weights are predicted from the images, without retraining. We compare the resulting EPE with that of each model's own predicted flow, reported in Table 3(a).

DACE's tokens could represent motion far more accurately than its current predictions do. In Table 11, DACE's EPE with ground-truth token flow is about 2 to 4 times lower than the fixed grid's on every dataset, so its tokens group pixels whose motions are compatible. This EPE is also about 8 to 13 times lower than that of DACE's predicted flow. Most of DACE's remaining error thus comes from estimating token motion, not from the grouping, so further gains lie in token matching.