Learning Optical Expansion from Scale Matching

Han Ling[1] Yinghui Sun[1] Quansen Sun[1*] Zhenwen Ren [1,2,3†]
[1]Nanjing University of Science and Technology
[2]Southwest University of Science and Technology
[3]SongShan Laboratory
{321106010190,yinghuisun,sunquansen,rzw}@njust.edu.cn

Abstract

This paper address the problem of optical expansion (OE). OE describes the object scale change between two frames, widely used in monocular 3D vision tasks. Previous methods estimate optical expansion mainly from optical flow results, but this two-stage architecture makes their results limited by the accuracy of optical flow and less robust. To solve these problems, we propose the concept of 3D optical flow by integrating optical expansion into the 2D optical flow, which is implemented by a plug-and-play module, namely TPCV. TPCV implements matching features at the correct location and scale, thus allowing the simultaneous optimization of optical flow and optical expansion tasks. Experimentally, we apply TPCV to the RAFT optical flow baseline. Experimental results show that the baseline optical flow performance is substantially improved. Moreover, we apply the optical flow and optical expansion results to various dynamic 3D vision tasks, including motion-in-depth, time-to-collision, and scene flow, often achieving significant improvement over the prior SOTA. Code is available at https://github.com/HanLingsgjk/TPCV.

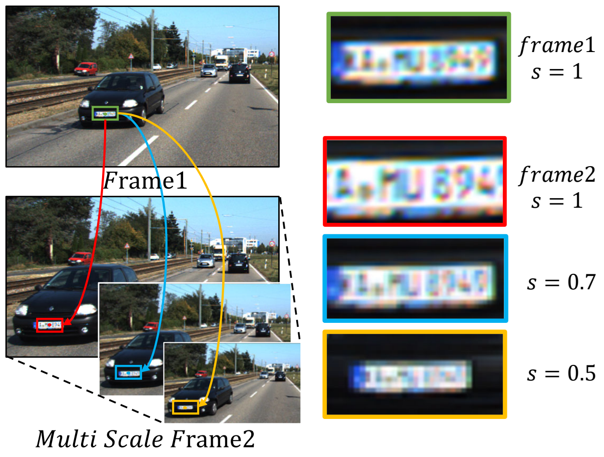


Figure 1. **Scale matching idea.** Left: multi-scale matching between two frames. Right: texture around the matching point, where s is the size of the image scaling. The core idea of scale matching is to match texture features at the correct location and scale. As seen above, the texture at the license plate can be better matched when the second frame is scaled 0.7 times in size, where the scaling magnification also reflects the optical expansion of that local pixel between the two frames. Furthermore, scale matching can better handle the motion in the depth direction and contains potential 3D motion information.

1. Introduction

Optical expansion (OE) is a fundamental and important concept in monocular dynamic 3D vision tasks [1, 3, 22, 23, 32]. OE describes the scale change of an object between two frames, which can be translated into motion in the depth direction. It has essential applications in time-to-collision, scene flow, and motion-in-depth estimation. OE schemes have unique advantages in 3D motion estimation tasks, requiring only a single camera and enabling dense and fixed baseline independent results. In this paper, we discuss a robust and novel approach for OE estimation.

Prior works In early time-to-collision (TTC) studies [4, 22, 23], OE was obtained from motion modeling, where the motion estimation was provided by optical flow or SIFT [10]. Such algorithms relied on optical flow results and specific model assumptions, often yielding only sparse and low-accuracy results. Some recent methods [32] regress OE based on existing optical flow results and achieve better outcomes. However, these two-stage estimation methods rely on accurate optical flow results and decrease computational efficiency. **Instead, we consider optical flow and expansion estimation as two complementary tasks.** Introducing OE in optical flow can realize matching features at the correct location and scale. As shown in Fig. 1, this fusion

*Corresponding author
†Corresponding author

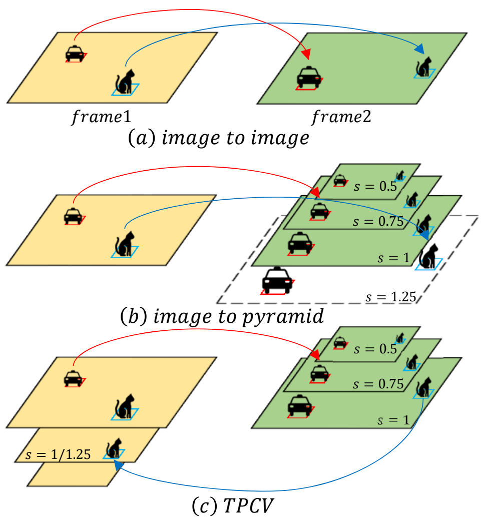


Figure 2. **Three different matching modes**. We match objects between two consecutive frames, where the cat is away from the camera and the car is close to the camera. (a) The 2D optical flow matches the cat and car in the original size image. (b) Match the second frame with the first frame after multiscale scaling, we found that cat can be better matched when magnified by 1.25 times, and car can be better matched when shrunk by 0.75 times. However, obtaining an accurate enlarged picture of a cat is impossible. (c) Transpose matching, where the cat zoomed in in frame 1 to 2 is equivalent to the cat zoomed out in frame 2 to 1. The core of TPCV is to convert the problem of zooming in the forward optical flow to zooming out in the reverse flow.

eliminates the effect of scale variation of moving objects on optical flow matching, allowing the algorithm to predict moving foreground objects more accurately. Specifically, we refer to the fused task as 3D optical flow, and the 3D optical flow consists of a 2D optical flow field $\boldsymbol{f}$ and a 1D scale flow field $\boldsymbol{s}$, where $\boldsymbol{s}$ describes the scale change of objects between two frames.

Solving for 3D Optical Flow We view solving 3D optical flow as a scale matching problem—matching the feature at the right location and scale. The matching idea has been applied in many successful optical flow methods [5, 27, 31, 36], where they iteratively update the optical flow by constructing a correlation volume, querying the correlation of pixel pairs in the correlation volume based on the current estimation. In this paper, we propose a plug-and-play upgrade module: TPCV, a multi-scale correlation volume from which the algorithm can query the correlation between pixel pairs at different locations and scales. We show that 3D optical flow can be robustly extracted from the correlation features provided by TPCV. Importantly, TPCV can be easily applied to matching-based optical flow methods or 3D visual tasks, increasing the performance.

Why TPCV? The computation of 3D optical flow requires computing pixel pairs correlation between feature maps at different scales. As shown in Fig. 2, firstly, we tried to directly scale the image for matching, but this did not present good results; the main reason is the unrealistic large-scale features. So we rethink the problem in reverse, where the size reduction in the forward optical flow is equivalent to the enlargement in the inverse optical flow. We convert the part that needs to be enlarged to the reduction part in the inverse flow, and combine the forward and inverse parts to form a transposed correlation volume (TPCV) to query the correlation of feature pairs.

TPCV Application In order to evaluate the performance of TPCV, we applied it to the classic and basic matched optical flow framework RAFT [27]. For upgrading 2D optical flow to 3D optical flow, only need to replace the original 4D correlation volume in RAFT with TPCV and add an output head of OE in the GRU iterative optimization layer. The relative OE and optical flow ground truth for training are obtained from the existing 3D scene flow training datasets [15, 19].

Contributions We summarize our contributions as follows:

- We propose a plug-and-play module: TPCV, which can be applied to matching-based optical flow framework. Based on TPCV, the original 2D optical flow can be easily upgraded to 3D optical flow, improving accuracy.
- We applied TPCV to the optical flow framework RAFT, which significantly improved the performance without changing the optimizer structure, especially on the moving foreground target.
- We display the effectiveness of TPCV across a variety of benchmark tasks, establishing new SOTA results for scene flow, time-to-collision (TTC) and motion-in-depth estimation - while maintaining a competitive speed.

2. Relate works

Optical Flow The main task of optical flow is to estimate the dense pixel displacement between two frames. Recent matching-based methods have achieved great success [5, 25, 31, 36]. They query the correlation between pixel pairs through a 4D correlation volume for subsequent matching and regression. In these matching methods, the calculation of correlation is the algorithmic core. However, all of these methods ignore the correlation mutation caused by the scale change when the object moves in depth, resulting in the incapacity to handle the foreground moving objects. In this paper, we upgrade traditional image-to-image feature matching to image-to-scale-space matching based on scale matching. which allows our method to deal with

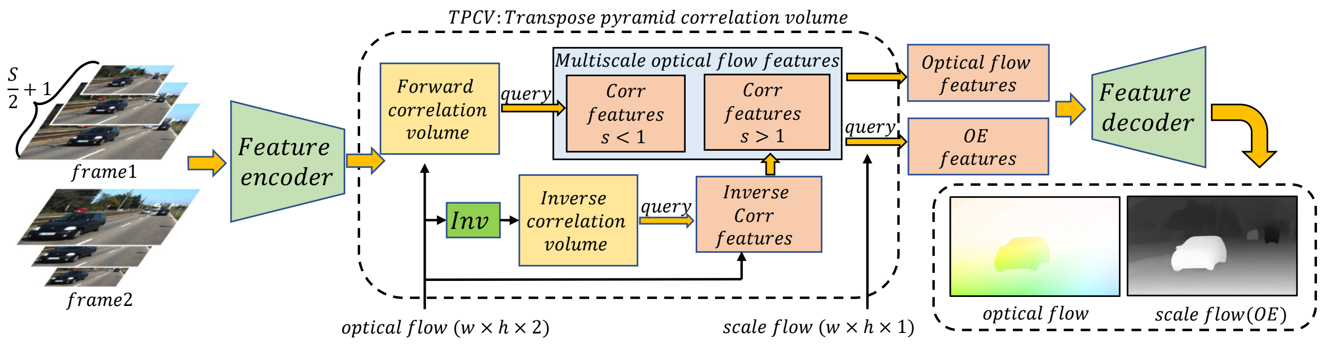


Figure 3. **A typical 3D optical flow calculation process.** The estimation of 3D optical flow using TPCV is mainly divided into two parts. First, query the correlation features of optical flow and OE from TPCV according to the current estimation, and then regress the optical flow and OE results based on the correlation features. The TPCV module does not contain any parameters to be optimized, even the calculation of inverse optical flow in the green box, so it can be easily applied to other methods.

scale changes in motion. Although some previous methods [6, 11, 34] also introduced the idea of the multi-scale image pyramid, but their pyramid is often used to save computing costs, expand network receptive field, speed up reasoning, and optimize loss function, which is still essentially image-to-image matching.

Scene flow Scene flow calculates the motion of dense pixels in 3D space. Recently, some successful scene flow methods [2, 8, 13, 18, 29, 33] are based on the rigid assumption, which assumes that the motion state of pixels on the same rigid object should be the same. These hypothesis-based optimization methods transform the scene flow task into a segmentation problem. Judging the foreground rigid object mask has become a key factor affecting the method's performance. However, it will inevitably introduce too much semantic information to assist judgment. Although these methods perform well on existing datasets, but require additional semantic labels and depth information to aid training.

There is also a class of methods [9] that highly fuse depth and optical flow information by combining the inference process of depth and optical flow networks, so that the results can complement each other. However, the segmentation of rigid masks is still implicitly involved in this process. Another kind of Monocular method [1, 32] calculates scene flow by optical expansion. Owing to optical expansion is directly regressed from optical flow or RGB images, it often obtains poor accuracy. Unlike them, TPCV obtains scale changes based on scale matching, eliminating the dependence on depth and semantic information.

3. Method

We propose a general optical flow upgrade module: Transpose pyramid correlation volume (TPCV), which can be widely applied to various 2D optical flow frameworks and upgraded to 3D optical flow. In this section, we will explain the construction details of TPCV, examples applied to RAFT, and supervision of 3D optical flow.

3.1. Construction Details of TPCV

The main function of the TPCV module is to index the correlation features of pixel pairs from different scales at different locations. These features will be used for subsequent optical flow estimation and scale estimation.

As shown in the Fig. 3, TPCV consists of three steps: Constructing the correlation volume of forward flow and reverse flow based on the input multi-scale features; Querying the correlation features from the correlation volume; Calculation of inverse optical flow.

3.1.1 Correlation Volume

The correlation volume of TPCV is divided into forward and inverse, where the forward correlation volume is essentially the dot product of the original-scale feature map of the frame 1 and the multi-scale feature map of the frame 2. The structure of the inverse correlation volume is the same as the forward; only the order of dot products is swapped.

Specifically, We extract the multi-scale features F_1^x and F_2^s respectively from the input image pyramid, where x and s refers to the scale change. The forward correlation volume consists of 4D correlation volumes C_s with different scales:

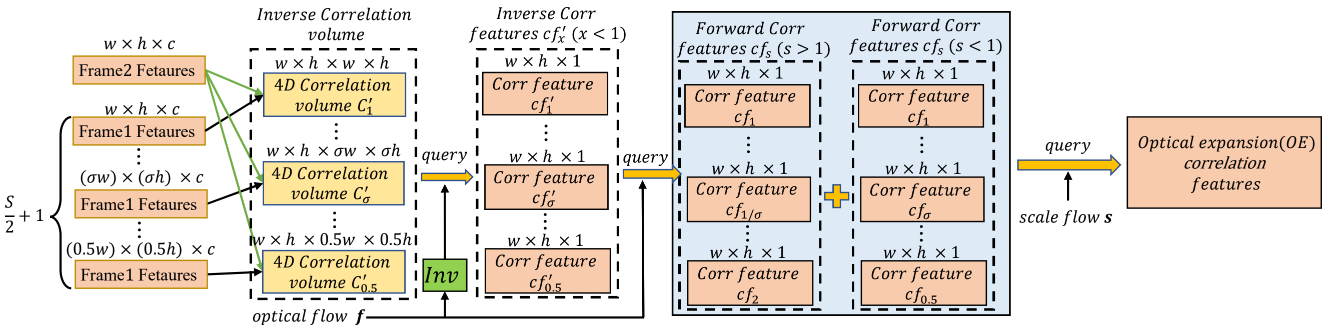
$$C_s(i,j,u,v) = F_1^1(i,j) \cdot F_2^s(u,v) \tag{1}$$

where (i,j) and (u,v) refer to locations in F_1^1 and F_2^s, $F_1^x \in \mathbb{R}^{xH \times xW \times C}$, $F_2^s \in \mathbb{R}^{sH \times sW \times C}$, $s, x = 0.5, 0.5 + 1/S, ..., 1$, $C_s \in \mathbb{R}^{H \times W \times sH \times sW}$, in our experiment $S = 4$.

Similarly, the inverse correlation volume C'_x is computed as:

$$C'_x(u,v,i,j) = F_2^1(u,v) \cdot F_1^x(i,j) \tag{2}$$

where (u,v) and (i,j) refer to locations in F_2^1 and F_1^x, $C'_x \in \mathbb{R}^{H \times W \times xH \times xW}$.



$w \times h \times c$
Frame2 Fetaures
$w \times h \times c$
Frame1 Fetaures
$(\sigma w) \times (\sigma h) \times c$
Frame1 Fetaures
$(0.5w) \times (0.5h) \times c$
Frame1 Fetaures
$\frac{S}{2}+1$
Inverse Correlation volume
$w \times h \times w \times h$
4D Correlation volume C_1'
$w \times h \times \sigma w \times \sigma h$
4D Correlation volume C_σ'
$w \times h \times 0.5w \times 0.5h$
4D Correlation volume $C_{0.5}'$
Inv
optical flow f
query
Inverse Corr features cf_x' $(x<1)$
$w \times h \times 1$
Corr feature cf_1'
Corr feature cf_σ'
Corr feature $cf_{0.5}'$
query
Forward Corr features cf_s $(s>1)$
Corr feature cf_1
Corr feature $cf_{1/\sigma}$
Corr feature cf_2
Forward Corr features cf_s $(s<1)$
Corr feature cf_1
Corr feature cf_σ
Corr feature $cf_{0.5}$
query
scale flow s
Optical expansion(OE) correlation features

Figure 4. **TPCV details.** This figure shows the details of the inverse flow query of TPCV: Build the inverse correlation volume C_x'; Query the inverse correlation feature cf_x' according to the optical flow estimation; Query the forward correlation feature cf_s from cf_x'; Query the OE features from cf_s according to the current OE estimation.

3.1.2 Query Correlation Feature

This section introduces how to query[1] the correlation features from the correlation volume, which is mainly divided into two steps. First, the multi-scale optical flow features are queried from the correlation volume based on the optical flow results. Then, the multi-scale optical flow features are sampled in the scale dimension based on the current OE results. As shown in Fig. 4.

After obtaining the forward optical flow $\boldsymbol{f}$ and the correlation volume C_s, the forward correlation feature $cf_s(s<1)$ can be queried as follows:

$$\begin{aligned} cf_s(i,j) &= C_s(i,j,su,sv) \\ &= C_s(i,j,s(i+\boldsymbol{f}(i,j,0)),s(j+\boldsymbol{f}(i,j,1))) \\ &= Corr(p,q^s) \end{aligned} \tag{3}$$

where $cf_s \in \mathbb{R}^{H\times W}$, (i,j) and (u,v) refer to the locations of p and q, $Corr(p,q^s)$ describes the Correlation between point p and point q scaled by s times, $s = 0.5, 0.5+1/S, ..., 1$.

Similarly, after acquiring the inverse optical flow $\boldsymbol{f'}$ and the correlation volume C_x', the inverse correlation feature $cf_x'(x<1)$ can be queried as follows:

$$\begin{aligned} cf_x'(u,v) &= C_x'(u,v,xi,xj) \\ &= C_x'(u,v,x(u+\boldsymbol{f'}(u,v,0)),x(v+\boldsymbol{f'}(u,v,1))) \\ &= Corr(q^x,p) \end{aligned} \tag{4}$$

where $\boldsymbol{f'}(u,v) = -\boldsymbol{f}(i,j), (u,v) = (i,j)+\boldsymbol{f}(i,j), cf_x' \in \mathbb{R}^{H\times W}$, $x = 0.5, 0.5+1/S, ..., 1$.

Then based on the forward optical flow $\boldsymbol{f}$ to query the forward correlation features cf_s when $s>1$:

$$\begin{aligned} cf_{1/x}(i,j) &= cf_x'(i+\boldsymbol{f}(i,j,0), j+\boldsymbol{f}(i,j,1)) \\ &= Corr(p,q^{1/x}) \end{aligned} \tag{5}$$

Combining $s<1$ and $s>1$, the complete forward optical flow correlation feature C_f is obtained:

$$C_f = \{cf_{0.5}, ..., cf_1, ..., cf_2\} \tag{6}$$

where $C_f \in \mathbb{R}^{H\times W\times(S+1)}$.

After getting C_f, the OE feature is obtained by querying the third scale dimension of C_f based on the current OE estimation $\boldsymbol{s}$. So far, we have described the query method of the OE features F_{oe}, and cf_1 is the optical flow correlation feature F_{of}.

In the process of applying TPCV to RAFT, we also query the neighborhood of optical flow and OE to obtain the optimization direction. We show more details about query in supplementary material.

3.1.3 Calculation of Inverse Optical Flow

In this section, we consider designing a inverse optical flow algorithm. In the previous work [12, 17], inverse optical flow is calculated by an extra network. However, this method is not applicable to TPCV. First, the extra network will increase the time and memory cost; second, to ensure the stability of querying, we hope that the forward and inverse flow can be perfectly closed-loop (about closed-loop, see Eq. (4), where $\boldsymbol{f'}(u,v) = -\boldsymbol{f}(i,j)$, $(u,v) = (i,j)+\boldsymbol{f}(i,j), (i,j) = (u,v)+\boldsymbol{f'}(u,v)$).

In order to ensure the perfect closed-loop of the forward and inverse flow, we refer to the idea of Javier [24], that is, the forward flow directly solves the inverse flow. The specific process is shown in Algorithm 1, where **corr** is cf_1, $\boldsymbol{x_w}$ is the current point to be processed, $\boldsymbol{x_i}$ is the four coordinate points around $\boldsymbol{x_w}$, $\boldsymbol{w_i}$ and $\boldsymbol{wght}$ are intermediate variables that store correlation weights.

[1]In this paper, the query is implemented by the gridsample function in Pytorch.

Algorithm 1 Inverse optical flow

```
Input: $\boldsymbol{f}$, $\boldsymbol{corr}$
Parameter: $\boldsymbol{wght}$
Output: $\boldsymbol{f'}$
1:  Initialize $\boldsymbol{f'} = 0$, $\boldsymbol{wght} = -\infty$.
2:  for all $\boldsymbol{x} \in \Omega$ do
3:      $\boldsymbol{x_w} \leftarrow \boldsymbol{x} + \boldsymbol{f}(\boldsymbol{x})$
4:      Find the four neighbors of $\boldsymbol{x_w}$ : $\{\boldsymbol{x_1}, \boldsymbol{x_2}, \boldsymbol{x_3}, \boldsymbol{x_4}\}$
5:      for $i = 1$ to 4 do
6:          $\boldsymbol{w_i} \leftarrow \boldsymbol{corr}(\boldsymbol{x_i}, \boldsymbol{x_w})$
7:          if $\boldsymbol{w_i} > \boldsymbol{wght}(\boldsymbol{x_i})$ then
8:              $\boldsymbol{f'}(\boldsymbol{x_i}) \leftarrow -\boldsymbol{f}(\boldsymbol{x})$
9:              $\boldsymbol{wght}(\boldsymbol{x_i}) \leftarrow \boldsymbol{w_i}$
10:         end if
11:     end for
12: end for
13: return $\boldsymbol{f'}$
```

3.2. Apply TPCV in RAFT

RAFT is a typical matching optical flow method, which iteratively maintains an optical flow field through the GRU module. In this section, we will explain how to apply TPCV to RAFT.

We do not change the structure of the RAFT feature extractor and GRU optimizer. The only change is that we use the TPCV module to replace the original 4D correlation volume to obtain optical flow correlation features and OE correlation features. We provide more details of network construction in the supplementary material.

Optimization After applying TPCV, we maintain an optical flow field $\boldsymbol{f}$ and a scale flow field $\boldsymbol{s}$ at the same time. The calculation of the optical flow update amount is the same as RAFT, and we add a new output header to update the scale flow from hidden features h_t generated by GRU:

$$\begin{aligned} h_t &= GRU(h_{t-1}, F_{of}, F_{oe}) \\ \Delta \boldsymbol{s} &= \tanh(Conv(ReLU(Conv(h_t)))) \end{aligned} \tag{7}$$

where $\Delta \boldsymbol{s}$ is the update amount of scale flow, initial h_0 is given by feature encoder.

3.3. Supervision of 3D Optical Flow

3.3.1 2D Optical Flow Loss

We follow RAFT and supervise the 2D optical flow with L_1 distance between the predicted flow and ground truth, the 2D optical flow loss is defined as:

$$\mathcal{L}_f = \sum_{k=1}^{N} \gamma^{N-k} \left\| \boldsymbol{f}^k - \boldsymbol{f}_{gt} \right\|_1 \tag{8}$$

where we set $\gamma = 0.8$, $N = 12$ in our experiments.

3.3.2 Scale Flow Loss

Existing datasets do not directly provide supervised scale data. We refer to the scale change and motion in depth conversion formula established by Gengshan Yang [32] to train the 3D optical flow indirectly.

Yang pointed out that under the assumption of small rotation, the scale change and motion-in-depth are approximately equal, which allows us to use the scene flow dataset to obtain the scale loss, which is defined as follows:

$$\mathcal{L}_s = \sum_{k=1}^{N} \gamma^{N-k} \left\| \boldsymbol{s}^k - z'_{gt}/z_{gt} \right\|_1 \tag{9}$$

where z'_{gt} and z_{gt} are the ground truth depths of the pixels in the second frame and the first frame, $\boldsymbol{s}$ is the scale flow field.

The overall loss function of 3D optical flow is:

$$\mathcal{L} = \mathcal{L}_f + \mathcal{L}_s \tag{10}$$

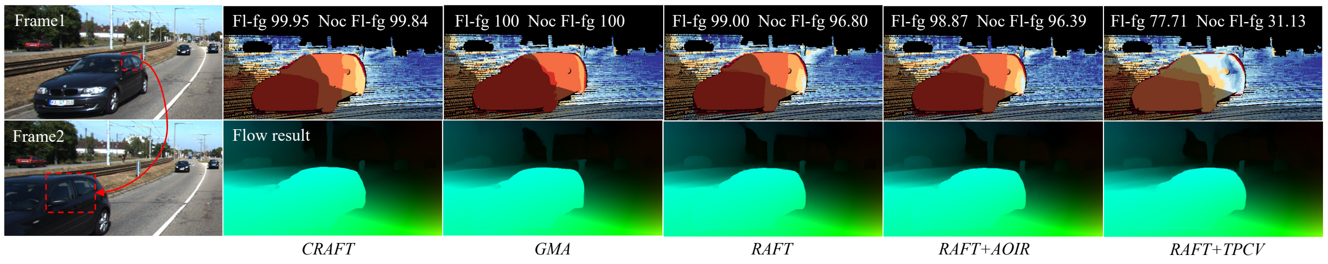
4. Experiments

Experimental results are discussed in this section to demonstrate the effectiveness of TPCV. Experiments show that TPCV can significantly improve the estimation accuracy of the optical flow for foreground objects under the same number of optimizer parameters. Furthermore, the ability of motion-in-depth estimation far exceeds the prior SOTA method. Specifically, we first compared the optical flow performance between the TPCV and baseline models. Then we applied the 3D optical flow results to the 3D scene perception tasks, including motion-in-depth, time-to-collision and scene flow estimation.

Setup We first pre-train our model on the Driving (D) for 80k iterations (batch = 6, lr = 0.002). For the optical flow estimation task, we finetune it on KITTI (K-200) with 60k iterations (batch = 6, lr = 0.00125). For motion-in-depth and time-to-collision task, we follow the split of optical expansion [32]. Specifically, we select one for every 5 images in K-200 for validation (K-40) and add the rest 160 images for training (K-160) with 60k iterations (batch = 6, lr = 0.00125). For the scene flow task, we use K-200 for training with 60k iterations (batch = 6, lr = 0.00125) and submit the results to the KITTI scene flow benchmark.

4.1. Optical Flow

In this section, we compare the effect of applying TPCV on the baseline optical flow model. We chose RAFT [27] as the baseline model, not because it offers better performance, but because it is the basic framework for most advanced optical flow models [5, 16, 25, 26] and has strong representativeness.



Frame1
Frame2
Fl-fg 99.95 Noc Fl-fg 99.84
Fl-fg 100 Noc Fl-fg 100
Fl-fg 99.00 Noc Fl-fg 96.80
Fl-fg 98.87 Noc Fl-fg 96.39
Fl-fg 77.71 Noc Fl-fg 31.13
Flow result
CRAFT
GMA
RAFT
RAFT+AOIR
RAFT+TPCV

Figure 5. **Results for KITTI test image.** The upper line is the optical flow error. The redder the colour represents the greater the error, where Noc Fl-fg is the outlier rate of the non-occluded part, and the lower line is the colour-coded optical flow result. Observing the car window in the red box, there is a large-scale change between the two frames, and only our TPCV can get the correct result.

The main indicator we compare is the outlier rate of foreground and background optical flow, where outliers are the points whose endpoint error is greater than 3px or 5%.

Method	Fl-bg	Fl-fg	Fl-all	Improvement
RAFT [27]	4.74	6.87	5.1	-
GMA [5]	4.78	7.03	5.15	+attention
CRAFT [25]	4.58	5.85	4.79	+attention
RAFT+AOIR [26]	4.68	6.99	5.07	regularisation
RAFT+TPCV(ours)	**4.53**	**5.52**	**4.69**	scale correlation

Table 1. **Quantitative optical flow performance comparison on KITTI test**. Fl-bg and Fl-fg are the outlier rates of the foreground and background optical flow. The best among the all are bolded. Our method greatly improves the performance of foreground optical flow.

Validation performance We first compare the performance of TPCV and baseline RAFT. As shown in Tab. 1, TPCV improves the accuracy of the baseline model comprehensively, especially the foreground error is reduced from 6.87 to 5.52, which proves that TPCV can help the baseline model adapt to the foreground object scale changes effectively.

We also compare some state-of-the-art RAFT-based improvement methods' performance, where CRAFT and GMA both introduce attention mechanisms into the RAFT framework to solve the occlusion problem. While RAFT+AOIR studies new regularization schemes, these methods all change the structure of RAFT's optimizer and feature extractor, increasing the computational cost and training cost. However, our TPCV still achieves the best results, although the optimizer structure is not changed. Fig. 5 shows optical flow results under large-scale changes. All methods except TPCV cannot handle large-scale changes, proving the superiority of TPCV's scale matching strategy.

4.2. Motion-in-depth (MID)

In this section, we discuss the performance of 3D optical flow in the MID estimation task. Based on the relationship established in the previous work [32], we can equate the scale change s to MID τ. For example, an object is doubled in size between two frames ($s = 0.5$, the second frame is scaled by 0.5 times and matches the first frame), which means that its distance from the camera is shortened by 0.5 times ($\tau = z'/z = s = 0.5$, where z' and z are the depth of frame2 and frame1).

MID is a key indicator in tasks such as time-to-collision, scene flow, and depth estimation. To demonstrate the superiority of TPCV, we compare it with the current state-of-the-art MID methods. The error of MID is defined as:

$$MID_{error} = ||log(\tau) - log(\tau_{gt})||_1 \cdot 10^4 \tag{11}$$

where τ_{gt} is obtained by dividing the matching depth of the second frame and the first frame.

Method	Input	MID_{error}	Time/s
OSF [20]	Stereo	115	3000
PRSM [30]	Stereo	124	300
Binary TTC [1]	Mono	73.55	2.2
Optical expansion [32]	Mono	75	**0.2**
RAFT+TPCV(ours)	Mono	**42.84**	**0.2**

Table 2. **MID estimation on K-40 validation set**. Our method outperforms other methods by a large margin.

We first compare with traditional optimization methods that use stereo information: PSMR and OSF decompose pixels into rigid blocks, and iteratively update the optimized optical flow and depth fields based on rigid assumptions and hand-designed priors. To calculate the MID we divide the depth of the second frame by the depth of the first frame. Experimental results show that our errors are much smaller than them (115 vs. 42.84) and in less time[2].

We also compare with the state-of-the-art MID methods Optical expansion and Binary TTC. TPCV outperforming

[2]Timing results are computed on 375x1242 images with an RTX 3090 GPU using 200 updates.

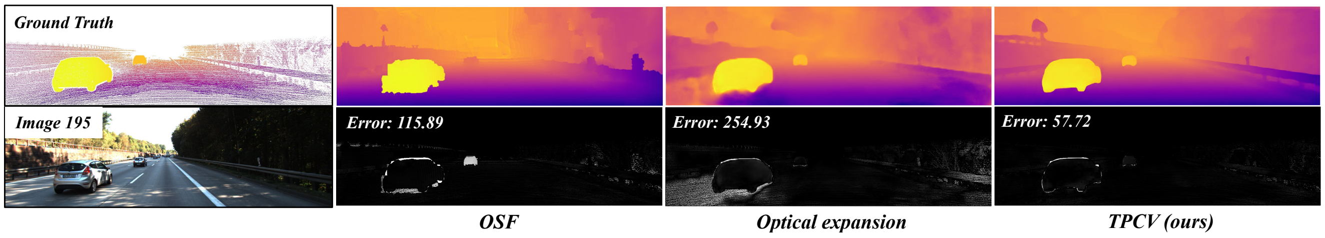


Figure 6. **Motion-in-depth results for image "195" in the K-40.** Up: Ground truth and motion-in-depth results, where the darker indicates the objects moving towards the camera. Down: The input image and error map, where the whiter the pixels, the greater the error. In both foreground and background regions, our method is much more accurate than other methods.

them by a large margin (73.55 vs 42.84), and the average error is reduced by 42%. Different from their directly regress MID, TPCV calculates MID based on matching, making the result robust and accurate. The above experiments prove that our TPCV module is progressiveness in depth motion estimation.

4.3. Time-to-collision (TTC)

Time-to-collision estimation is essential for obstacle avoidance and path planning tasks [3, 7, 14, 21]. In fact, the time for a pixel to reach the camera plane (TTC) can be directly obtained from MID:

$$TTC = \frac{Z}{Z - Z'}T = \frac{T}{1 - \frac{Z'}{Z}} \tag{12}$$

where T is the sampling interval of the camera (in KITTI $T = 0.1s$) and Z'/Z is the motion-in-depth τ.

Method	Err-1s	Err-2s	Err-5s	Time/s
Binary TTC [1]	2.4	3.16	3.25	**0.0192**
OSF [20]	1.79	2.93	4.03	3000
Optical expansion [32]	1.86	1.87	2.45	0.2
RAFT+TPCV(ours)	**1.14**	**1.65**	**1.57**	0.2

Table 3. **Percentage errors of time-to-collision estimation on K-40 validation set**

We treat TTC as a binary classification problem, and for each pixel, we judge whether the TTC time is less than $\{1s, 2s, 5s\}$. Only the points moving towards the camera are evaluated (the TTC of points farther away is meaningless).

In this experiment, we use the binary mode of Binary TTC. In this mode, a binary mask will be directly output. Although the accuracy is decreased, the reasoning speed is greatly improved. Optical expansion and OSF perform reasonably well, consistent with their high precision performance in the motion-in-depth task. Our monocular method achieves the lowest error in all time intervals, which proves that the 3D optical flow estimated by TPCV can well predict future collisions.

4.4. Scene Flow

Correctly estimating the 3D motion of objects in space is crucial for autonomous driving and dynamic environment path planning for robots. The scene flow consists of the optical flow and the depth of matched pixels between two frames. In order to unify the standard with the most advanced methods, we use GA-Net [35] to obtain the depth of the first frame d_1, and obtain the depth of the second frame d_2 based on the motion-in-depth τ estimated by TPCV ($d_2 = d_1 * \tau$). The model was trained on the complete K-200 dataset and tested on the KITTI public scene flow evaluation website[3].

We first compare TPCV with monocular methods, as shown in Tab. 4 TPCV leads in all monocular methods with significant advantages, consistent with the previous superior performance of TPCV in motion-in-depth and optical flow tasks. It proves the effectiveness of our matching strategy in the field of 3D motion estimation.

We also compare with the state-of-the-art depth-based methods CamLiFlow, RigidMask+ISF and RAFT3D. Noticing that they tend to get smaller Fl-bg errors, which is thanks to the aid of depth information to optimize the background optical flow. Nonetheless, our monocular approach achieves the best results on all foreground metrics and maintains a competitive inference speed. We note that TPCV outperforms even most depth-based methods on the D2-all metric, which demonstrates the high potential of our 3D optical flow for estimating 3D motion.

RAFT3D + TPCV We apply TPCV to RAFT3D. Specifically, we directly use TPCV to replace the correlation volume in RAFT3D, and change the original direct regression depth change to the estimated depth change ratio. The experimental results show that the addition of TPCV significantly improves the performance of SF-all ($5.77 \rightarrow 5.00$). However, we found that Fl-fg did not improve. We think it is because RAFT3D introduced potential semantic information when estimating the foreground.

[3]http://www.cvlibs.net/datasets/kitti/eval_scene_flow.php

Method	Input	D1-bg	D1-fg	D1-all	D2-bg	D2-fg	D2-all	Fl-bg	Fl-fg	Fl-all	SF-bg	SF-fg	SF-all
CamLiFlow [9]	D+M	1.48	3.46	1.81	**1.92**	8.14	**2.95**	**2.31**	7.04	**3.1**	**2.87**	12.23	**4.43**
RigidMask+ISF [33]	D+M	1.53	3.65	1.89	2.09	8.92	3.23	2.63	7.85	3.5	3.25	13.08	4.89
RAFT3D [28]	D+M	1.48	3.46	1.81	2.51	9.46	3.67	3.39	8.79	4.29	4.27	13.27	5.77
RAFT3D+TPCV(ours)	D+M	1.48	3.46	1.81	1.93	8.79	3.07	2.48	10.19	3.76	3.03	14.82	5.0
Optical expansion [32]	M	1.48	3.46	1.81	3.39	8.54	4.25	5.83	8.66	6.3	7.06	13.44	8.12
Binary TTC [1]	M	1.48	3.46	1.81	3.84	9.39	4.76	5.84	8.67	6.31	7.45	13.74	8.5
RAFT+TPCV(ours)	M	1.48	3.46	1.81	2.29	**7.63**	3.18	4.53	**5.52**	4.69	5.34	**10.6**	6.21

Table 4. **State-of-art published methods on KITTI scene flow benchmark.** D+M means depth and monocular information, and M means monocular information. D1, D2, Fl, and SF is the percentage of disparity, optical flow and scene flow outliers. -bg,-fg and -all represent the percentage of outliers averaged only over background regions, foreground regions and overall ground truth pixels. The best among all are bolded, and the second best are underlined. Our monocular method performs quite well, not only far ahead of similar monocular methods, but also outperforming stereo-based methods in foreground scene flow estimation, which proves that the TPCV module can accurately estimate the 3D motion.

5. Ablation

Setup We conduct more extensive experiments to verify the advanced nature of the TPCV module. For all experiments, we pre-train with 80K on the Driving dataset (batch=6, lr = 0.00025) , refinement training on K-160 (batch=6, lr = 0.000125) , and testing on the K-40 dataset.

Method	$Flow_{epe}$	MID_{error}	Memory
RAFT*	1.61	56.2	2.7GB
RAFT+pyramid	1.42	48.9	5.2GB
RAFT+TPCV	1.31	42.8	3.0GB

Table 5. **Ablation study on optical flow and motion-in-depth estimation**. $Flow_{epe}$ is the average endpoint pixel error of optical flow estimation, the lower the better.

Comparison to regression (RAFT*) We first replace the TPCV in "RAFT+TPCV" with 4D correlation volume, at this time, the estimation of scale flow degenerates into direct regression, similar to optical expansion [32]. As shown in Tab. 5, after replacement, the optical flow error increased from 1.31 to 1.61 (increases 22.9%), and the motion-in-depth error increased from 42.8 to 56.2 (increases 31.3%). It proves that the TPCV module plays a crucial role in advanced optical flow and motion-in-depth estimation.

Comparison to pyramid matching (RAFT+pyramid) We also tried a pyramid matching scheme, shown in Fig. 2 (b), which matched the first frame feature with the same scale feature in the image pyramid of the second frame. we directly obtain large-scale images through interpolation. As shown in Tab. 5, the optical flow and motion-in-depth errors are greatly reduced after introducing the matching strategy, which once again proves the superiority of the matching method. However, it is still not as good as the performance of TPCV (1.42 vs. 1.31; 48.9 vs. 42.8); we believe that it is mainly due to the construction of false large-scale features, which makes it difficult to perform correct feature matching for objects far away from the camera.

6. More experimental results

We followed the experimental procedure of RAFT3D [28] to conduct more experiments on the sizeable synthetic dataset Flyingthings3D. The results show that after applying TPCV, the end-to-end 3D error of RAFT3D is reduced from 0.062m to 0.048m, and the end-to-end 2D error dropped from 2.2px to 1.95px. See supplementary materials for details of results.

Limitation We also conducted experiments on Sintel, but the baseline results did not improve significantly after applying TPCV; we believe this is because Sintel contains many non-rigid scenes, while the conversion between MID and OE depends on the rigid assumption.

7. Conclusion

We combine optical flow and optical expansion, proposing the concept of 3D optical flow, which can eliminate the influence of scale change on optical flow matching and provide motion information in the depth direction. Then we design a plug-and-play 2D to 3D optical flow upgrade module: TPCV, associated with a set of supervised learning strategies. Experimentally, we apply TPCV to the RAFT, achieving significant performance improvements in 2D optical flow tasks and multiple 3D perception tasks, including motion-in-depth, scene flow, and time-to-collision.

Acknowledgements This research was supported by the National Natural Science Foundation of China (Grant nos. 61673220, 62106209), and the Project of SongShan Laboratory (Grant no. YYJC012022015).

References

[1] Abhishek Badki, Orazio Gallo, Jan Kautz, and Pradeep Sen. Binary ttc: A temporal geofence for autonomous navigation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12946–12955, 2021. 1, 3, 6, 7, 8

[2] Aseem Behl, Omid Hosseini Jafari, Siva Karthik Mustikovela, Hassan Abu Alhaija, Carsten Rother, and Andreas Geiger. Bounding boxes, segmentations and object coordinates: How important is recognition for 3d scene flow estimation in autonomous driving scenarios? In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2574–2583, 2017. 3

[3] Jeffrey Byrne and Camillo J Taylor. Expansion segmentation for visual collision detection and estimation. In *2009 IEEE International Conference on Robotics and Automation*, pages 875–882. IEEE, 2009. 1, 7

[4] TA Camus. Calculating time-to-contact using real-time quantized optical flow. 1995. 1

[5] 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. 2, 5, 6

[6] Rico Jonschkowski, Austin Stone, Jonathan T Barron, Ariel Gordon, Kurt Konolige, and Anelia Angelova. What matters in unsupervised optical flow. In *European Conference on Computer Vision*, pages 557–572. Springer, 2020. 3

[7] David N Lee. A theory of visual control of braking based on information about time-to-collision. *Perception*, 5(4):437–459, 1976. 7

[8] Congcong Li, Haoyu Ma, and Qingmin Liao. Two-stage adaptive object scene flow using hybrid cnn-crf model. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 3876–3883. IEEE, 2021. 3

[9] Haisong Liu, Tao Lu, Yihui Xu, Jia Liu, Wenjie Li, and Lijun Chen. Camliflow: Bidirectional camera-lidar fusion for joint optical flow and scene flow estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5791–5801, 2022. 3, 8

[10] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004. 1

[11] Kunming Luo, Chuan Wang, Shuaicheng Liu, Haoqiang Fan, Jue Wang, and Jian Sun. Upflow: Upsampling pyramid for unsupervised optical flow learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1045–1054, 2021. 3

[12] Kunming Luo, Chuan Wang, Shuaicheng Liu, Haoqiang Fan, Jue Wang, and Jian Sun. Upflow: Upsampling pyramid for unsupervised optical flow learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1045–1054, June 2021. 4

[13] Wei-Chiu Ma, Shenlong Wang, Rui Hu, Yuwen Xiong, and Raquel Urtasun. Deep rigid instance scene flow. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3614–3622, 2019. 3

[14] Thiago Marinho, Massinissa Amrouche, Venanzio Cichella, Dušan Stipanović, and Naira Hovakimyan. Guaranteed collision avoidance based on line-of-sight angle and time-to-collision. In *2018 Annual American Control Conference (ACC)*, pages 4305–4310. IEEE, 2018. 7

[15] 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. 2

[16] Lukas Mehl, Cedric Beschle, Andrea Barth, and Andrés Bruhn. An anisotropic selection scheme for variational optical flow methods with order-adaptive regularisation. In *International Conference on Scale Space and Variational Methods in Computer Vision*, pages 140–152. Springer, 2021. 5

[17] Simon Meister, Junhwa Hur, and Stefan Roth. Unflow: Unsupervised learning of optical flow with a bidirectional census loss. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018. 4

[18] 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. 3

[19] Moritz Menze, Christian Heipke, and Andreas Geiger. Joint 3d estimation of vehicles and scene flow. In *ISPRS Workshop on Image Sequence Analysis (ISA)*, 2015. 2

[20] Moritz Menze, Christian Heipke, and Andreas Geiger. Object scene flow. *ISPRS Journal of Photogrammetry and Remote Sensing*, 140:60–76, 2018. 6, 7

[21] Tomoyuki Mori and Sebastian Scherer. First results in detecting and avoiding frontal obstacles from a monocular camera for micro unmanned aerial vehicles. In *2013 IEEE International Conference on Robotics and Automation*, pages 1750–1757. IEEE, 2013. 7

[22] Dennis Muller, Josef Pauli, Christian Nunn, Steffen Gormer, and Stefan Muller-Schneiders. Time to contact estimation using interest points. In *2009 12th International IEEE Conference on Intelligent Transportation Systems*, pages 1–6. IEEE, 2009. 1

[23] Maria Sagrebin, Anastasia Noglik, and Josef Pauli. Robust time-to-contact calculation for real time applications. In *Proceedings of 18th International Conference on Computer Graphics and Vision*, pages 128–133, 2008. 1

[24] Javier Sánchez, Agustín Salgado, and Nelson Monzón. Computing inverse optical flow. *Pattern Recognition Letters*, 52:32–39, 2015. 4

[25] 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. 2, 5, 6

[26] Deqing Sun, Daniel Vlasic, Charles Herrmann, Varun Jampani, Michael Krainin, Huiwen Chang, Ramin Zabih, William T Freeman, and Ce Liu. Autoflow: Learning a better training set for optical flow. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10093–10102, 2021. 5, 6

[27] Zachary Teed and Jia Deng. Raft: Recurrent all-pairs field transforms for optical flow. In *European conference on computer vision*, pages 402–419. Springer, 2020. 2, 5, 6

[28] Zachary Teed and Jia Deng. Raft-3d: Scene flow using rigid-motion embeddings. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8375–8384, 2021. 8

[29] Christoph Vogel, Konrad Schindler, and Stefan Roth. 3d scene flow estimation with a rigid motion prior. In *2011 International Conference on Computer Vision*, pages 1291–1298. IEEE, 2011. 3

[30] Christoph Vogel, Konrad Schindler, and Stefan Roth. 3d scene flow estimation with a piecewise rigid scene model. *International Journal of Computer Vision*, 115(1):1–28, 2015. 6

[31] 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. 2

[32] Gengshan Yang and Deva Ramanan. Upgrading optical flow to 3d scene flow through optical expansion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1334–1343, 2020. 1, 3, 5, 6, 7, 8

[33] Gengshan Yang and Deva Ramanan. Learning to segment rigid motions from two frames. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1266–1275, 2021. 3, 8

[34] Hanchao Yu, Xiao Chen, Humphrey Shi, Terrence Chen, Thomas S Huang, and Shanhui Sun. Motion pyramid networks for accurate and efficient cardiac motion estimation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 436–446. Springer, 2020. 3

[35] Feihu Zhang, Victor Prisacariu, Ruigang Yang, and Philip HS Torr. Ga-net: Guided aggregation net for end-to-end stereo matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 185–194, 2019. 7

[36] Shiyu Zhao, Long Zhao, Zhixing Zhang, Enyu Zhou, and Dimitris Metaxas. Global matching with overlapping attention for optical flow estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 17592–17601, June 2022. 2