

---

# Crowd Flow Analysis using RAFT

---

**Ritika Kishore Kumar**  
University of California, San Diego  
rkishorekumar@ucsd.edu  
A59015651

**Surya Prakash Thoguluva Kumaran Babu**  
University of California, San Diego  
stk222@ucsd.edu  
A59017860

## Abstract

This paper focuses on the application of the RAFT (Recurrent All-Pairs Field Transforms) network for optical flow estimation in crowd scenes. The RAFT network has shown promise in handling complex motion dynamics, and this paper evaluates its performance on the CrowdFlow dataset. Four variants of RAFT were implemented and compared, including a baseline model, attention-based model, Charbonnier loss model, and a combined modification. Evaluation metrics such as End Point Error (EPE) and total test loss were recorded for each variant. The results demonstrate that incorporating self-attention in the context layer improves performance, while Charbonnier loss has a negative effect on the model. The combined modification did not surpass the performance achieved by using self-attention alone.

**Keywords:** Optical flow estimation, crowd scenes, RAFT, self-attention, Charbonnier loss.

## 1 Introduction

Optical flow estimation [1] is a fundamental task in computer vision that involves the analysis and tracking of the motion of objects within a sequence of images. It quantifies the apparent motion of objects by estimating the displacement vector for each pixel between two frames. Accurate optical flow estimation is crucial for a wide range of applications, including video analysis, object tracking, motion-based recognition, autonomous navigation (Visual Odometry) [2], augmented reality, and visual effects in movies.

In recent years, the field of optical flow estimation has witnessed significant advancements, especially with the emergence of deep learning-based approaches. These techniques have shown promise in handling complex motion patterns and challenging scenarios. However, the accurate estimation of optical flow in crowd scenes remains a formidable task due to the unique characteristics and complexities associated with dense crowds.

Crowd scenes pose several challenges for optical flow estimation algorithms. High crowd density, occlusions, diverse motion patterns, and complex interactions among individuals make it difficult to capture and model the intricate motion dynamics accurately. By addressing the challenges posed by crowd scenes, we aim to contribute to a better understanding of crowd dynamics, crowd management, behavior prediction, anomaly detection, and intelligent surveillance, thus benefiting various real-world problems.

In this paper, we present our approach to applying the RAFT (Recurrent All-Pairs Field Transforms) [3] network for optical flow estimation on the CrowdFlow dataset [4]. We evaluate the performance of the proposed method using endpoint error metric, for comparing it with existing techniques and showcasing its effectiveness in handling crowd scenes. The results obtained from our experiments demonstrate the potential of the RAFT network for crowd-related applications and provide valuable insights for further advancements in the field of optical flow estimation.

## 2 Related work

Optical flow estimation has a rich history, beginning with classic methods such as the **Lucas-Kanade algorithm** proposed by Lucas and Kanade [5]. This iterative algorithm utilized local image gradients for estimating optical flow. Another influential approach was the **Horn-Schunck method** introduced by Horn and Schunck [6], which formulated optical flow estimation as a global optimization problem based on brightness constancy and smoothness constraints.

Advancements in deep learning has led to significant advancements in optical flow estimation. Dosovitskiy et al. introduced **FlowNet** [7], one of the pioneering deep learning models specifically designed for optical flow estimation. FlowNet employed a stacked convolutional neural network (CNN) architecture and achieved state-of-the-art performance on benchmark datasets.

The subsequent works focused on improving different aspects of optical flow estimation. For instance, the **PWC-Net** architecture proposed by Sun et al. [8] incorporated pyramid processing, differentiable warping, and cost volume computation to refine flow estimation in a coarse-to-fine manner. **SPyNet** by Ranjan et al. [9] utilized a pyramidal framework with spatial pyramid pooling and image warping to capture multi-scale motion information. Hui et al. [10] introduced **Lite-FlowNet**, which aimed at achieving real-time performance while maintaining accuracy through a compact network architecture and cost volume filtering.

More recently, Teed and Deng [3] introduced the **RAFT (Recurrent All-Pairs Field Transforms)** network, which employed a recurrent neural network with a differentiable patch-matching algorithm and recurrent update steps for iterative refinement of optical flow estimation. Liu et al. [11] extended RAFT with **SMURF (Self-Teaching Multi-Frame Unsupervised RAFT with Full-Image Warping)**, leveraging self-supervision and full-image warping to improve unsupervised optical flow estimation.

To facilitate the training and evaluation of optical flow models, synthetic datasets have been developed. The **Flying Chairs** dataset introduced by Mayer et al. [12] provided a large-scale synthetic dataset for training deep networks for optical flow estimation. It offered diverse scenes and motion patterns, enabling robust model training. **FlyingThings3D** dataset, introduced by Lempitsky and Zisserman [13] is yet another synthetic dataset which provided a photorealistic synthesized dataset for studying shape, motion, and stereo.

Real-world datasets play a crucial role in assessing the performance of optical flow methods. While synthetic datasets have been widely used due to their availability and ease of generating ground truth annotations, real-world datasets provide valuable insights into the challenges posed by complex and dynamic scenes.

The **KITTI** dataset by Menze et al. [14] has emerged as a widely-used benchmark, particularly in the context of autonomous driving. It offers real-world driving sequences with accurate ground truth annotations for evaluating optical flow algorithms in realistic driving scenarios. **Sintel** dataset by Butler et al. [15] has also played a significant role in advancing optical flow estimation. It provides high-quality, complex synthetic scenes with realistic motion and occlusions.

In the domain of crowd analysis, the availability of suitable datasets is limited. To address this, Luiten et al. [16] introduced the **CrowdFlow** dataset, which specifically addresses challenges related to crowd analysis and understanding.

## 3 Dataset

In this project, we have used CrowdFlow dataset benchmark [4] to test our RAFT implementation. Although there existed reference benchmarks like KITTI and MPI-Sintel, CrowdFlow is argued to produce new challenges in the optical flow estimation algorithm. Optical flow estimation algorithms heavily depend on the contents in the frame. In CrowdFlow, hundreds of people’s small movements need to be tracked without explicitly mentioning them in the frame. This is in contrast with the existing benchmarks like KITTI, MPI-Sintel because of the fact that:

- Many small pixel displacements that are independent of each other need to be estimated.












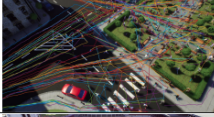

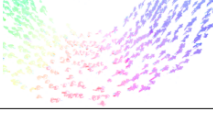

Sequence	Sample	Description	Optical flow field	Person trajectories
IM01 (Static/Dynamic) 371 individuals 300 frames		Few pedestrians walking against a main crowd flow.		
IM02 (Static/Dynamic) 631 individuals 300 frames		Bottleneck dividing one major flow into three.		
IM03 (Static/Dynamic) 878 individuals 250 frames		Two dense flows walking close past each other.		
IM04 (Static/Dynamic) 344 individuals 300 frames		Spread of collective panic and subsequent escape.		
IM05 (Static/Dynamic) 1451 individuals 450 frames		Marathon sequence. Long temporal tracking.		

Figure 1: CrowdFlow Dataset Overview. Image credits [4]

- There is a high potential for two closeby pixels to have movements in different directions of flow independent of each other.
- Algorithm on these datasets will perform better only if it has consistent performance over long temporal ranges.

The application of this CrowdFlow dataset is in crowd surveillance, crowd segmentation, and behavior analysis. Our key goal is to test the generalization of the RAFT network to a new dataset which has new challenges compared to the datasets with which the RAFT was trained initially.

The dataset has 10 sequences with 300 to 400 frames each that are captured at 25 Hz with a resolution of  $1280 \times 720$ . A unique feature of the dataset is that all the sequences are continuous, unlike the Sintel dataset benchmark. This allows to evaluation the algorithm for temporal consistency. There are 5 unique sequences of scenes that are rendered twice, one with static viewpoint and the other with a dynamic, airborne viewpoint which accounts for external disturbances like wind effects. Each sequence has a different crowd density varying from 371 individuals to 1451 individuals. The motion of the crowd has two different variants where a single crowd moves in one direction or 2 crowds cross each other and move in opposite directions.

Optical flow and optical trajectory ground truth is provided with the dataset. The optical flow ground truth is used for training and testing our RAFT implementation. A list of example sequences and their ground truth visualization is shown in figure 1.

## 4 Methodology

RAFT is a novel deep-learning architecture for estimating optical flow between 2 image frames. In this project, we have re-implemented RAFT network. It was shown that RAFT had main advantages like state of accuracy, strong generalization, and high efficiency which makes RAFT stand out compared to its counterparts. As mentioned in the original RAFT paper [3] RAFT draws inspiration from computer vision techniques to solve the optical flow as an energy minimization optimization problem except that it now learns to define its own objective function and solve it. RAFT has unique features compared to other existing methods. RAFT maintains one single high-resolution estimate of the flow field and iteratively updates it using the update block. This eliminates the limitations of recovering from noise when we start with a lower-resolution flow estimate and scale to a higher resolution. RAFT uses recurrent update blocks which share weights. This reduced the number of

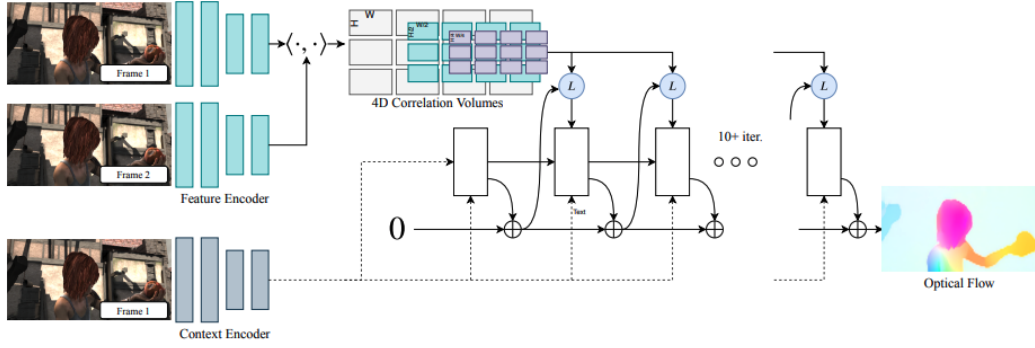


Figure 2: RAFT Network Overview. Frame 1 and Frame 2 are passed through the feature encoder layers to get 2 different feature maps. Inner product between all pairs of these features is computed to obtain the 4D correlation block. Frame 2 is passed through context layers to get context feature map. Finally, the update block takes inputs from the context feature maps, subset of 4D correlation volume, hidden state from the previous block and the flow to make a prediction of the optical flow. Image credits [3]

parameters and makes it easier to train faster. This has also unlocked the potential to go up to any number of iterations to estimate flow rather than being limited to a fixed number. The number of iterations is now a hyperparameter that can be tuned for the best performance. A brief of the RAFT architecture that is being implemented and the motivation behind them is discussed in this section.

Broadly RAFT architecture can be split into 4 categories namely,

- **Feature encoder:** A CNN network that takes both the images as inputs and gives a feature map at a lower resolution.
- **Correlation Block:** A 4D volume is constructed by taking the inner product of all pairs of feature vectors generated by the feature encoder. This 4D volume is then average pooled in the last two dimensions to get many lower resolution volumes that are stacked like a pyramid.
- **Context encoder:** This is similar to the feature encoder network except that this is only applied to the first input image. The output feature map is passed on to the update block.
- **Update Block:** A Gated Recurrent Unit (GRU) is used to recursively update a zero-initialized flow field by using a subset of correlation volume, feature map from the context encoder and the hidden state from the previous update unit.

A general working of RAFT architecture is shown in 2. We have implemented RAFT-S architecture in this project. RAFT-S is a lighter version of RAFT with 1M parameters.

#### 4.1 Input and Output

The input to the RAFT network are 2 RGB images  $I_1, I_2$  of size  $H \times W \times 3$  each, a dense displacement field  $f^1, f^2$  which maps each pixel  $(u, v)$  in  $I_2$  to its corresponding coordinates  $(u', v') = (u + f^1(u), v + f^2(v))$  in  $I_1$ . All images in CrowdFlow dataset are of the size 384 x 512 x 3. The output from the architecture is of the shape 384 x 512 x 2 corresponding to the x and y directional vectors for each pixel and matches the ground truth flow field size of the CrowdFlow dataset.

#### 4.2 Feature Encoder

A CNN architecture is used to extract the features from the input images before computing the correlation volume. This is analogous to the traditional optical flow methods, where handcrafted features from the images are used as input to those algorithms. The advantage of using a neural network to extract features is that, now the network has the ability to learn the feature transform as

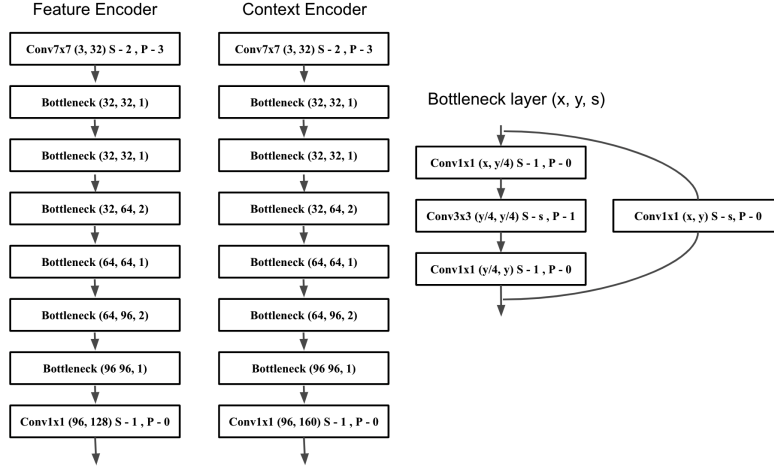


Figure 3: Feature encoder defined in terms of convolution blocks and bottleneck layer

the entire RAFT network is end-to-end differentiable, and gradients from the loss function can be backpropagated all the way to the input images. The encoder  $g_\theta$  takes the input and gives features at  $1/8$  resolution  $g_\theta : \mathbb{R}^{H \times W \times 3} \mapsto \mathbb{R}^{H/8 \times W/8 \times D}$  where  $D = 128$ . The architecture of the feature encoder is shown in figure 3

### 4.3 Context Encoder

The context encoder in the RAFT shares the same CNN architecture as the feature encoder but is specifically applied to the second image in the input pair. The context encoder, represented by **cmapH (Hidden context)** and **cmapC (Input context)**, extracts contextual features from the input frames. cmapH contains processed feature maps used directly for flow estimation, capturing the global and local context of the scene. cmapC represents a lower-resolution version of the feature maps, providing coarser details to assist in regularization during flow estimation. cmapH contributes to the correlation layer, where feature map correlations guide flow estimation and establish correspondences. cmapC, on the other hand, aids in iterative refinement by combining with estimated flow fields to generate higher-resolution flow estimations. The architecture of the context encoder is shown in figure 3

### 4.4 Correlation Block

The correlation volume is constructed to extract the visual similarities between the two images. The inner product is computed between all pairs of feature maps obtained from the feature encoder output. The output from the feature encoder is of the form  $g_\theta(I_1) \in \mathbb{R}^{H/8 \times W/8 \times 128}$  and  $g_\theta(I_2) \in \mathbb{R}^{H/8 \times W/8 \times 128}$ . The correlation  $\mathbb{C}$  is computed as a single matrix multiplication as defined:

$$\mathbb{C}(g_\theta(I_1), g_\theta(I_2)) \in \mathbb{R}^{H/8 \times W/8 \times 128}, \mathbb{C}_{ijkl} = \sum_h g_\theta(I_1)_{ijh} \cdot g_\theta(I_2)_{klh}$$

From the constructed 4D volume, average pooling with kernel size 2, 2 is applied in the last 2 dimensions to get a lower resolution volume. In our implementation 4 levels of average pooling is applied thereby creating a volume of  $\mathbb{C}^1 \in \mathbb{R}^{H \times W \times H \times W}$ ,  $\mathbb{C}^2 \in \mathbb{R}^{H \times W \times H/2 \times W/2}$ ,  $\mathbb{C}^3 \in \mathbb{R}^{H \times W \times H/4 \times W/4}$ ,  $\mathbb{C}^4 \in \mathbb{R}^{H \times W \times H/8 \times W/8}$  respectively. All the maps  $\mathbb{C}^1, \mathbb{C}^2, \mathbb{C}^3, \mathbb{C}^4$  are stacked as a pyramid. The intuition for performing average pooling is to capture large displacements between the frames, which can be obtained only at pooled resolution within pixel neighborhood. By doing average pooling, we are essentially bringing closer together the features that might have undergone larger displacements. During updating flow a small neighborhood of this pyramid is looked up and used.

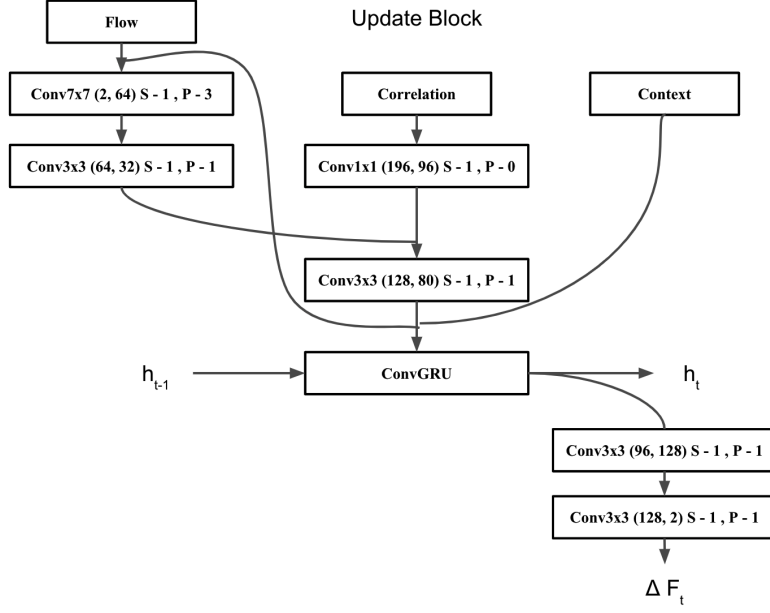


Figure 4: RAFT Architecture for the update block

#### 4.5 Update Block

An initial flow field is initialized to 0. The update block iteratively updates this flow field and uses Gated Recurrent Unit as its building block. A  $\Delta f$  is estimated during each iteration and is added to the previous flow as  $f_{k+1} = \Delta f + f_k$ . The update operation is analogous to the update step in the optimization problem.

The update block consists of a set of convolution layers that operate over the correlation features, previous flow estimates, and the output from the context encoder. Primarily the update block is a Gated Recurrent Unit which takes the inputs as flow, context, correlation features and the previous states. The architecture of the update block is shown in the figure. The update operations are defined as below:

$$\begin{aligned}
 Z_t &= \sigma(\text{Conv}_{3 \times 3}([h_{t-1}, x_t], W_z)) \\
 r_t &= \sigma(\text{Conv}_{3 \times 3}([h_{t-1}, x_t], W_r)) \\
 \bar{h}_t &= \tanh(\text{Conv}_{3 \times 3}([r_t \odot h_{t-1}, x_t], W_h)) \\
 h_t &= (1 - z_t) \odot h_{t-1} + z_t \odot \bar{h}_t
 \end{aligned}$$

where  $x_t$  is obtained by stacking flow, correlation, and context features from the previous update block. The output  $h_t$  is then passed through two convolution layers to obtain  $\Delta f$ .

The output from the update block is at  $1/8$  resolution of the original image. It is then upsampled to the original dimension using bilinear interpolation and the flow is then predicted as  $\in \mathbb{R}^{H \times W \times 2}$ .

#### 4.6 Loss Function

The loss between the estimated flow and the ground truth is calculated using the L1 distance metric. The loss is computed as an exponentially weighted average of the L1 loss between ground truth and the predicted flow at every stage. We have chosen the discount factor to be 0.8 in our case.

$$L = \sum_{i=1}^N \gamma^{N-i} \|f_{gt} - f_i\|_1$$

where  $N$  is the number of iterations used by the update block,  $f_i$  corresponds to the predicted flow field at the end of  $i$  iterations, and  $f_{gt}$  is the ground truth of the dataset.

#### 4.7 End Point Error

We have followed endpoint error (EPE) as the evaluation metric for comparing the performance of the RAFT network. EPE is defined as the Euclidean distance between the ground truth and the predicted flow. Large displacements in the predicted flow are ignored. A displacement of 400 pixels is considered as noise and ignored while computing the EPE metric.

### 5 Novelty

In this section, we discuss the two main novelties introduced in our approach for CrowdfLOW analysis using the RAFT-S network: the utilization of Charbonnier loss and the incorporation of self-attention. We first describe each novelty individually and then present the results of our experiments individually and combining both techniques.

#### 5.1 Charbonnier Loss

We replace the L1 loss function with the Charbonnier loss, also known as the pseudo-Huber loss, which is a smooth approximation of the L1 loss. It is commonly used in computer vision tasks, such as optical flow estimation and image reconstruction. The Charbonnier loss incorporates a small constant  $\epsilon$  to ensure differentiability. The loss is computed as the weighted sum of the square root of the squared differences between corresponding elements of the input tensors and  $\epsilon$ .

The Charbonnier loss is defined as follows:

$$L_{ch} = \sum_{i=1}^N \gamma^{N-i} \sqrt{(f_{gt} - f_i)^2 + \epsilon}$$

where  $N$  is the number of iterations used by the update block,  $f_i$  corresponds to the predicted flow field at the end of  $i$  iterations,  $f_{gt}$  is the ground truth of the dataset and  $\epsilon$  is a small constant added to the squared error to ensure numerical stability.

By minimizing this loss function, the model aims to reduce the overall difference between the predicted and target values while maintaining smoothness in the predictions. Using the Charbonnier loss instead of the L1 loss in crowdfLOW analysis using RAFT provides a smoother loss surface compared to the L1 loss. It also makes the network less sensitive to outliers compared to the L1 loss.

#### 5.2 Self-Attention for Global Context

Self-attention is a mechanism that allows a neural network to weigh the importance of different pixels in an image when making predictions. It captures the relationships between different elements in the image and assigns attention scores to them based on their relevance to each other. In the context of a RAFT-S network for optical flow estimation, by adding self-attention to the hidden dimension **cmaph** of the context encoder, it helps capture attended global context information by allowing the network to attend to different spatial locations and establish dependencies between them.

Mathematically, self-attention can be expressed as follows:

Given an input feature map  $X$  with dimensions  $H \times W \times C$ , where  $H$  represents the height,  $W$  represents the width, and  $C$  represents the number of channels, we can compute self-attention as follows:

$$Q = XW_Q \quad K = XW_K \quad V = XW_V$$

$$\text{Attention}(Q, K, V) = \text{softmax} \left( \frac{QK^T}{\sqrt{d_k}} \right) V$$

where:

- $Q$ ,  $K$ , and  $V$  represent the query, key, and value matrices, respectively, obtained by projecting the input feature map  $X$  onto learnable weight matrices  $W_Q$ ,  $W_K$ , and  $W_V$ .
- $d_k$  is the dimension of the key vectors.
- The attention scores are computed by taking the dot product between the queries  $Q$  and keys  $K$  and scaling it by  $\sqrt{d_k}$  to prevent the scores from becoming too large.
- The attention scores are normalized using the softmax function to obtain attention weights that sum up to 1.
- The final output is computed as the weighted sum of the values  $V$  using the attention weights.

By applying self-attention in a RAFT-S network, the model can effectively capture long-range dependencies and consider the global context of the input image when estimating optical flow. This helps improve the model’s ability to handle complex motion patterns and better capture the relationships between pixels in different regions of the image.

## 6 Experimental Setup

In this study, we evaluated the performance of the RAFT (Recurrent All-Pairs Field Transforms) algorithm, specifically training the RAFT-S variant with **1M learnable parameters** on the CrowdfLOW dataset. We focus on testing with the **IM02 (Static Dataset)** with a downsized images of resolution 384 x 512.

We initialized the training process by loading a **pretrained model** trained on the Flying Chairs dataset, aiming to enhance RAFT’s performance on CrowdfLOW. Further training with **300 epochs** and **12 iterations per epoch** was performed to fine-tune the pretrained weights.

During training, we used the following hyperparameters: a **learning rate of 1e-4**, a **gamma weighing factor of 0.9**, and a **weight decay of 2e-5**. The learning rate determined the step size during optimization, the gamma weighing factor adjusted the contribution of different loss terms during training, and the weight decay controlled the amount of L2 regularization applied to the network’s parameters.

Our experimental setup utilized **CUDA version 11.8** on an **RTX 1080 GPU** and accelerated the training process. To handle inconsistent GPU resources and allow resuming from intermediate stages, we implemented a checkpoint-saving strategy, creating checkpoints every 20 epochs.

The subsequent section will present the results of evaluating the RAFT-S variant with 1 million parameters on the CrowdfLOW datasets IM02.

## 7 Results

A set of 4 unseen images from the CrowdfLOW dataset was used to compare the performance of different variants of RAFT. Test metrics like EPE, and total test loss were recorded from the flow estimates for each individual network. The evaluation metrics for these 4 versions of RAFT are tabulated in Table 1 and 2.

- **Baseline:** Reimplemented the RAFT-S architecture from the original work.
- **Attention:** Self-attention was added to the image context encoder.
- **Charbonnier Loss:** The loss function was modified to Charbonnier loss instead of the L1 loss used in the baseline model.
- **Combined novelty:** Added Self-attention in the context encoder and Charbonnier loss was implemented simultaneously.

By obtaining crowd flow estimation using RAFT-S, we have evaluated the RAFT-S network on the benchmark CrowdfLOW dataset. We have obtained an End Point Error (EPE) of 0.209 on RAFT-S. We aim to use this set standard for evaluating our other models on the same set of test images.



Table 1: Test Results: End Point Error (EPE)

Model	Test 1	Test 2	Test 3	Test 4	Average
Basic RAFT-S	0.200	0.210	0.217	0.211	0.209
RAFT-S with Attention	0.201	0.183	0.202	0.172	<b>0.189</b>
RAFT-S with Charbonnier Loss	0.201	0.232	0.213	0.223	0.217
Attn and Char Loss	0.200	0.204	0.205	0.194	0.200

Table 2: Test Results: Test Loss

Model	Test 1	Test 2	Test 3	Test 4	Average
Basic RAFT-S	0.589	0.599	0.607	0.596	0.597
RAFT-S with Attention	0.584	0.582	0.566	0.576	<b>0.578</b>
RAFT-S with Charbonnier Loss	0.599	0.623	0.583	0.600	0.601
Attn and Char Loss	0.594	0.607	0.610	0.603	0.603

The evaluation results demonstrate that the attention model outperformed the base case on both evaluation metrics. This outcome aligns with our expectations, as incorporating attention in the context block enables the model to prioritize important image features for the update block. Figure 5 showcase the test case results for the RAFT-S with Attention network. However, using Charbonnier loss as a replacement for the standard loss function yielded worse results compared to the baseline. This indicates that Charbonnier loss has negative effects on the model, due to the variable motion vectors in a small pixel neighborhood, owing to the unaligned movement of humans.

To further investigate the effects of self-attention and Charbonnier loss, we combined both modifications and trained and tested the modified RAFT model. The evaluation metric for the combined modification was better than the base case, but it did not surpass the performance achieved by using self-attention alone for the context layer. This asserts the negative effect of the Charbonnier Loss on the entire mode.

In conclusion, we propose that implementing self-attention in the context layer has strong correlation with better performance in the EPE metric. It is worthwhile to note that in the original paper [3], removing the context layer did not affect the performance of the network on datasets like KITTI. However, our study demonstrates the importance of the context layer and how heuristic reasoning can provide better accuracy in flow estimation.

## 8 Future Work

One avenue for improvement is to increase the complexity of the network by utilizing the **RAFT model** which has **4.8 million** parameters. This enhanced model complexity can potentially improve its learning capacity and enable more accurate optical flow estimation.

Additionally, the crowdflow dataset used in this study involves continuous motion, making it suitable for exploring the application of consistency loss. Introducing **consistency loss**, which compares subsequent flow estimates, can promote smoother and more consistent optical flow estimations over time. We aim to investigate the impact of incorporating consistency loss within the RAFT-S and RAFT framework and evaluate its effectiveness on the crowdflow dataset.

Moreover, instead of downsizing the input resolution, utilizing the **full resolution (720 x 1280)** during training can capture more fine-grained details and subtle motion patterns. This approach has the potential to enhance the accuracy and precision of the estimated optical flow.

Drawing inspiration from **Feature Pyramid Networks (FPNs)** [17], integrating FPNs into the RAFT architecture can help capture both local and global motion information. This integration can lead to improved performance in handling scale variations and enhance the overall accuracy of optical flow estimation.

Finally, exploring a training strategy that incorporates utilizing estimated flow to warp or transform the first input image (image 1) using the estimated flow vectors. The resulting transformed image, known as the rendered image, can then be compared with the second input image (image 2) to

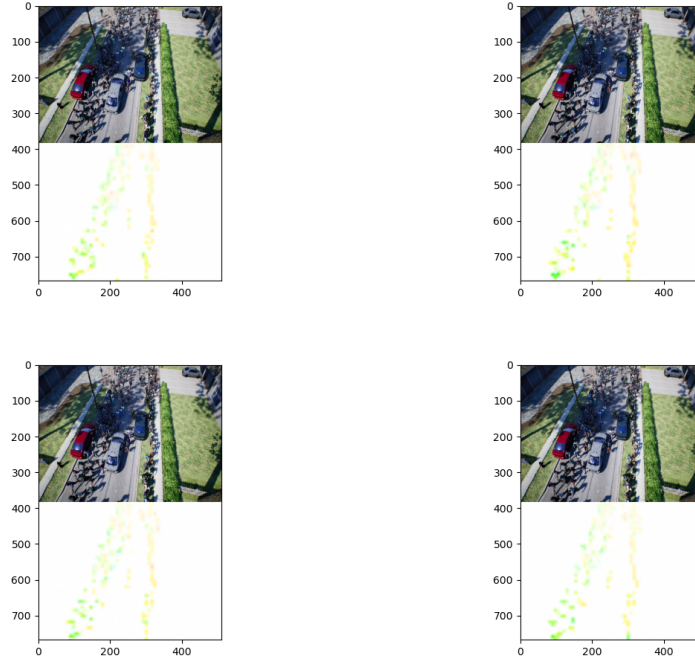


Figure 5: Optical Flow Estimate: RAFT-S with Self-Attended Context

compute a loss. By backpropagating this loss, the model can learn to produce flow estimates that generate visually consistent rendered images, thereby improving the overall quality of optical flow estimation.

By pursuing these future research directions, the RAFT model can be further refined to achieve more accurate, robust, and efficient optical flow estimation in the context of crowd monitoring and analysis.

## 9 Conclusion

In conclusion, this study evaluates the application of the RAFT-S network for optical flow estimation in crowd scenes. The results highlight the effectiveness of incorporating self-attention in the context layer, improving flow estimation accuracy. However, replacing the standard loss function with Charbonnier loss leads to inferior results, likely due to the unaligned movement in crowd scenes. The combined modification does not surpass the performance achieved. Overall, the study underscores the potential of RAFT-S with self-attention for addressing optical flow estimation challenges in crowd scenes, contributing to advancements in crowd dynamics understanding and enabling various real-world applications.

## 10 Supplementary Material

Github Repository - <https://github.com/Ritika9494/ECE285-OpticalFlow/tree/main>

Video Presentation - <https://youtu.be/97UIxLAJ1r8>

## References

- [1] B. K. Horn and B. G. Schunck, “Determining optical flow,” *Artificial Intelligence*, vol. 17, no. 1, pp. 185–203, 1981.

- [2] P. Muller and A. Savakis, “Flowdometry: An optical flow and deep learning based approach to visual odometry,” in *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, 2017, pp. 624–631.
- [3] Z. Teed and J. Deng, “Raft: Recurrent all-pairs field transforms for optical flow,” 2020.
- [4] G. Schröder, T. Senst, E. Bochinski, and T. Sikora, “Optical flow dataset and benchmark for visual crowd analysis,” in *IEEE International Conference on Advanced Video and Signals-based Surveillance*, 2018.
- [5] B. D. Lucas and T. Kanade, “An iterative image registration technique with an application to stereo vision,” in *International Joint Conference on Artificial Intelligence (IJCAI)*, vol. 2, 1981, pp. 674–679.
- [6] B. K. Horn and B. G. Schunck, “Determining optical flow,” *Artificial Intelligence*, vol. 17, no. 1-3, pp. 185–203, 1981.
- [7] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazirbas, V. Golkov, P. van der Smagt, D. Cremers, and T. Brox, “Flownet: Learning optical flow with convolutional networks,” in *IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 2758–2766.
- [8] D. Sun, X. Yang, M.-Y. Liu, and J. Kautz, “Pwc-net: Cnns for optical flow using pyramid, warping, and cost volume,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2018, pp. 8934–8943.
- [9] A. Ranjan and M. J. Black, “Optical flow estimation using a spatial pyramid network,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2017, pp. 2720–2729.
- [10] T. Hui, X. Wang, C. Gao, Y. Zhang, C. Ma, Z. Liu, and W. Sun, “Liteflownet: A lightweight convolutional neural network for optical flow estimation,” in *European Conference on Computer Vision (ECCV)*. Springer, 2018, pp. 484–499.
- [11] Z. Liu, H. Li, Y. Luo, Z. Liu, J. Huang, and Y. Ma, “Smurf: Self-teaching multi-frame unsupervised raft with full-image warping,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2021, pp. 6022–6031.
- [12] N. Mayer, E. Ilg, P. Hausser, P. Fischer, D. Cremers, A. Dosovitskiy, and T. Brox, “A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 4040–4048.
- [13] V. Lempitsky and A. Zisserman, “The flyingthings3d dataset: A photorealistic synthesized dataset for studying shape, motion, and stereo,” *International Journal of Computer Vision*, vol. 128, no. 3, pp. 824–834, 2020.
- [14] M. Menze, A. Geiger, R. Gomez-Ojeda, and C. Heipke, “The kitti vision benchmark suite: A challenge for autonomous driving,” in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 8493–8502.
- [15] 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 (ECCV)*. Springer, 2012, pp. 611–625.
- [16] J. Luiten, N. Tishby, and M. Barz, “CrowdfLOW: A comprehensive benchmark dataset for crowd analysis and understanding,” in *International Conference on Computer Vision (ICCV)*, 2022.
- [17] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, “Feature pyramid networks for object detection,” 2017.