

Shape and Texture: What Influences Reliable Optical Flow Estimation?

Libo Long, Xiao Hu, Jochen Lang
 EECS, University of Ottawa

{ llong014, xhu008, jlang}@uottawa.ca

Abstract

Recent methods have made significant progress in optical flow estimation. However, the evaluation of these methods focuses mainly on improved accuracy in benchmarks and often overlooks the analysis of network robustness, which may be important in safety-critical scenarios such as autonomous driving. In this paper, we propose a novel method for robustness evaluation by modifying data from original benchmarks. Unlike previous benchmarks that focus on complex scenes, we propose to modify shape and texture of objects from the original images in order to analyze the sensitivity to these changes observed in the output. Our aim is to identify common failure cases of state-of-the-art (SOTA) methods to evaluate their robustness and understand their behaviors. We show that: Optical flow methods are more sensitive to shape changes than to texture changes; and optical flow methods tend to “remember” objects seen during training and may “ignore” the motion of unseen objects. Our experimental results and findings provide a more in-depth understanding of the behavior of recent optical flow methods. (See <https://github.com/llesky/Robustness-Analysis-Optical-Flow>).

1. Introduction

Optical flow estimation, which aims to estimate pixel-level motion in videos, is fundamental for high-level computer vision applications, such as autonomous driving [5, 24, 37], video editing [2, 16, 18, 29], and 3DGS [10, 48]. Given their widespread use, evaluating the robustness of optical flow methods is crucial considering safety.

To enhance the robustness of optical flow methods, various solutions have been proposed. Some studies have addressed occlusion maps using dedicated networks [14, 23, 30]. Others have incorporated attention mechanisms to improve performance, particularly for large displacements [39]. Recently, studies have begun to replace convolutional neural networks (CNNs) with transformer blocks to further improve accuracy [13, 25, 38]. However, a systematic analysis of the robustness of deep models in optical

flow is still lacking.

Inspired by [8, 15], we start by analyzing how shape and texture affect matching in optical flow. SOTA optical flow methods estimate optical flow by relying on matching pixels (i.e., 4D cost volume) between input images. However, there has been limited study to explain how this matching process works. Motivated by this gap, we introduce Flow-R, a synthetic test set generated by a diffusion model for optical flow tasks. Unlike existing datasets, we modify the shape and texture of a key object (e.g., a car) in the KITTI benchmark [7] and evaluate the changes in the output compared to the original results. This allows us to measure the sensitivity of the method to certain types of semantic change in the input. Through this, we aim to explore two fundamental questions about optical flow methods:

1. Shape vs. Texture: Do optical flow methods estimate motion primarily based on an object’s shape or texture?
2. Memorization: Do these methods tend to memorize objects from previously seen scenes?

Our Flow-R test set includes three subsets: Flow-R-T for texture replacement, Flow-R-S for shape replacement, and Flow-R-O for adding objects changing both shape and texture. Figure 1 presents two samples of data. We develop an application using the Segment Anything Model (SAM) [19] to generate the corresponding masks of the main objects in each pair of images. We then manually select high-quality samples.

To build Flow-R-T, we apply a set of 24 common texture prompts to ControlNet with specific masks to generate new textures for an object. We evaluate the performance of SOTA optical flow methods on Flow-R-T and find that these methods can still match the motion of an object with entirely different textures.

To build Flow-R-S, we directly scale the selected object in the image by a random factor θ and assign an image motion to the object based on its real non-linear motion. We demonstrate that optical flow methods rely more on shape information than texture, and changes of the object flow can significantly affect the optical flow of the background.

To build Flow-R-O, we introduce a set of 13 unseen objects, i.e., objects that do not occur in the training data, and



Figure 1. **Samples of Flow-R.** Original and modified images (Row 1 and 3), ground truth optical flows (Row 2 and 4) are shown. The first column displays the original data from the KITTI dataset. The second column shows the data with the texture of the main object of car replaced. The third column shows the data with the shape of the main object rescaled. The fourth row presents the data with an unseen object added.

22 seen objects. We randomly select and add one of these objects to a KITTI image. By comparing the performance of the optical flow method on images with seen vs. unseen objects, we show that SOTA methods can easily estimate the motion of seen objects but tend to ignore the motion of unseen objects.

2. Related Works

2.1. Optical Flow Estimation

Recent optical flow methods [14, 40, 43, 45] have achieved significant progress in benchmarks [7]. RAFT [41] proposes a 4D cost volume and iterative refinement architecture as a foundational structure for recent methods. CRAFT [39] introduces a cross-attention mechanism to revitalize the cost volume for large displacements. GMFlow [42] proposes directly matching the global motion, then propagating local flow with self-attention. FlowFormer [13] introduces a transformer-based architecture for optical flow estimation. FlowFormer++ [38] enhances the cost volume through masked autoencoding. Recently, SAMFlow [47] embeds the encoder of SAM [19] into FlowFormer to address the fragmentation problem.

However, the primary evaluation of optical flow methods is by measuring performance on benchmarks (e.g., [1]). Common benchmark datasets include synthetic Chairs [4] and FlyingThings3D [26] as well as the computer graphics (CG) based MPI Sintel [3] and real-world KITTI [7, 28].

Further datasets have been produced with a game engine, e.g., Virtual KITTI [6] or with the KUBRIC [9] simulator, or based on a CG movie such as the high-resolution SPRING [27]. Real-world datasets are more challenging to collect. HD1K [20] contains driving scenarios but different from KITTI includes various challenging imaging conditions. None of these datasets are designed to study the sensitivity of an optical flow method to changes in the texture and shape of objects in the scene or the robustness of optical flow to unseen objects. Unlike previous studies, we focus with Flow-R on identifying the common “weaknesses” of SOTA methods.

2.2. Adversarial Attacks in Optical Flow Estimation

To study the robustness of optical flow estimation, many adversarial attack methods have been proposed. Ranjan et al. [31] proposed learning a small patch to attack optical flow methods. Schrodi et al. [36] introduced a less constrained global attack on optical flow, inspired by the Iterative Fast Gradient Sign Method (I-FGSM) [22]. Koren et al. [21] proposed corrupting the flow predictions of specific object categories using a regularization term that promotes consistency in off-target predictions. Schmalfluss et al. [34] proposed the Perturbation-Constrained Flow Attack (PCFA), which emphasizes the destructiveness of adversarial perturbations while maintaining control over their magnitude. They also proposed adversarial weather attacks [35], which simulate weather noise such as snowflakes

using a 3D rendering framework.

Those adversarial attack methods all rely on training and adding perturbations to make the network fail. In contrast, our methods directly add real-world objects instead of perturbations and do not require any training. Most adversarial attack methods aim to explore "worst-case" robustness with specifically designed patches or perturbations, which are unlikely to occur by chance in real-world scenarios. The goal of an adversarial attack is to make the network fail. In contrast, we analyze cases that optical flow methods may face in real-world situations even without an adversary. Our goal is to analyze when and how optical flow methods fail in general scenarios.

3. Methods

3.1. Data Manipulation

Figure 2 indicates the main pipeline of our methods. To construct Flow-R, we utilize the Segment Anything Model (SAM) [19] to identify the corresponding pairs of main objects across images. Given a pair of images I_1, I_2 , we generate a mask with SAM for object x :

$$(S_1, S_2) = \text{SAM}((I_1, I_2), (P_1, P_2)) \quad (1)$$

Where the prompts P_1 and P_2 are sets of indices corresponding to object x in each image. S_1 and S_2 are binary masks for object x in I_1 and I_2 .

Flow-R consists of three parts. Flow-R-T includes images with different textures of the same object, which we use to evaluate the performance of methods with unseen textures. Flow-R-S modifies the shape of the object by scaling by a factor θ . Flow-R-O adds new unseen objects or replaces object x with a new object in I_1 and I_2 .

Texture Replacement (Flow-R-T). We employ a stable diffusion model [32] to generate new textures for object x . The diffusion model is capable of producing diverse textures but our approach remains compatible with other generative models that can be directed through language prompts.

To generate a texture with the same shape as the object, we incorporate ControlNet [46] to guide the generation process. We generate the new texture for I_2 as follows

$$I_N = \text{ControlNet}(N, S_2) \quad (2)$$

where N represents the texture within the region defined by S_2 . S_2 acts as a prompt to focus ControlNet on generating an object with the same shape. The new I_2^N with texture N can be presented as:

$$I_2^N = I_2 \times (1 - S_2) + I_N \times S_2 \quad (3)$$

To preserve consistency of optical flow, we use ground-truth optical flow f_{gt} to warp the new object from I_2^N to

I_1^N .

$$I_1^N = I_1 \times (1 - S_1) + \text{warp}(I_2^N, f_{gt}) \times S_1 \quad (4)$$

Finally, we use Bi-directional Hole Filling [11] to fill the holes caused by occlusion. The optical flow remains the same as the original. The new generated data is (I_1^N, I_2^N, f_{gt}) .

Shape Replacement (Flow-R-S). We scale the object shape x by a factor θ enabling the use of its real motion instead of adding unreal motion as in [26]. Specifically, we directly upscale the object x in I_1 and in I_2 with a scale factor $\theta > 1$. Then, we paste it back aligned with the center of its own respective bounding box in the image.

$$I_1^x = I_1 \times S_1 \quad (5)$$

The center point (c_1^x, c_1^y) of the bounding box of x in I_1^x can be easily calculated. The new image I_1^s can be presented as:

$$I_1^{x\uparrow} = \text{Upsample}(I_1^x, \theta) \quad (6)$$

$$I_1^s = \text{Paste}(I_1, I_1^{x\uparrow}, (c_1^x, c_1^y)) \quad (7)$$

Where Paste() is used to paste the scaled x back to the position (c_1^x, c_1^y) in I_1 . Then, by using the ground truth optical flow, we can trace the center position of the object in I_2 . Using the same process, we get I_2^s .

Given the center position (c_1^x, c_1^y) for x in I_1 and (c_2^x, c_2^y) in I_2 , the optical flow needs to be recalculated in the object area. We use Eqs. (5) to (7) to get the upscaled x optical flow $f_{gt}^{x\uparrow}$. Then, for all pixels in $x \uparrow$, the flow can be updated by:

$$f_{ij}^s = \theta f_{ij} + (\theta - 1)(c_1^x - c_2^x, c_1^y - c_2^y), f_{ij} \in x \uparrow \quad (8)$$

Where f_{ij} contains the optical flow for the 2D offset at index i, j (The detailed inference process is described in the supplemental material).

For $\theta < 1$, we first remove the object using [44], then rescale the object $x \downarrow$. However, the motion of the background covered by the original object $x - x \downarrow$ is unknown. Due to the unknown background motion, we mask pixels for $x - x \downarrow$ in the optical flow evaluation.

Adding Objects (Flow-R-O). We collect real-world image I_M of unseen objects. Then, we get new objects y from I_n using the SAM model with prompt P_M

$$S_M = \text{SAM}(I_M, P_M) \quad (9)$$

$$y = I_M * S_M \quad (10)$$

We paste y in I_2 and move it to I_1 using an affine transform, then recalculate the optical flow f_{aff} . The new data is (I_1^y, I_2^y, f_{aff}) .

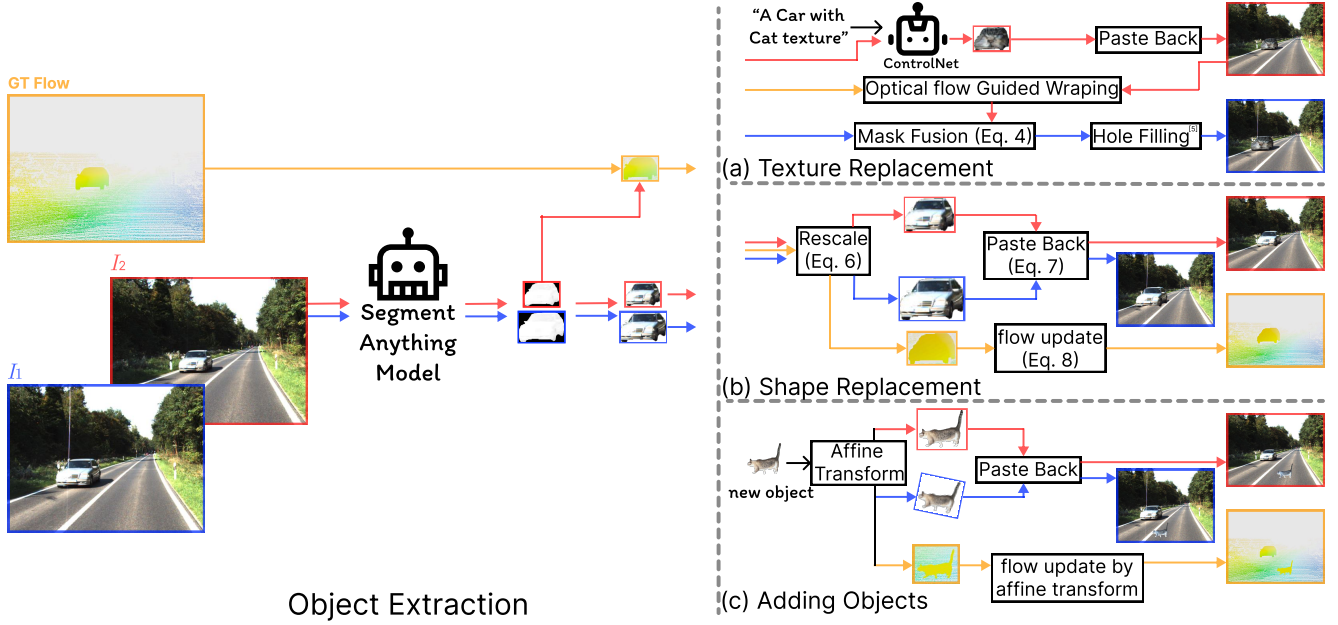


Figure 2. **Pipeline of our methods.** Given the data I_1, I_2, f_{gt} from KITTI, we manipulate the data by changing the texture or shape of an object or adding new objects.

Manual Quality Control. The above process will automatically generate new test data. However, SAM and generative models can produce poor quality masks and images that do not match the object x . To ensure the masks and images are correct and of high quality, we resort to human checking.

3.2. Robustness Evaluation

We are inspired by sensitivity analysis [33] and evaluate the robustness for an image pair. We treat texture, shape and new objects as variables, assume a model could accurately predict flow by given I_1, I_2 , we have

$$f_p = \text{Model}(I_1, I_2) \approx f_{gt} \quad (11)$$

where f_{gt} is the ground truth flow for I_1, I_2 . If we manipulate I_1, I_2 by replacing the texture of an object to obtain I_1^t, I_2^t , and predict the flow by

$$f_p^t = \text{Model}(I_1^t, I_2^t) \quad (12)$$

We can evaluate the sensitivity by

$$|f_p - f_p^t| \approx |f_{gt} - f_p^t| \quad (13)$$

This approach allows us to easily test the robustness and effects of different textures, shapes, and new objects. In practice, we primarily test methods with weights fine-tuned on the KITTI dataset that achieve high accuracy (< 0.7 EPE).

4. Experiments

In this section, we describe the experiments that evaluate the behaviors of state-of-the-art (SOTA) optical flow approaches with our proposed method. We focus primarily on the common behaviors and failure cases of methods. These failure cases can potentially guide future studies of optical flow methods.

We mainly use black-box evaluation to explore two important questions: *Texture vs. Shape*: What do optical flow methods rely on for matching? *Memorization*: Do optical flow methods tend to remember objects?

In this experiment, we select the following methods: RAFT [41], GMA [17], GMFlow [42], FlowFormer [13], FlowFormer++ [38] and SAMFlow [47].

4.1. Texture vs. Shape

Understanding how networks match objects is an important problem. It may provide fundamental insights into the functioning and potential failure points of optical flow methods. However, to the best of our knowledge, no studies have explored this aspect. In this section, we conduct three experiments to investigate whether the selected optical flow methods rely on texture or shape.

To investigate the dependence of optical flow methods on texture versus shape, we evaluated their performance using KITTI [7], Flow-R-T, and Flow-R-S datasets as shown in Table 1. The findings reveal that most methods yield comparable results on KITTI [7] and Flow-R-T datasets. In



Figure 3. **Texture Attack.** The texture of the main object (car) in the first frame is replaced, while the second frame remains unchanged. The resulting optical flow is similar to that derived from the original images.

Training Data	Method	KITTI-15		Flow-R-T		Flow-R-S		R-S ($\theta < 1$)
		AEPE	Fl-all (%)	AEPE	Fl-all (%)	AEPE	Fl-all (%)	AEPE
C + T	RAFT[41]	5.04	17.4	5.18	18.3	5.97	24.9	6.02*
	GMFlow[42]	7.77	23.4	8.67	26.0	8.91	31.0	-
	GMA[17]	4.69	17.1	4.82	17.9	5.77	24.6	5.64*
	Flowformer[13]	4.09	14.7	4.28	15.5	5.19	22.3	5.13*
	Flowformer++[38]	3.93	14.1	4.30	15.0	5.07	21.8	4.97*
	SAMFlow[47]	3.44	12.3	3.58	13.0	5.01	20.6	-
C + T + S/K (+ H)	RAFT[41]	0.63	1.5	0.73	1.66	1.96	10.2	1.83*
	GMFlow[42]	1.36	5.2	1.55	6.2	2.78	13.9	-
	GMA[17]	0.57	1.2	0.67	1.4	1.90	10.0	1.72*
	Flowformer[13]	0.53	1.11	0.64	1.26	1.89	9.9	1.77*
	Flowformer++[38]	0.57	1.16	0.71	1.55	1.95	10.1	1.74*

Table 1. **Quantitative results on KITTI 2015 and Flow-R datasets.** We report the average end-point error (AEPE) and the Fl-all measure (Percentage of optical flow outliers with an error larger than 3 pixels). Methods are pre-trained on the Chairs and Things datasets "C + T" or fine-tuned on the Sintel and KITTI datasets, with some also fine-tuned on the HD1K dataset "S/K (+ H)". In Flow-R-T the texture of the main object is replaced, and Flow-R-S replaces the shape of main object. Column R-S ($\theta < 1$) replaces the shape with a smaller object. Values marked with * exclude the unknown area. All data is generated from KITTI 2015.

Attack	Texture		Shape	
	AEPE	F1-all	AEPE	F1-all
RAFT [41]	0.82	1.99	2.21	10.5
GMFlow[42]	1.63	6.5	3.03	14.2
GMA[17]	0.72	1.63	2.09	10.5
Flowformer[13]	0.64	1.26	2.04	10.2
Flowformer++[38]	0.76	1.66	2.13	10.1

Table 2. **Texture attack and Shape attack.** Quantitative results of the attacked first frame and original second frame.

contrast, all methods demonstrate a notable increase in errors on the Flow-R-S dataset, suggesting a higher sensitivity of optical flow methods to shape variations compared to texture variations. We separately report shape replacement

with $\theta < 1$ in column R-S ($\theta < 1$) because the reduction in object size may also introduce errors by incorrect filling of background flow.

Texture Attack. Given that texture replacement does not significantly impact the performance of SOTA methods, we conduct an additional experiment by altering the texture of only one image. Given data from the KITTI dataset ($I_1, I_2, f_g t$), we replace the texture of object x from I_2 by Equation (3). Despite the texture difference, the object is still matched, and the optical flow is "accurately" predicted (see Figure 3). Table 2 presents the results of the Texture Attack experiment. Surprisingly, we observe that the error rate for texture attack of attention-based methods is relatively small compared with errors in KITTI, suggesting that the matching of objects in optical flow methods is less dependent on texture than shape.



Figure 4. **A failed example.** We add seen and unseen objects to a similar area. The first row indicates that all methods successfully predict the motion of the added seen (car) objects. The second row indicates that all SOTA methods failed to predict the unseen (fox) objects.

Methods	Add Seen				Add Unseen			
	all		foreground	background	all		foreground	background
	AEPE	F1-all	AEPE	AEPE	AEPE	F1-all	AEPE	AEPE
PWC-Net[40]	4.16	14.3	13.22	1.83	12.7	28.4	34.30	13.1
RAFT[41]	2.54	8.2	11.26	0.77	7.43	20.4	25.68	7.18
GMFlow[42]	3.24	12.1	11.32	1.64	9.42	22.6	23.73	9.03
GMA[17]	2.55	8.1	10.89	0.81	7.44	20.2	22.62	7.19
FlowFormer[13]	2.45	7.7	10.51	0.77	7.81	21.0	23.70	7.52
FlowFormer++[38]	2.55	8.1	10.71	0.83	7.50	20.4	20.53	7.26
SAMFlow [47]	2.43	7.9	10.76	0.85	7.73	20.1	20.57	7.33

Table 3. **Adding new objects.** Quantitative results of adding new objects to original images. “Add Seen” indicates the data where we add KITTI objects to the images. “Add Unseen” refers to the data where we add real-world objects not included in KITTI. Foreground indicates the EPE errors in the area of the new object. Background indicates the EPE error that do not include the new object.

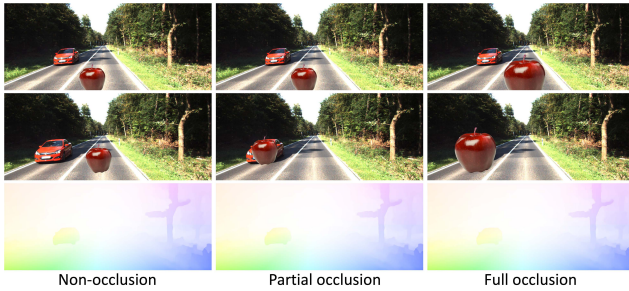


Figure 5. **Failed example with different positions.** The unseen object (apple) is non-occluded, partially occluded, or fully occluded relative to the original object (car). The predicted flow from FlowFormer++ shows similar results; all flows ignore the unseen objects. The third column shows that the original motion of the car is still predicted even when the car is fully covered.

Shape Attack. Shape changes are a common challenge in optical flow estimation, particularly when objects move closer to or further away from the camera. Hence, to assess the robustness of models to this kind of shape changes, we

design a shape attack experiment, where we scale the object in only one image.

Specifically, we scale the first frame I_1 by θ (as described by Equation (7)), and recalculate the optical flow (details provided in the supplementary material). Table 2 presents the quantitative evaluations of state-of-the-art methods. We observe a significant increase in errors between the shape attack results and the original results, indicating that the matching of objects in optical flow methods is sensitive to shape changes. More experiments with Sintel [3] and Shape Replacement can be found in the supplementary material.

4.2. Does the Network Remember Objects?

Optical flow methods still suffer from poor cross-dataset generalization. Most SOTA methods require fine-tuning and testing on the same dataset. One hypothesis is that optical flow methods tend to memorize objects during training and attempt to find similar objects during testing. In this section, we investigate this hypothesis experimentally.

We define an unseen object as one that the methods do not see during training, while a seen object is one that is included in the KITTI dataset. Assuming opti-

cal flow methods successfully predict the correct optical flow given data (I_1, I_2, f_{gt}) , using Equations (9) and (10), we add unseen objects from real-world images to obtain $(I_1^{unseen}, I_2^{unseen}, f_{gt}^{unseen})$ and add seen objects from KITTI in another scene to get $(I_1^{seen}, I_2^{seen}, f_{gt}^{seen})$. By comparing the results in Table 3 for data with unseen and seen objects, we observe that flow prediction is significantly less accurate for unseen objects. This suggests that unseen objects negatively impact the methods. Furthermore, Table 3 reports the error in the foreground (EPE error only in the new object area) and the background error (areas not including the new object). We find that adding unseen objects also affects the background accuracy, even if the unseen object is not adjacent to the background area.

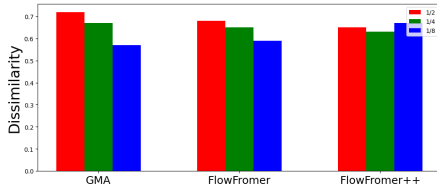


Figure 6. **Feature Dissimilarity.** All levels in the encoder have high dissimilarity compared with the original features for the added object region, indicating the encoder retains the new object.

4.3. Failures for Unseen Objects

To further study why unseen objects lead to poor results, we visualize the results in Section 4.2. We find that SOTA methods successfully predict the flow for seen objects, i.e., cars, but usually predict worse or even ignore unseen objects in the results. For example, in Figure 4, we add a car (first row) and a fox (second row) in similar positions. All methods successfully predict the motion of the car, but all methods fail to predict the motion of the fox. This indicates that SOTA methods tend to reduce or even ignore the flow of unseen objects.

Unseen object attack. To study how an unseen object x_u affects objects x in original images, we place x_u in different positions to observe its impact on the motion of x in the images. Specifically, we categorize the scenarios into three cases: unoccluded, where x_u does not cover x ; partially occluded, where x_u partially covers x ; and fully occluded, where x_u completely covers x in the target image.

Figure 5 presents an example using FlowFormer++[38]. In the first column, the apple does not interact with the car. In the second column, the apple partially covers the car. In the third column, the apple completely covers the car. However, the flow results for all three scenarios ignore the motion of the apple and successfully predict the car and its

flow, even when it is covered by the apple. In the third example, where the car is fully covered by the apple, the cost volume should indicate that the car in the first frame cannot be matched to the second image due to occlusion. However, the method still predicts the car’s motion, similar to the original results. This indicates that the methods tend to remember the motion of the scene or object and produce similar output.

5. A Study of Unseen Objects

To investigate why certain methods ignore unseen objects, we collect failure cases where the object is ignored and visualize the encoder and decoder components of GMA [17], FlowFormer [25], and FlowFormer++ [38]. As GMA is primarily based on a CNN architecture, FlowFormer on a transformer architecture and FlowFormer++ incorporates an Masked AutoEncoder [12] pretraining strategy, it allows us to investigate whether these designs contribute to robustness.



Figure 7. **Example of Feature Dissimilarity.** A clear difference of encoder features with or without new objects is shown at each level. The encoder retains the information related to new objects.

5.1. Feature Visualization in the Encoder

In the encoder, we extract the contextual features from the original image I_1 and the image containing the unseen object I_1^y . For each feature at level (1/2, 1/4, and 1/8), we treat each pixel as a vector and perform cross-similarity calculations between the original features and unseen object features to generate a similarity map, as illustrated in Figure 7 where 1/2, 1/4 and 1/8, respectively, refer to the image resolution. The results reveal a clear difference in the areas corresponding to the new object, indicating that the encoder successfully extracts features related to the added object at all levels rather than ignoring it.

In Figure 6, we statistically analyze the dissimilarity of features in the various methods. We measure the dissimilarity by calculating the similarity score, normalizing the score to a range of 0 to 1 and inverting it. Higher values indicate greater dissimilarity between two sets of features. We only evaluate the dissimilarity within the new object region. We found that at all three levels (1/2, 1/4, and 1/8), the features exhibited a relatively high dissimilarity, suggesting that the encoder effectively retains information about the unseen object. The feature difference between the original image and the image with the apple using cross-similarity in GMA is visualized in Figure 7. It clearly indicates the dissimilarity



Figure 8. **Visualization of Intermediate Flows in the Decoder.** Decoder GRU iterations in original and failed samples.

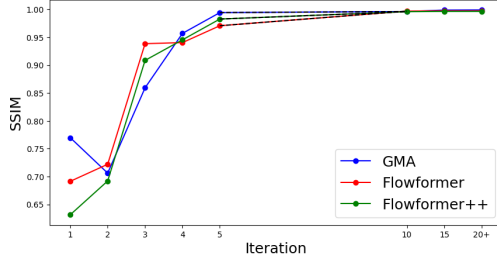


Figure 9. SSIM of intermediate flow in each iteration between failed and original samples.

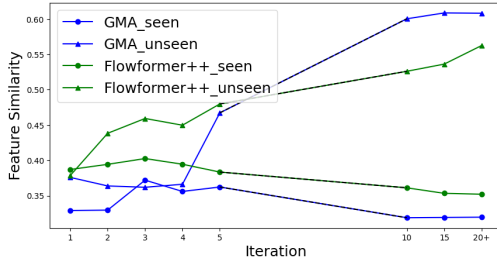


Figure 10. **Feature similarity of GRU block.** A higher value indicates that the features of the image with the added object are similar to those of the original image.

in the apple region.

5.2. Feature Visualization in the Decoder

Recent methods employ an iterative refinement architecture. We chose GMA as a representative for such a framework and visualize each iterative step of the optical flow decoder. As illustrated in Figure 8, we initially observe the motion of an unseen object (e.g., an apple) during the coarse optical flow estimation. This object is subsequently treated as noise and removed in later iterations.

We use SSIM to quantitatively assess the structural similarity between the optical flow of the original input images I_1, I_2 and the images that contain the unseen object I_1^y, I_2^y . The evaluation is only for the region of the new object. As depicted in Figure 9, we observe significant differences in

SSIM values during the first four iterations but much closer similarity after ten iterations. As illustrated in Figure 8, the first row displays the initial five iterations of optical flow applied to the original image input, while the second row presents the first five iterations for an image with an unseen object (an apple). The apple is distinctly visible in the first iteration, but visibility decreases with each subsequent iteration, and becomes nearly imperceptible by the fourth iteration. This observation suggests that the decoder component may accidentally interpret the unseen object as noise or error, leading to its gradual removal throughout the iterations.

Next, we conduct experiments to analyze each main component of the decoder, including the cost memory, attention block, GRU block, and upsampler. By comparing the features extracted from I_1, I_2 and I_1^y, I_2^y , we observe that the GRU block exhibits behavior similar to that observed in the flow iterations. Figure 10 reports the similarity between the original features and the features with new objects within the GRU block. Circles indicate images with seen objects, while triangles denote images with unseen objects. In both GMA and Flowformer++, the GRU block effectively retains the features of seen objects across iterations, while it weakens the features of unseen objects. This suggests that the GRU block is designed to remember and preserve the characteristics of seen objects, while attempting to eliminate the influence of unseen objects during the iterations.

6. Conclusion

In this paper, we propose a robustness evaluation method by manipulating the shape and texture of original objects or by adding new objects. We generate a dataset and conduct a comparative evaluation of the robustness of recent SOTA methods. Our findings reveal two notable common behaviors among these SOTA methods: 1. The matching performance of optical flow methods depends more on shape than on texture. 2. The GRU block tends to remember seen objects while attempting to diminish the influence of unseen objects. Given these observations, we express concerns regarding the safe usage of current optical flow methods. We hope that our methodology and findings will guide further studies aimed at enhancing the robustness of optical flow techniques in safety-critical scenarios.

References

- [1] Simon Baker, Daniel Scharstein, J. P. Lewis, Stefan Roth, Michael J. Black, and Richard Szeliski. A database and evaluation methodology for optical flow. In *International Journal of Computer Vision*, pages 1–8, 2007.
- [2] Wenbo Bao, Wei-Sheng Lai, Chao Ma, Xiaoyun Zhang, Zhiyong Gao, and Ming-Hsuan Yang. Depth-aware video frame interpolation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3698–3707, 2019.
- [3] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black. A naturalistic open source movie for optical flow evaluation. In *European Conference on Computer Vision*, pages 611–625, 2012.
- [4] A. Dosovitskiy, P. Fischer, E. Ilg, P. Häusser, C. Hazırbaş, V. Golkov, P. v.d. Smagt, D. Cremers, and T. Brox. FlowNet: Learning optical flow with convolutional networks. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2758–2766, 2015.
- [5] Yi Feng, Ruge Zhang, Jiayuan Du, Qijun Chen, and Rui Fan. Freespace optical flow modeling for automated driving. *IEEE/ASME Transactions on Mechatronics*, pages 1511–1520, 2024.
- [6] Adrien Gaidon, Qiao Wang, Yohann Cabon, and Eleonora Vig. Virtualworlds as proxy for multi-object tracking analysis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4340–4349. IEEE Computer Society, 2016.
- [7] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3354–3361, 2012.
- [8] Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A Wichmann, and Wieland Brendel. Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness. In *International Conference on Learning Representations*, 2019.
- [9] Klaus Greff, Francois Belletti, Lucas Beyer, Carl Doersch, Yilun Du, Daniel Duckworth, David J. Fleet, Dan Gnanaprasam, Florian Golemo, Charles Herrmann, Thomas Kipf, Abhijit Kundu, Dmitry Lagun, Issam H. Laradji, Hsueh-Ti Derek Liu, Henning Meyer, Yishu Miao, Derek Nowrouzezahrai, A. Cengiz Öztireli, Etienne Pot, Noha Radwan, Daniel Rebain, Sara Sabour, Mehdi S. M. Sajjadi, Matan Sela, Vincent Sitzmann, Austin Stone, Deqing Sun, Suhani Vora, Ziyu Wang, Tianhao Wu, Kwang Moo Yi, Fangcheng Zhong, and Andrea Tagliasacchi. Kubric: A scalable dataset generator. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3739–3751, 2022.
- [10] Zhiyang Guo, Wengang Zhou, Li Li, Min Wang, and Houqiang Li. Motion-aware 3d gaussian splatting for efficient dynamic scene reconstruction. *arXiv preprint arXiv:2403.11447*, 2024.
- [11] Yunhui Han, Kunming Luo, Ao Luo, Jiangyu Liu, Haoqiang Fan, Guiming Luo, and Shuaicheng Liu. Realflow: Embased realistic optical flow dataset generation from videos. In *Proceedings of the European Conference on Computer Vision*, pages 288–305, 2022.
- [12] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16000–16009. IEEE, 2022.
- [13] 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. *European Conference on Computer Vision*, 2022.
- [14] Junhwa Hur and Stefan Roth. Iterative residual refinement for joint optical flow and occlusion estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, page 4761–4770, 2019.
- [15] Md Amirul Islam, Matthew Kowal, Patrick Esser, Sen Jia, Bjorn Ommer, Konstantinos G Derpanis, and Neil Bruce. Shape or texture: Understanding discriminative features in cnns. In *International Conference on Learning Representations*, 2021.
- [16] Huaizu Jiang, Deqing Sun, Varan Jampani, Ming-Hsuan Yang, Erik Learned-Miller, and Jan Kautz. Super SloMo: High quality estimation of multiple intermediate frames for video interpolation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9000–9008, 2018.
- [17] 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.
- [18] Soo Ye Kim, Jihyong Oh, and Munchurl Kim. FISR: Deep joint frame interpolation and super-resolution with a multi-scale temporal loss. In *AAAI*, pages 11278–11286, 2020.
- [19] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything. *arXiv:2304.02643*, 2023.
- [20] Daniel Kondermann, Rahul Nair, Katrin Honauer, Karsten Krispin, Jonas Andrusis, Alexander Brock, Burkhard Gussefeld, Mohsen Rahimimoghaddam, Sabine Hofmann, Claus Brenner, and Bernd Jahne. The HCI benchmark suite: Stereo and flow ground truth with uncertainties for urban autonomous driving. In *CVPRW*, pages 19–28, 2016.
- [21] Tom Koren, Lior Talker, Michael Dinerstein, and Roy J. Jevnisek. Consistent semantic attacks on optical flow. In *Proceedings of the Asian Conference on Computer Vision (ACCV)*, pages 1658–1674, 2022.
- [22] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial examples in the physical world. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2017.
- [23] Jianfeng Li, Junqiao Zhao, Shuangfu Song, and Tiantian Feng. Occlusion aware unsupervised learning of optical flow from video. In *Thirteenth International Conference on Machine Vision*, page 224 – 231, Rome, Italy, 2021. International Society for Optics and Photonics, SPIE.

- [24] Tianran Liu, Morteza Mousa Pasandi, and Robert Laganieri. Fsmdet: Vision-guided feature diffusion for fully sparse 3d detector. *arXiv preprint arXiv:2409.06945*, 2024.
- [25] Yawen Lu, Qifan Wang, Siqi Ma, Tong Geng, Yingjie Victor Chen, Huaijin Chen, and Dongfang Liu. Transflow: Trans-former as flow learner. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18063–18072, 2023.
- [26] 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 *CVPR*, pages 4040–4048, 2016.
- [27] Lukas Mehl, Jenny Schmalfuss, Azin Jahedi, Yaroslava Nali-vayko, and Andrés Bruhn. Spring: A high-resolution high-detail dataset and benchmark for scene flow, optical flow and stereo. In *CVPR*, 2023.
- [28] Moritz Menze and Andreas Geiger. Object scene flow for autonomous vehicles. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3061–3070, 2015.
- [29] Simon Niklaus, Long Mai, and Feng Liu. Video frame interpolation via adaptive convolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2270–2279, 2017.
- [30] Bojun Ouyang and Dan Raviv. Occlusion guided scene flow estimation on 3d point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1049–1058, 2021.
- [31] Anurag Ranjan, Joel Janai, Andreas Geiger, and Michael J. Black. Attacking optical flow. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1–10, 2019.
- [32] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10807–10817. IEEE, 2022.
- [33] Andrea Saltelli, Marco Ratto, Terry Andres, Francesca Campolongo, Jessica Cariboni, Debora Gatelli, Michaela Saisana, and Stefano Tarantola. *Global Sensitivity Analysis: The Primer*. John Wiley & Sons, 2008.
- [34] Jenny Schmalfuss, Philipp Scholze, and Andrés Bruhn. A perturbation-constrained adversarial attack for evaluating the robustness of optical flow. *Proceedings of the European Conference on Computer Vision*, 2022.
- [35] Jenny Schmalfuss, Lukas Mehl, and Andrés Bruhn. Adversarial weather attacks for motion estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10106–10116, 2023.
- [36] Philipp Schrodi, Jenny Schmalfuss, and Andrés Bruhn. A less constrained global attack on optical flow. In *Proceedings of the European Conference on Computer Vision*, pages 183–200. Springer, 2022.
- [37] Hao Shi, Yifan Zhou, Kailun Yang, Xiaoting Yin, and Kai-wei Wang. Csfow: Learning optical flow via cross strip correlation for autonomous driving. In *2022 IEEE Intelligent Vehicles Symposium (IV)*, pages 1851–1858, 2022.
- [38] Xiaoyu Shi, Zhaoyang Huang, Dasong Li, Manyuan Zhang, Ka Chun Cheung, Simon See, Hongwei Qin, Jifeng Dai, and Hongsheng Li. Flowformer++: Masked cost volume autoen-coding for pretraining optical flow estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1599–1610, 2023.
- [39] 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 123–132, 2022.
- [40] Deqing Sun, Xiaodong Yang, Ming-Yu Liu, and Jan Kautz. PWC-Net: CNNs for optical flow using pyramid, warping, and cost volume. In *CVPR*, page 8934–8943, Salt Lake City, UT, USA, 2018.
- [41] Zachary Teed and Jia Deng. RAFT: Recurrent All-Pairs Field Transforms for Optical Flow. In *European Conference on Computer Vision*, page 402–419, Glasgow, UK, 2020.
- [42] Haofei Xu, Jing Zhang, Jianfei Cai, Hamid Rezaatofghi, and Dacheng Tao. Gmflow: Learning optical flow via global matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 795–804, 2022.
- [43] Gengshan Yang and Deva Ramanan. Volumetric correspon-dence networks for optical flow. In *Advances in Neural In-formation Processing Systems*, pages 794–805, 2019.
- [44] Tao Yu, Runsen Feng, Ruoyu Feng, Jinming Liu, Xin Jin, Wenjun Zeng, and Zhibo Chen. Inpaint anything: Segment anything meets image inpainting. *arXiv preprint arXiv:2304.06790*, 2023.
- [45] Feihu Zhang, Oliver J. Woodford, Victor Adrian Prisacariu, and Philip H. S. Torr. Separable flow: Learning motion cost volumes for optical flow estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10807–10817, 2021.
- [46] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3836–3847, 2023.
- [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*, 2024.
- [48] Ruijie Zhu, Yanzhe Liang, Hanzhi Chang, Jiacheng Deng, Jiahao Lu, Wenfei Yang, Tianzhu Zhang, and Yongdong Zhang. Motiongs: Exploring explicit motion guidance for deformable 3d gaussian splatting. In *Conference on Neural Information Processing Systems*, 2024.