MOTION AND ACTIVITY UNDERSTANDING IN 360° VIDEOS: AN EGOCENTRIC PERSPECTIVE

by

Keshav Bhandari, BSc.CSIT.

A dissertation submitted to the Graduate College of
Texas State University in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
with a Major in Computer Science
August 2022

Committee Members:

Yan Yan, Chair

Ziliang Zong, Co-Chair

Anne Hee Hiong Ngu

Lawrence V. Fulton

FAIR USE AND AUTHOR'S PERMISSION STATEMENT

Fair Use

Duplication Permission

DEDICATION

To my dearest wife, finally, we can have a baby.

ACKNOWLEDGEMENTS

This endeavor would not have been possible without the continuous support of my advisor Dr. Yan Yan, co-advisor Dr. Ziliang Zong, and committee members, Dr. Anne H.H. Ngu and Dr. Lawrence V. Fulton. I am also grateful to the Department of Computer Science at Texas State University for providing me with an opportunity to accomplish my academic goals.

Dr. Yan Yan has been a great support as an excellent advisor and a good friend whose continuous moral and academic support has fueled my passion for achieving my Ph.D. Similarly, my deepest gratitude goes to Dr. Ziliang Zong for all the valuable discussion, brainstorming, continuous guidance and excellent academic support. I am also thankful to Dr. Anne H.H. Ngu for providing me an opportunity to involve in research project other than my dissertation, which helped me expand my knowledge and connections with people like Dr. John Milton, and Dr. Joshua T. Chang.

I want to acknowledge all faculty members and professors who directly and indirectly helped me during this journey. I would like to acknowledge the support from Dr. Martin Burtscher for the parallel computing course, which helped me a lot in my research projects. I am deeply indebted to my colleague Mr. Bin Duan, Mr. Cody Blakeney, Mr. Gentry Atkinson, Mr. Bibek Aryal, and other lab members for their continuous support as a friend, mentors and collaborators. I would be remiss in not mentioning Karen A. Hollensbe, Michelle P. Hageman, and other administrative members for helping me with administrative workloads, along with Jerry Rosado and CS Help Desks for technical support.

Finally, words cannot express my gratitude to my dearest wife, Anar Niroula, and my family for their continuous emotional support.

TABLE OF CONTENTS

Page

LIST OF TABLES

Table **Page**

LIST OF FIGURES

Figure **Page**

ABSTRACT

Activity recognition is one of the popular fields in computer vision research. Though recent development in deep learning-based methodologies has shown tremendous progress in traditional video-based activity recognition, 360° activity recognition is still in its infancy. 360°-based activity recognition imposes challenges like lack of datasets, domain-specific frameworks, and motion understanding in 360° videos. This research focuses on two critical aspects of activity recognition in 360° videos from an egocentric perspective, (i) Egocentric activity recognition and (ii) Motion understanding in 360° videos. Under (i) Egocentric activity recognition, we present two important works, EgoK360 and VIT360. EgoK360 is an egocentric kinetic human activity dataset comprised of human activities and smaller actions from a first-person view. The dataset intends to fill the gap of egocentric video-based activity recognition in 360° videos context. Similarly, VIT360 is a rotation-invariant activity recognition model designed using representation learning techniques and current transformer-based techniques in the literature. Similarly, under (ii) Motion understanding in 360° videos, we present three important works LiteFlowNet360, FLOW360, and SLOF. LiteFlowNet360 is a domain adaptation framework for transferring perspective videos based motion estimation techniques to 360° videos settings. On the other hand, we present FLOW360, a perceptually natural synthetic optical flow dataset for motion understanding in 360° videos. This dataset is the first in the literature, leveraging several newer opportunities in this domain. Finally, we present SLOF, a siamese representation learning-based framework for motion estimation in 360° videos. We will also discuss several challenges in these areas and the contribution of our work compared with the state-of-the-art frameworks.

I. INTRODUCTION

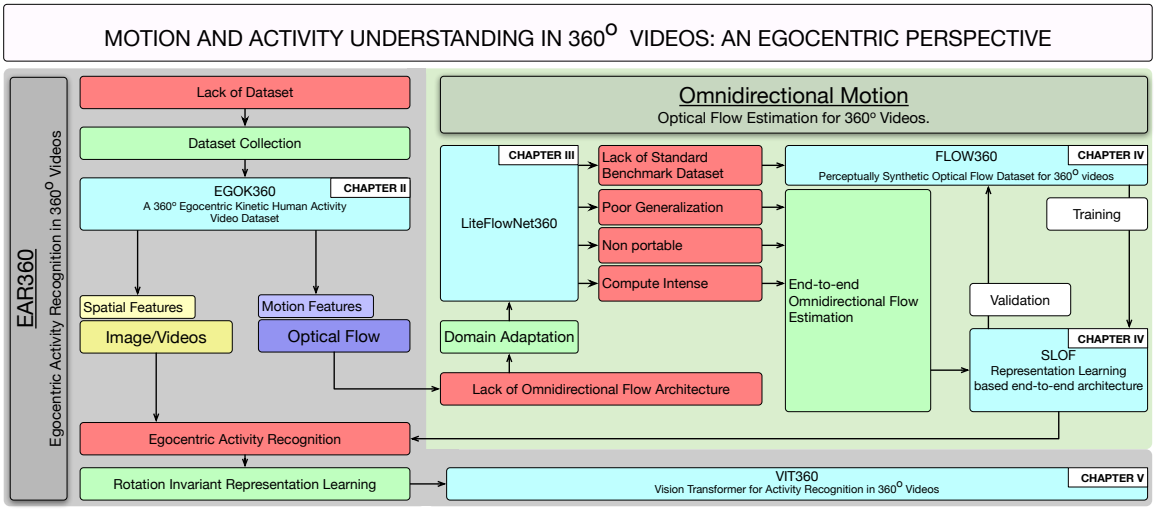


Figure 1.1: **Framework of the dissertation.** The design of overall dissertation is divided into two main sections: (i) Activity Recognition and (ii) Omnidirectional Motion Estimation in 360° videos. Chapter II & IV is focused on activity recognition and Chapter III & V is focused on omnidirectional motion estimation.

The idea of 360° imaging dates back to 1787 when English painter Robert Barker coined the term "panorama" for his paintings displayed on Leicester Square, London [1]. The awe moment of the visitors who visited the display would soon change with the invention of digital photography by Frenchman Joseph Nicéphore Nièpce. With a series of iterations, advanced 360° panoramic cameras started hitting the market in the 1980s, changing how we capture the world. On the other hand, as general photography advanced and became more accessible, the amount of data generated over time compounded significantly. These data would impact areas of machine learning (or deep learning), so-called a "hype" in computer/information science to be one of the important fields of 21^{st} century. Today, we have seen a massive success of deep learning in computer vision applications advancing fields like health, e-commerce, autonomous driving, surveillance, security, and many more. Compared to general imaging, the development and accessibility of 360° imaging

technology paced slower over time, resulting in computer vision applications less vivid in 360° domain. However, the current trend of increasing interest in 360° imaging in several applications like vlogging, surveillance, gaming, Augmented Reality (AR), Virtual Reality (VR), and many more is attracting researchers to explore the 360° based imaging for computer vision applications. This dissertation focuses on one such computer vision application for 360° videos called as motion and activity understanding.

360° imaging compared to regular or "perspective" imaging provides an infinite field of view, making it superior in applications where a complete understanding of the environment (as it unfolds in the real world) in images/videos is essential. Several applications we mentioned above benefit from 360° whenever the omnidirectional aspect is considered. E.g., in autonomous driving, omnidirectional information is crucial to achieving human-like maneuverability where synchronization of disjoint sensors will add an extra layer of complexity. A similar example can be seen in drone navigation, intelligent surveillance, gaming, and AR/VR. There are several use cases where 360° videos make more sense than regular videos, given omnidirectional information is crucial; however, this dissertation only explores a narrow aspect focusing on activity recognition and motion understanding in 360° videos.

The focus on egocentric activity recognition on 360° videos is motivated for surveillance and security. E.g., this dissertation establishes a proof of concept on how computer vision research can be used to understand activity performed by an agent in 360° videos. This finding would be necessary for several real-world applications where humans are involved in performing dangerous jobs in the field. To maintain the security and safe working environment understanding the omnidirectional world and actions performed by the agent is equally important. This can be achieved by implementing computer vision software to alarm the hazardous situation and

prompt the emergency aid. Similarly, another application area we think is relatable to life-vlogging, where understanding the events could be important for several downstream tasks like summarizing the vlog, surveillance, and security.

Crafting an intelligent computer vision software or model for 360° videos impose several challenges like lack of dataset, motion representation (360° optical flow, or omnidirectional flow) and need of reliable frameworks for egocentric activity recognition in 360° videos. The organization of the overall dissertation, as shown in Fig. 1.1 highlights two major aspects of this dissertation: (i) Understanding activity and (ii) Understanding Motion. The remainder of this dissertation is organized as follows:

- In Chapter II, we propose a novel egocentric activity recognition dataset for 360° videos, EGOK360. This dataset is the first of its kind to address omnidirectional egocentric activity recognition.

  Summary: Recently, there has been a growing interest in wearable sensors which provides new research perspectives for 360° video analysis. However, the lack of 360° datasets in literature hinders the research in this field. To bridge this gap, in Chapter II, we propose a novel Egocentric (first-person) 360° Kinetic human activity video dataset (EgoK360). The EgoK360 dataset contains annotations of human activity with different sub-actions, *e.g.*, activity Ping-Pong with four sub-actions which are pickup-ball, hit, bounce-ball and serve. To the best of our knowledge, EgoK360 is the first dataset in the domain of first-person activity recognition with a 360° environmental setup, which will facilitate the egocentric 360° video understanding. We provide experimental results and comprehensive analysis of variants of the two-stream network for 360 egocentric activity recognition. The EgoK360 dataset can be downloaded from `https://egok360.github.io/`.

- In Chapter III, we present a novel framework for 360 optical flow estimation, dubbed as LiteFlowNet360 as an adaptation of existing best practices from "optical flow estimation for perspective videos" and "spherical convolution for 360 videos/images".

  Summary: Nowadays 360 video analysis has become a significant research topic in the field since the appearance of high-quality and low-cost 360 wearable devices. In Chapter III, we propose a novel LiteFlowNet360 architecture for 360 videos optical flow estimation. We design LiteFlowNet360 as a domain adaptation framework from perspective video domain to 360 video domain. We adapt it from simple kernel transformation techniques inspired by Kernel Transformer Network (KTN) to cope with inherent distortion in 360 videos caused by the sphere-to-plane projection. First, we apply an incremental transformation of convolution layers in feature pyramid network and show that further transformation in inference and regularization layers are not important, hence reducing the network growth in terms of size and computation cost. Second, we refine the network by training with augmented data in a supervised manner. We perform data augmentation by projecting the images in a sphere and re-projecting to a plane. Third, we train LiteFlowNet360 in a self-supervised manner using target domain 360 videos. Experimental results show the promising results of 360 video optical flow estimation using the proposed novel architecture.

- In Chapter IV, we present the first perceptually natural-synthetic benchmark dataset for omnidirectional flow estimation, FLOW360 and representation learning based omnidirectional flow estimation framework, SLOF.

  Summary: Optical flow estimation in omnidirectional videos faces two

significant issues: the lack of benchmark datasets and the challenge of adapting perspective video-based methods to accommodate the omnidirectional nature. In Chapter IV we propose the first perceptually natural-synthetic omnidirectional benchmark dataset with a 360° field of view, FLOW360, with 40 different videos and 4,000 video frames. We conduct comprehensive characteristic analysis and comparisons between our dataset and existing optical flow datasets, which manifest perceptual realism, uniqueness, and diversity. To accommodate the omnidirectional nature, we present a novel Siamese representation Learning framework for Omnidirectional Flow (SLOF). We train our network in a contrastive manner with a hybrid loss function that combines contrastive loss and optical flow loss. Extensive experiments verify the proposed framework's effectiveness and show up to 40% performance improvement over the state-of-the-art approaches. We will publish the FLOW360 dataset with all raw Blender scenes and Blender add-ons for researchers to create custom optical flow datasets for perspective and omnidirectional videos.

- In Chapter V, we present VIT360 - a vision transformer based network pretrained with siamese representation - to achieve rotational invariance in 360° videos for egocentric activity recognition.

  Summary: Research on egocentric activity recognition mainly focuses on improving accuracy using sophisticated architectures based entirely on convolution layers. However, when it comes to 360° videos, egocentric activity recognition imposes three significant challenges: i) Performance deterioration due to random field-of-view projection; ii) Radial distortions caused by sphere-to-plane projections; and, iii) Overheads of model adaptation, transformation and refinement for 360° optical flow. In Chapter V, we propose Vision Transformer 360 (VIT360) to address

aforementioned issues. This work shows how accuracy-centric convolution-based architecture performs better for fixed field-of-view projection but fails in the case of random field-of-view projection. VIT360 is rotationally invariant, and therefore does not suffer from performance degradation when the field-of-view projection is randomized. In addition, VIT360 mitigates radial distortions as it learns features on tangential patches. VIT360 provides the foundation to leverage motion inference techniques from off-the-shelf optical flow architecture. Although this chapter focuses on egocentric activity recognition, the projection invariant properties of VIT360 can be applied to other applications involving 360° videos as well.

To summarize, the contributions of this dissertation are as follows:

- Egocentric Activity Recognition (EAR) in 360° videos opens several new oppurtinities in areas like surveliance, security, navigation, video understanding, and AR/VR. However, the lack of datasets has hinders the development of the field. We propose an egocentric activity dataset, EgoK360 to facilitate the advancement of the field.

- Motion understanding in 360° videos is crucial for many vision related downstream applications like action/activity recognition, motion anticipation, video summarization & understanding, computational videography/photography, and so on. Following are the contributions we made for motion understanding in 360° videos

    1. **LiteFlowNet360.** This framework advocates best practices for exploiting off-the-shelf optical flow architecture for estimating optical flow for 360° videos. LiteFlowNet360 is basically a multi-stage domain adaptation framework involving transformation of tradional CNNs to

spherical CNNs, fine-tuning the transformed optical flow framework via supervised training with augmented dataset and domain transfer via self-supervised training on target dataset.

2. **SLOF.** A siamese representation learning based framework for motion understanding in 360° videos. SLOF overcomes the challenges appeared in LiteFlowNet360 by eliminating the requirement of laborous task involving multi-stage domain adaptation of existing off-the-shelf framework.

3. **FLOW360.** Availibility of a reliable dataset is the first important steps for the advancement of any domain. Optical flow estimation in 360° or omnidirectional flow estimation domain lacks reliable standard benchmark dataset for optical flow estimation. Following the best practices from the state-of-the-art persepctive videos based optical flow dataset, Sintel [2], we propose a perceptually natural synthetic dataset for omnidirectional flow estimation to facilitate the advancement of motion understanding in 360° videos.

4. **Transformer Based Flow Inference.** This approach exploits best practices in transformer based optical-flow framework PercieverIO [3] to infer optical flow in different rotational view of 360° videos and later combining the results into final optical flow.

- Prior practices focusing on modifications of existing architecture via re-adjustments (replacing traditional CNNs with spherical CNNs) and refinement for computer vision applications in 360° videos results in several complications like difficulty in portability, over-parameterization and multi-stage training/inference protocols. We present techniques involving representation learning and transformer based frameworks to mitigate such

overheads. These techniques results in better performance with simpler design resulting in easy deployment, training and inference.

1. **SLOF.** Siamese representation Learning for Omnidirectional Flow (SLOF) focuses only on learning the spherical nature of 360° data via training the existing architecture with different representation view of same input and maximizing the latent similarity across rotationally augmented views of same 360° videos input. Design of SLOF eliminates the overheads of network re-adjustments and multi-stage training protocols making it effective in deployment settings with improved accuracy.
2. **VIT360.** VIT360 extends Vision Transformer [4] framework for activity recognition in 360° videos. The design of VIT360 ensures two fold adaptation on 360° videos, (i) siamese representation learning as pre-training stage and (ii) instead of considering entire view as an input VIT360 takes multiple tangential planes covering the 360° field-of-view ensuring the trivial amount of distortions. VIT360 shows the efficacy of proposed architecture with representation learning and shows potential of such design to other vision related task for 360° videos.

II. EGOK360: A 360° EGOCENTRIC KINETIC HUMAN ACTIVITY VIDEO DATASET

Introduction

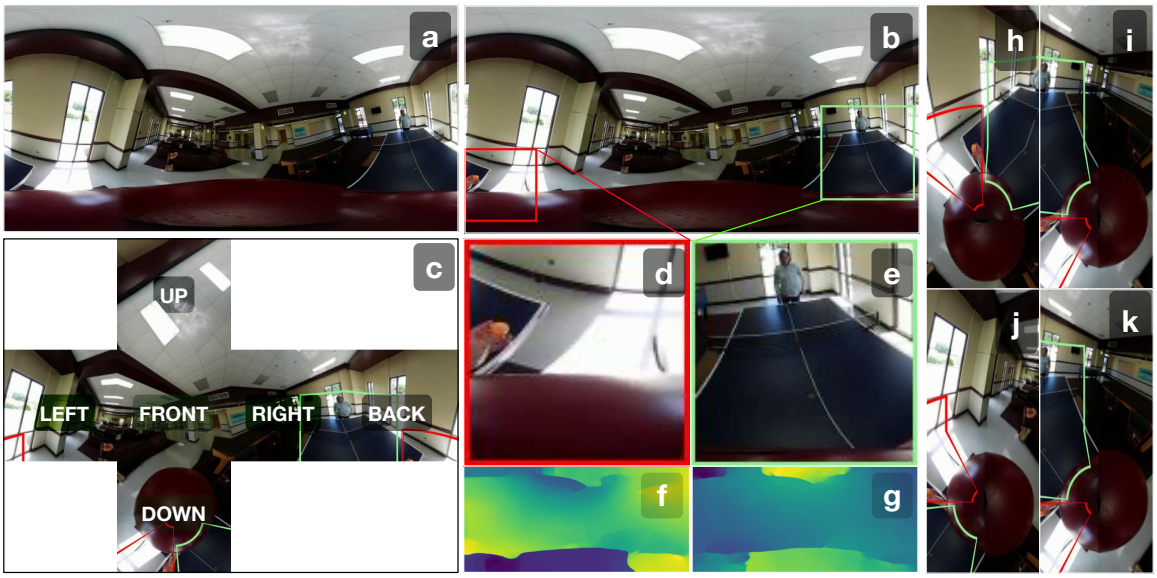


Figure 2.1: **Sample video frames from EgoK360 dataset.** (a,b) Consecutive frames (I_i, I_{i+1}) for action "Serve" from "Ping-Pong" activity in equirectangular projection. (c) Cubemap projection of (b) showing six different cubic faces. (d,e) Cropped section of wearer showing action 'serving' (red box in (b)) and front-view from wearer's perspective (green box in (b)). (f,g) Optical flow $(\vec{u}_i, \vec{v}_i)$. (h,i,j,k) Normal field-of-view for front-down, back-down, left-down and right-down.

Wearable devices like Apple smartwatch, GoPro and Google Clip, have been widely used in our daily life nowadays. Meanwhile, the appearance of 360° cameras and the growing services on social media platforms such as Facebook and YouTube are changing the way how we consume multimedia. Having the advantage of 360 field-of-view over perspective videos from traditional cameras, 360° cameras have the superiority in many applications such as self-driving cars, virtual-reality, life-logging, augmented reality, film-making and surveillance [5, 6]. The popularity of 360° videos is also changing computer vision and virtual reality research area

recently. Egocentric Activity Recognition (EAR) from videos is one of such fields. However, to the best of our knowledge, there is no public 360 egocentric human activity dataset in literature.

In this chapter, we propose a novel 360 egocentric human activity recognition (EgoK360) dataset. The EgoK360 dataset is inspired by action recognition datasets such as UCF-101 [7], HMDB-51 [8] and Kinetics [9]. Our EgoK360 dataset contains three different types of actions: Person-Person, Person-Object, and Singular actions. These categories of actions can be described in the following manner. Person-Person actions involve two or more people interacting with each other such as hugging and speaking with someone; Person-Object actions refer to a person interacting with some objects such as picking up something or moving something from one location to another; Singular person actions involve a single person performing some actions independent of others such as reaching towards something or combing hair. In this chapter we perform experiments with two popular action classification deep nerual networks on our introduced EgoK360 dataset, *i.e.*, two-stream network [10] and Inflated 3-Dimensional network (I3D) [9]. In the following sections, we present the related work, datasets, experimental results and conclusions.

Related Work

Action recognition datasets such as HMDB [8], UCF101 [7] and Kinetcs [11] are widely used in literature. They are captured by perspective cameras (single field-of-view) and have limitations in terms of applications. Singh et.al., [12] use a novel dominant motion feature derived from optical flow for egocentric action recognition and also propose a convolutional neural networks (CNN) [13] for end-to-end training. Xia et.al., [14] present a framework to analyze RGBD videos captured from a robot for activity recognition. Lee et.al., [15] present an egocentric video summarization approach by identifying important people and object in the

video. Two-stream network is the popular architecture in literature for action recognition, such as Two-stream Convnet [10] and Inflated 3D ConvNet (I3D) [9]. I3D architecture is the state-of-the-art in the two-stream genre for action recognition.

In recent years, a few 360° datasets [16, 17, 18, 19, 20, 21, 22, 16, 6] appeared in the applications such as autonomous driving, human-computer interaction, virtual reality, and others. However, they are target to different applications other than Egocentric Activity Recognition in 360° field-of-view (EAR360).

Meanwhile, in the egocentric action recognition field, popular datasets such as Epic Kitchens [23], EgoHands [24], EGTEA Gaze+ [25] are perspective video datasets with a person interacting with an object or another person. Similarly, large-scale datasets such as Charades-Ego [26] contains both the first-person and third-person videos. Pirsiavash et.al., [27, 28, 29, 30, 31] present an egocentric dataset for understanding activities and the context in the video. However, all these datasets in literature are only limited to perspective videos.

EgoK-360 Dataset

Our EgoK-360 dataset contains activity classes that represent all three categories, *i.e.,* Person-Person, Person-Object, and Singular actions. There are a few differences in the video content compared with other datasets because of the properties of Egocentric 360 videos. Given that the footage encompassing the dataset captured from an egocentric perspective, the Person-Person actions would involve the interaction between the wearer and other people. This differs with Person-Person actions captured in the traditional third-person perspective cameras. Likewise, a Person-Object action such as bouncing a ping-pong ball on a table would only be identified when the particular action was performed by the wearer. Most action classes in EgoK360 are in the Person (singular) category because the

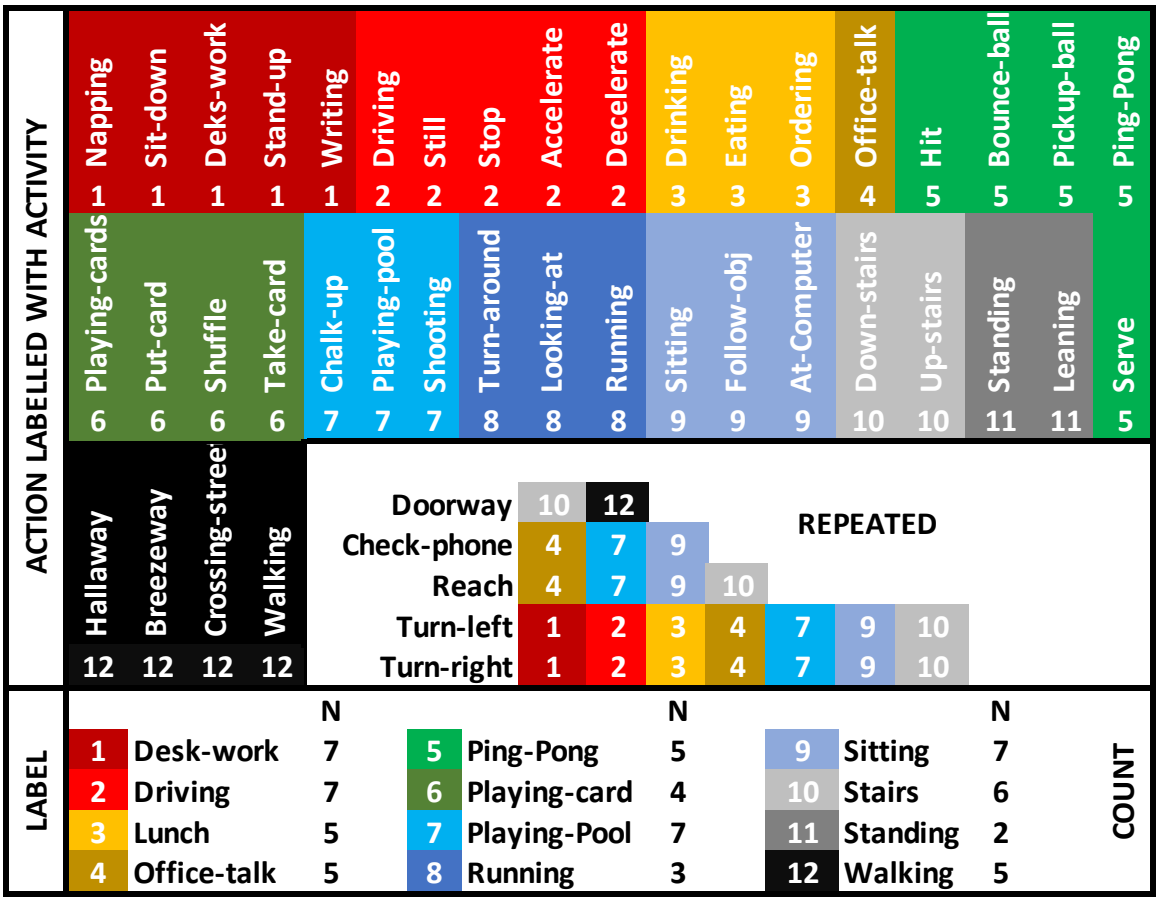


Figure 2.2: **Activities and action classes in EgoK360.** Actions are colored and numbered with corresponding activity. The same actions may appear in different activities.

egocentric perspective inherently privatizes the action or content recorded.

The 360 field-of-view naturally makes everything egocentric in EgoK360. Egocentric actions entirely depend on the field-of-view where a wearer (first-person) is engaged. Meanwhile, the rest of fields-of-view that he/she is not engaged are irrelevant for action recognition. The significant contribution to action recognition is the wearer's egocentric view and his/her engagement in actions.

Activity/Action Classes

We show action/activity instances of EgoK360 in Fig. 2.2. Our dataset contains 12 activities and 45 actions, collectively making 63 activity-action unique cases. An

activity is defined as collections of shorter actions. For example, an activity 'driving' is composed of actions such as accelerate, decelerate, idle, stop, driving, turn-left and turn-right. Action classes such as turn-left, turn-right, reach, doorway and check-phone are frequently occurring actions. However, there is a significant difference in these actions depending upon the category of activity. For example, turn-left in driving is completely different than turn-left in activity office-talk. We collected 127 videos with approximately 11 minutes each.

EgoK-360 Characteristics

The EgoK-360 dataset has its uniqueness of 360 fields-of-view, egocentric and kinetic properties. We discuss the following characteristics of EgoK-360 dataset in terms of its diversity, statistics and properties.

Diversity. Our EgoK-360 dataset contains common different activities in daily life. Around 11% of actions (such as turn-left, turn-right, reach, check-phone and doorway are frequent actions) are overlapping actions. Activity such as desk-work, driving, playing-pool and running have the most number of actions. Activity such as standing has the least number of actions.

Properties. We present sample frame in Fig. 2.1. The dataset is a collection of videos from a 360 camera projected on the 2D plane using equirectangular projection, as shown in Fig. 2.1 (a and b). The frames size is 640x320. Frames exhibit huge distortion as shown in Fig. 2.1 (b-c, h-k) using red and green bounding box), making it challenging for regular convolution. We calculate optical flows using FlowNet [32].

Experiment

We conduct our experiments using two-stream and I3D networks. We implement two-stream architecture with resnet-101 model pre-trained on UCF101

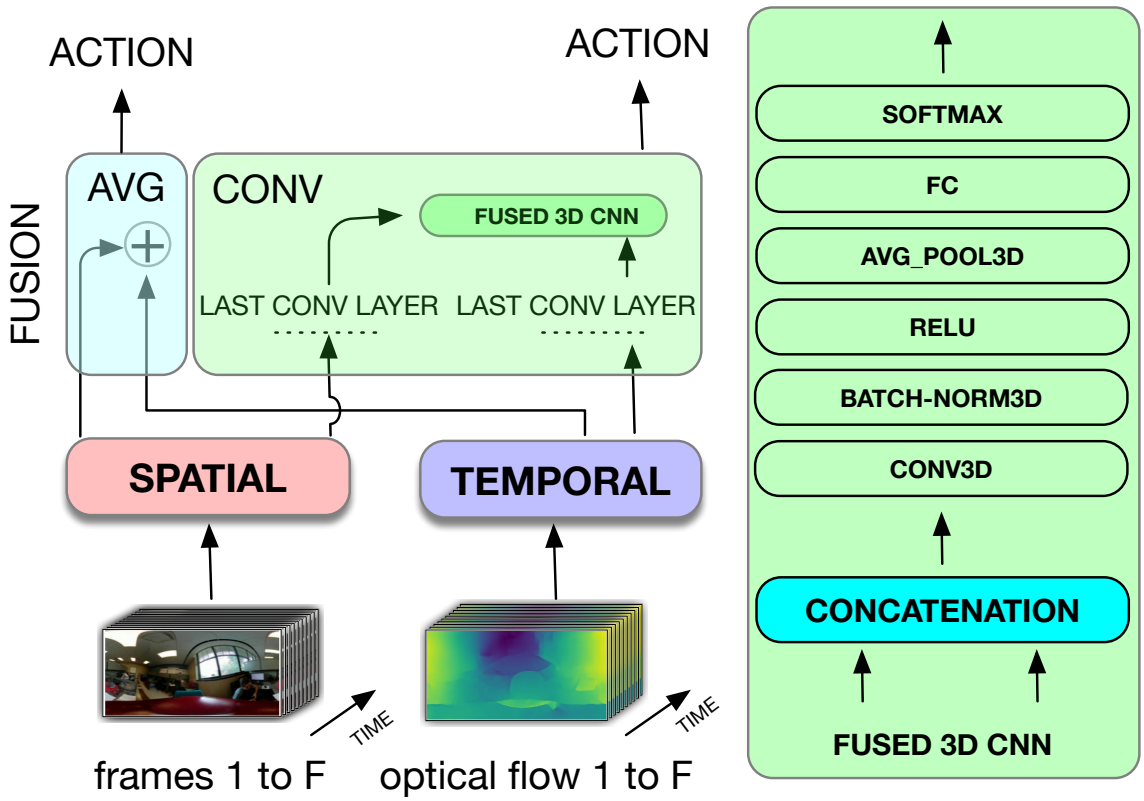


Figure 2.3: **Spatio-temporal architecture for action/activity classification.** We implement resnet-101 and I3D architecture. Average and convolution fusion are adopted. For average fusion we simply average probabilities of two networks and map them into single probability. For convolution fusion, we concatenate (depth-wise) output from last convolution layer and feed to the convolution module.

which outperforms state-of-the-art I3D model. EgoK360 exhibits complexity of spherical representation of 360° video on 2D plane (equirectangular projection) which makes challenge for these models to prioritize a significant field-of-view responsible for the wearer's engagement in certain actions and makes difficult to train. Therefore, 3D-representation of the video does not perform well in the 360 environment.

Implementation Details

We adopt the network architecture as shown in Fig. 2.3 in our experiments. Our model inputs are consecutive frames. Videos are down-sampled in the rate of 10 fps. We calculate optical flows beforehand using FlowNet [32]. We adopt the two-stream and I3D architectures with average and convolution fusion. For two-stream architecture, video is represented as 2D inputs with $[N \times F_c \times H \times W]$ dimensions. For I3D architecture, video is represented as 3D input with $[N \times C \times F \times H \times W]$ dimensions, where $F_c = F \times C$. Here F is the number of frames, N is the batch-size, C, H and W are channel, height and width of the frames.

Two-stream Architecture

Residual learning framework [33] provides convenient optimization and rapid high accuracy as network becomes deeper. With this in mind, we change the two-stream architecture by replacing the spatial and temporal network with resnet-101 model pre-trained on UCF101. We use size of 10 to stack frames in sequence for both spatial and temporal networks. This brings the channel size changing from 3 to 20 in the temporal network and to 30 in the spatial network. The resnet-101 requires input as 3 channel images. To fix this, we use the method in the cross-modal learning [34]. We do not observe better results compared to UCF101 implemented with the same architecture, which achieves at least 80% accuracy.

I3D Architecture

We implement I3D architecture as proposed in [9]. I3D architecture relies on 3D receptive fields for video representation. Spatial and temporal network receive an input of 3 and 2 for the channel size respectively, along with depth of 10. The original idea in [9] using the entire video as one training sample. However, we do not

Table 2.1: **EgoK360 quantitative results.** Experimental results of EgoK360 datasets on two-Stream (modified version with trained Resnet-101 on UCF101) and I3D Architecture. (*Top accuracy in bold*)

Architecture	Network Mode							
	flow		rgb		avg_fused		conv_fused	
	Activity	Action	Activity	Action	Activity	Action	Activity	Action
Resnet	73.94	61.09	**77.05**	58.22	76.53	62.44	74.71	56.87
I3D	57.24	43.4	74.13	55.31	68.74	50.88	**74.47**	56.63

achieve performance increase on EgoK-360 dataset. We run experiments with the depth of 10 which is the optimal for our case.

Fusion

In this chapter we use both average and convolution fusion techniques. In average fusion, we take an average of probabilities from the last layers. For convolution fusion, we implement convolution-module inspired by [33]. We use the output from the last convolution layer and concatenate the features which later fed into the fusion convolution module. We freeze our spatio-temporal module and train the fusion layer. We can also train the spatio-temporal network along with the fusion layer.

Results

We present our experimental results in Table 2.1 and visualization in Fig. 2.4. We observe that activity classification accuracy is higher than action classification accuracy as shown in Table-2.1. The range of activity and action classification accuracy in I3D architecture is higher than in two-stream architecture.

The average fusion is remarkably better than the convolution fusion in our case. This quantitative results can be explained using Fig-2.4 visualization. We observe interesting results on two different architectures. Fusion techniques make a huge

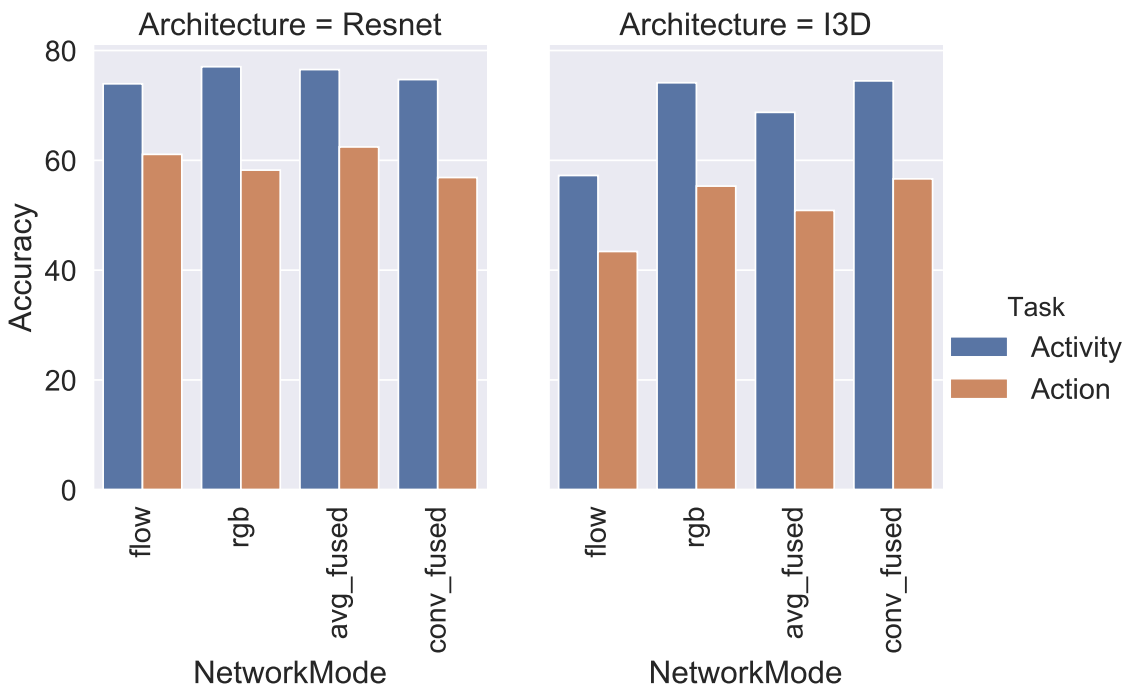


Figure 2.4: **Comparing quantitative results on EgoK360.** Visualization results of Table 2.1. The figure shows accuracy for each architecture and classification mode (activity *vs* action). Overall 'action' classification is better in resnet-101. We observe convolution fusion in resnet architecture and I3D makes significance difference in flow (temporal) and rgb (spatial) stream. Similarly, temporal-stream performs better in resnet compared with I3D architecture. Conv_fusion has same effect on both cases where as avg_fusion comparatively improve resnet architecture. In general resnet architecture shows consistent metrics relative to I3D arhcitecture.

difference in action/activity classification in resnet whereas spatial and temporal streams have significant differences in I3D architecture. From Fig. 2.4 we can infer that convolution fusion performs better whenever two streams have significant gap in accuracy.

We also investigate how well this two-stream architecture generalizes with our dataset. We use the technique presented in [35] to visualize the activation map. We show the activation map with a randomly selected action in Fig. 2.5. We derive these activation maps from the I3D and two-stream spatial network.

These activation maps represent salient features learned by the model. The

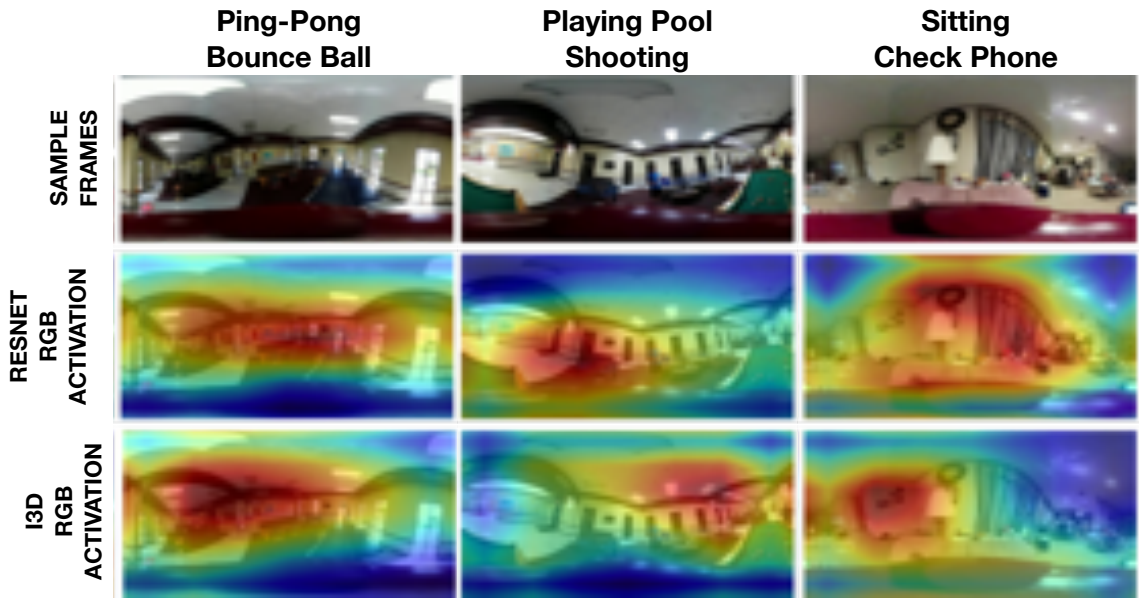


Figure 2.5: **Comparision of features learned on EgoK360.** Activation map showing salient features learned by the spatial network. The top row shows RGB frames, the second row represents activation map from two-stream networks, and the bottom row shows the activation map from I3D model.

model infers most edges as trivial regions, as the dataset has massive distortion near edges. We can visually inspect and analyze this behavior. For example, in the Fig. 2.5 activity Bounce_ball (playing Ping-Pong activity), the salient features are away from the actual region where a person wearing a camera is bouncing a ball. This region lies on the left-bottom-corner and has massive distortion. It is nearly impossible to judge the meaning of these activation maps accurately. However, if we carefully inspect the salient features learned by both architectures, we can conclude that the model is inferring the action classification task from other features rather than salient features as expected. The reason behind this poor response of the model is due to the naïve convolution, which is not rotation invariant. Features on EgoK360 have different spatial properties depending upon the position in equirectangular plane. This can be improved with techniques such as [36, 37].

Conclusion

This chapter introduces EgoK360 dataset with annotations of 63 unique activity and action classes. This dataset is challenging because of distortion, wide field-of-view and activities/actions properties. We implement two popular two-stream architectures in the experiments. We modify the two-stream convents architecture by replacing each stream with resnet-101. It outperforms state-of-the-art I3D architecture. EgoK-360 is the first to address egocentric activity recognition in 360 environment. We believe EgoK360 dataset will be beneficial to the EAR360 research.

III. REVISITING OPTICAL FLOW ESTIMATION IN 360° VIDEOS

Introduction

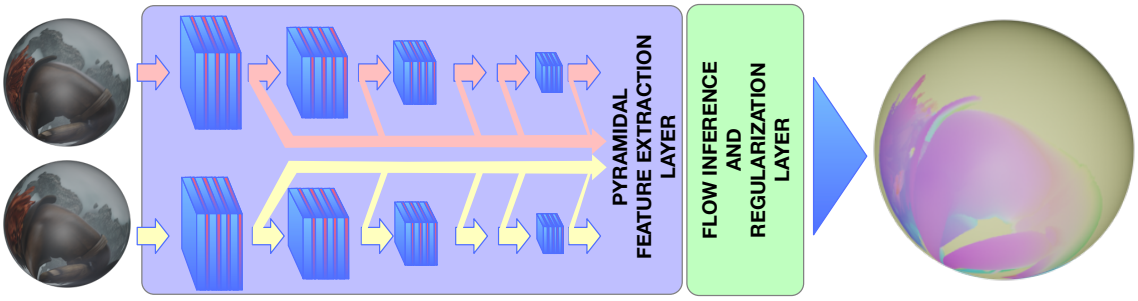


Figure 3.1: **Overview of LiteFlowNet360 network architecture.** We put focus only on feature extraction block which is shown in detail. Flow inference and regularization layer is similar to the original implementation. Input to the network are equirectangular or spherical data. Each convolution layer in pyramidal network is transformed to adapt spherical convolution(shown in red color). Final output is optical flow in spherical domain.

The immersive 360 video technology shows promising growth in the past years. Services such as GoPro, VeeR, Visbit, Facebook360 and YouTube have become great platforms for 360 videos. 360 videos are shaping the future of content creation and sharing. Hence, 360 videos will be an important digital medium in near future. This adds newer challenges and opportunities in computer vision research. One of such challenge is the motion and optical flow estimation in 360 videos.

Motion and optical flow estimation is important for 360 video understanding. Motion information can significantly aid tasks such as saliency detection, saliency prediction, gaze prediction, video piloting in 360 videos [38, 39, 40, 41]. Similarly, optical flow based panorama video stitching has shown impressive results compared with other methods. Deep Learning based optical flow estimation methods have shown significant improvement over classical methods [42, 43]. Evolution of optical flow estimation methods from simple CNN based architecture to complex feature pyramid based architecture shows significant improvements as well [44, 32, 45].

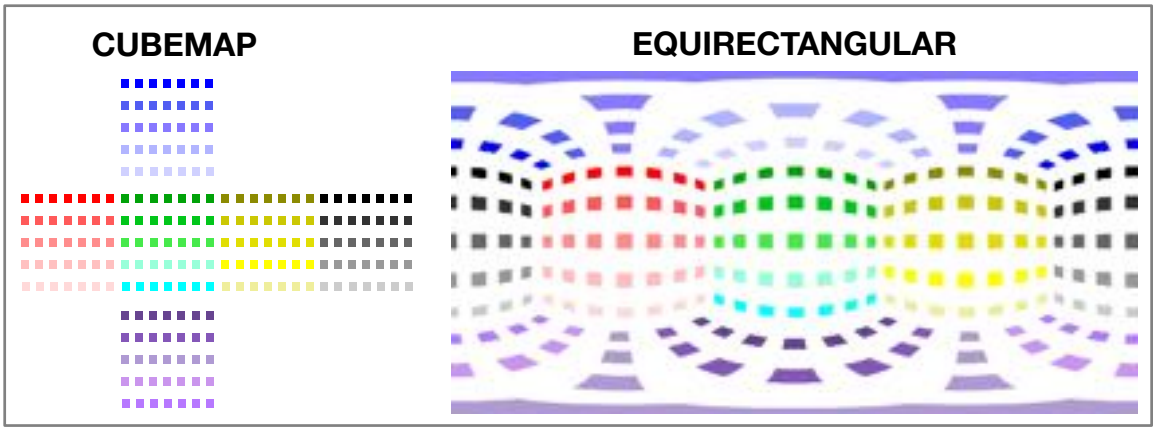


Figure 3.2: **Radial distortion.** Showing how regular kernel map does not work in equirectangular (right) projection. When kernel applied in cubemap (left) are mapped into equirectangular projection it suffers huge distortions.

However, regular CNN architectures are not suitable for 360 videos because of inherent distortion caused by projection of spherical videos to plane. We can use techniques such as [37, 46, 47, 48] to achieve spherical convolution. However, these methods have enormous overheads while converting existing complex pyramid based architecture to fit the needs of distortion free convolution for 360 videos. First, the training of the optical flow network is unstable. Having many transform convolution layers will lead entire process complicated as architectures becomes bigger. Second, we may not be able to guarantee that our model works even if we transform the architecture. We need some metrics such as EPE(End Point Error) to decide if our model works well. Since we lack labelled 360 video dataset for optical flow, the only method that fits our requirement is self-supervised methods. But how do we train this architecture in a self-supervised manner? The core part of self-supervised approach is calculating the loss between warped image and target image using predicted flow. Are warping techniques generalizable? In later sections we aim to answer these questions.

Choosing a right architecture for our framework was the initial challenge we faced. However, we set certain requirements (like size, speed and efficiency) as a

guide to choose the right architecture. There are many optical flow architectures to choose from, LiteFlowNet wins the competition. We will discuss more about this architecture in the following section. Framework we proposed would grow significantly as it includes significant changes as a part of perspective to a spherical domain transformation process. One more significant addition to this transformation process is the inclusion of special convolution to adapt the spherical nature for our dataset. This is important because the dataset we work on is an equirectangular plane, a sphere-to-plane projection. This planar projection incurs heavy distortion, which we have illustrated in Fig.3.2. These special convolution, termed as spherical convolution, are expensive in terms of computation, which voids our requirements. Therefore, we adopt techniques like kernel transformation using transformer network. One such architecture [36] dubbed as KTN has shown comparable improvement over later methods with less computation. We adopt KTN as our transformer network to learn spherical convolution.

Training an end-to-end optical flow architecture requires significant considerations of extra jobs like scheduling of training, implementing stacked architecture, considerations of motion magnitude and many other details to make it work on a par with the state-of-the-art results. Training architectures like this using better strategy to cope with gradually increasing task is an adoption of a popular philosophy, curriculum learning[49]. We have seen architecture like [32, 45] adopted this strategy to create a better model. Apart from traditional optical flow estimation strategies, we have additional requirements because of the spherical nature of the dataset. Therefore, end-to-end training of optical flow for 360 video is highly unstable as there are many parallel objectives to achieve. We need to make sure that our model size does not grow significantly. This can slow down the training and inference speed. We need to address the nature of optical flow in 360 domain which can change the interpretation of warping techniques, flow

representation and many other aspects. To make it brief, training optical flow architecture in 360 videos is not straightforward. To achieve the stable training process and fulfill our requirements, we adopt a divide-and-conquer strategy, thus dividing the entire training process into three major stages.

First stage of LiteFlowNet360 is to train LiteFlowNet architecture in perspective video datasets like [2]. Then, we transform source CNN to target CNN layer wise. This transformation technique is progressive which means we need to do everything in-order. We will explain details about process of transformation in method section. After transforming into target CNN we will now further train entire model in an end-to-end fashion in supervised manner. To achieve this task we augment both source perspective videos and target optical flows into spherical distortion setting. When the second stage is complete, we will use a self-supervised scheme to further train our model in target videos. To do this, we need to perform a task like back-warping of frames using predicted flow. We use these predicted or warped frames as the basis of the training process by minimizing the similarity loss between ground frames and predicted frames. We adopt occlusion aware scheme inspired by [50].

In this chapter we exploit the existing optical flow estimation techniques and distortion free convolution in 360 videos. Our contributions are three folds: **(i)** To the best of our knowledge, this is the first work to address deep learning based dense optical flow estimation in 360 videos. **(ii)** We present an algorithm inspired by [36] to transform learned representations from pre-trained network. **(iii)** We present a self-supervised learning approach since we do not have ground truth optical flow for 360 videos.

Related Work

Optical Flow Estimation. The classical optical flow estimation approaches [42, 43] used variational approaches to minimize energy based on brightness constancy and spatial smoothness. Recently, [44] proposed an end-to-end optical flow estimation with convolutional networks (FlowNet) using supervised scheme. Several other works based on CNN followed FlownNet including 3D convolution based approach [51], unsupervised approach [52, 53, 54] and pyramidal-coarse-to-fine approach [55, 56]. Recent variants such as [57, 58] used sparse matching by learning feature embedding. These methods were computationally expensive, making it impossible to train end-to-end fashion. FlowNet-2.0 [32] was an important addition in this series. It exploited curriculum learning approach [49]. In their work they address the weakness of FlowNet by addressing a smaller to larger range of displacement magnitude. However, the success of FlowNet-2.0 comes with a cost of over parametrization with around 160 Million parameters. [45] presented a more effective approach dubbed as LiteFlowNet. LiteFlowNet is around 30 times smaller and around 1.36 times faster. LiteFlowNet excelled FlowNet-2.0 by drilling down architecture details. LiteFlowNet proposed effective flow inference at each pyramid level, presented data fidelity and regularization as variational methods, whereas FlowNet only used a U-Net like architecture. Self-supervised [50, 59] approaches for optical flow estimation are intuitive and reasonable approaches, as warping is one of the fundamental techniques used in successful deep learning based architecture. These techniques motivate our self-supervised learning scheme in the final stage.

CNN for 360 Video/Images. Performing direct convolution on spherical data led to inaccurate models [60, 61]. An intuitive approach to perform convolution on spherical data is to use convolution directly in cube map projection [62, 63]. This introduced less distortion but the model will have discontinuities, which led to

sub-optimal model for several tasks. Another approach to learn rotation invariant CNN was to use graph convolution [64] techniques. This can be done by defining convolution in spectral domain [46, 47]. Similarly, this can also be done by projecting both feature maps and kernel in a spectral domain and apply regular CNN. These methods lose semantic information and were not useful in our case. In a recent year, several other spherical CNN based models have been proposed. Work such as [37, 48] considered distortion in sphere-to-plane projection of spherical images/videos. Recent method by [36], dubbed as Kernel Transformer Network(KTN) is a significant piece of work in this domain. This architecture efficiently transferred convolution kernels from perspective images to the equirectangular projection. Basically, KTN produced a function parametrised by a polar angle and kernel as output. This work preserved the source CNNs and maintained accuracy, meanwhile offering transferability and scalability. Our architecture is a modified version of KTN, which uses interleaving convolution techniques to reduce discontinuity during convolution.

360 Flow Estimation. [65] proposed an approach by back-projecting image points to a virtual curved retina intrinsic to the geometry of central panoramic camera. Their method could adopt to contemporary ego-motion algorithms. [66] implemented Lucas-Kanade based method for optical flow estimation in catadioptric images. They proposed new constraint based on motion model defined on perspective images. This new constraint-based model was used to compute optical flow for omnidirectional image sequences. [67] used multichannel spherical image decomposition techniques to compute optical flow for 360 image sequences. Similarly [68] proposed several variational regularization methods to estimate and decompose motion fields on the sphere. [69] implemented, adapted phase based method to compute optical flow using different treatments to account for 360 images.

Our work fall somewhere in the intersection of optical flow estimation and

emerging domain of omnidirectional computer vision. However, none of the above methods address the optical flow estimation problem using deep learning methods.

Method

We choose LiteFlowNet [45] as the basis for our newly proposed LiteFlowNet360 (shown in Fig. 3.1) architecture because of its simpler design, lightweight (5.37M parameters) and highly efficient implementation. We represent our LiteFlowNet360 architecture in terms of two important blocks, feature extractor block (F) and regularization block (P) as shown in Eq. III.1 where X is a sequence of two consecutive frames, k is the number of layer transform from 0^{th} to k^{th} layer in F and where n is an optimum number of layers eligible for transformation.

$$F_k(X) = \begin{cases} F_0(X) & : k = 0 \\ F_k(F_{k-1})(X) & : n > k > 0 \end{cases} \tag{III.1}$$

Feature extractor block F is transformed to adapt our need of 360 flow estimation as shown in Eq. III.2. Each convolution layers $F_0, F_1, ..., F_{(n-1)}$ in feature extractor block is parametrised by a function $\Omega = g(\theta, \phi)$ in sphere, by generating different kernels for distortion above and below the equatorial region in sphere such that layers $F'_0, F'_1, ..., F'_{(n-1)}$ are our target layers. We compute location dependent kernel using polar and azimuthal angle θ and ϕ respectively.

We keep inference and regularization block as the same as LiteFlowNet. However, feature warping techniques at each pyramid level is transformed to address warping in 360 video domain. Further details will be explained in the following section.

Algorithm 1 : Interleaving Convolution

$Y \leftarrow tensor()$;
$X \leftarrow input$;
$tied\_weights \leftarrow n_g$;
$n\_transform \leftarrow h/n_g$;
for *each* $row \in [0, n\_transform]$ **do**
  $start \leftarrow row \times tied\_weights$;
  if $row < (n\_transform - 1)$ **then**
    $end \leftarrow start + tied\_weights + n_l$;
  else
    $end \leftarrow start + tied\_weights$;
  end
  $Y[start : end] \leftarrow \sum_{i,j} K_{row}[i, j] * X_{row}[x - i, y - j]$;
end

$$F'_k(X) = \begin{cases} F'_0(F_0(X), \Omega) & : k = 0 \\ F'_k(F_k, \Omega)(F'_{k-1}(F_{k-1}, \Omega))(X) & : n > k > 0 \end{cases} \tag{III.2}$$

LiteFlowNet360 framework is an evolutionary architecture. We start from regular LiteFlowNet architecture and perform incremental transformation and training process to achieve final architecture. We formulate three important subsequent stages, transformation stage, intermediate refinement stage and final refinement.

Stage 1: Transformation

This stage starts with training the LiteFlowNet architecture with labeled data following [45]. The most important part of this stage is to transform convolution layers trained on perspective images to adapt to 360 images. We follow [36] with some improvements, which we will present later in this section. Since we use equirectangular projection method, distortion depends only on polar angle. This leads to direct correspondence of the polar angle to the height of the input image, i.e., $y = \theta h/\pi$. This means we can utilize a single kernel for optimal row-group

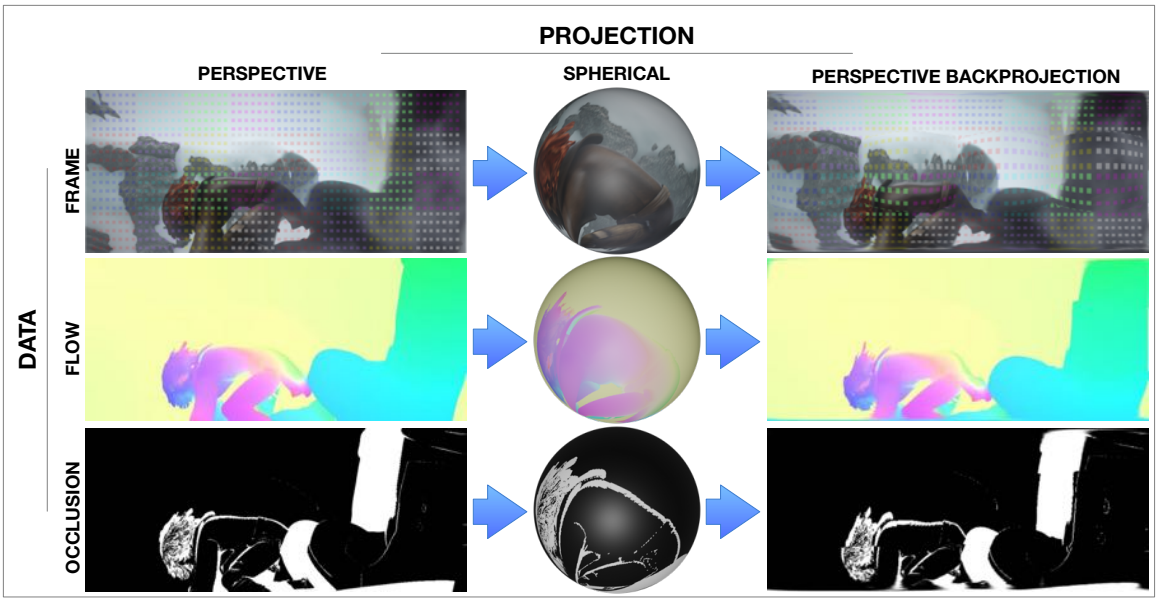


Figure 3.3: **Spherical data augmentation.** Perspective videos are projected in a unit sphere and then back projected to equirectangular plane. This is intentionally lossy process to create distortion artifact(shown in top row) in perspective data.

size(n_g) such that we have h/n_g projection matrices $P_i \in \mathbb{R}^{r_i \times k_h \times k_w}$, where $r_i = h_i \times h_w$ is target kernel for each row-group i and $(k_h \times k_w)$ is an original kernel size from source CNN as in [36]. Different from original implementation, we interleave these rows as shown in Algorithm-1 to maintain connectedness, where n_l is interleaving factor. We choose $n_l = 3$ in our case.

We train each layer of feature extractor evolutionarily. We feed augmented images created by warping perspective image sequence as inputs to transformed layer and original image sequence as inputs to source CNN. Warping process is done by plane-to-sphere and sphere-to-plane projection of perspective image sequences. This back projection trechnique introduce distortion in perspective videos, as shown in Fig.3.3 (top row).

$$\begin{aligned} Y_k &= F_k(X), Y'_k = F'_k(F_k(X), \Omega) \\ L_k &= ||Y'_k - Y_k||^2 \end{aligned} \tag{III.3}$$

We train each layer with objective function presented in Eq.III.3 to minimize the L2

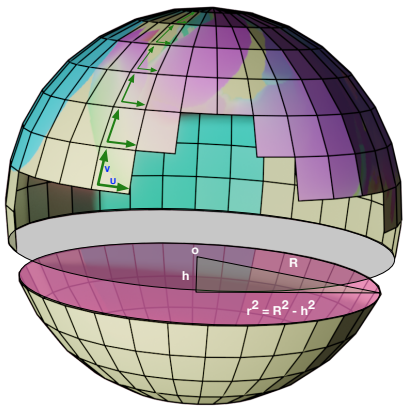


Figure 3.4: **Flow representation in spherical domain.** (u, v) component changes as we move away from equator.

norm between feature map generated by source CNN and transformed CNN layers, where Y_k and Y'_k represent output at k_{th} layers in source and target architecture respectively and L_k is an L2 norm between feature map from source and target CNN.

$$L'_k = \frac{1}{n_g} \sum_{i}^{n_g} L_k(\Omega(Y'^i_k), Y^i_k) \tag{III.4}$$

We project feature map row-group wise in tangential plane and compute loss with respect to corresponding feature map row-group from perspective source CNN. We combine these losses by averaging all the losses in row-group as shown in Eq.III.4, where i refers to i-th row-group, and L'_k is an L2 norm averaged over row-group.

Stage 2: Intermediate Refinement

Though first stage can transform convolution layers to adapt spherical images, it does not guarantee that estimated flow are well represented. A common problem will arise when we try to warp inferred flow around the sphere. This is due to the nature of sphercial coordinates. We can observe in Fig. 3.4 that the size and the

shape of the patches decreases as we move away from the equatorial region. This means u and v component(shown in figure) changes as we move away from the equatorial region. This is because of the difference between idea behind the optical flow representation in perspective domain vs spherical domain. Optical flow in perspective domain is represented by the displacement in terms of euclidean distance. However, in spherical domain flow information makes sense only if we represent flow in terms of angular displacement. This means we need to present u and v in terms of u_θ and v_ϕ component. Instead of obtaining a direct solution, which is beyond the scope of our work, we introduce some correction factor on original u and v and project it in spherical domain.

The second stage is to refine the representation learning of optical flow in spherical domain. The intermediate refinement process is all about end-to-end training of the transform network. The training process is supervised as we want to make sure our network learn the actual representation. The problem with this scheme is that we do not have labelled dataset. Core part of the intermediate refinement stage is to use data augmentation techniques to convert labelled data, both images and optical flow in a spherical domain. We show sample augmented image sequences and corresponding optical flow in Fig.3.3.

Equirectangular plane is expressed from $(-\pi, \pi)$, $(-\pi/2, \pi/2)$ for length and height respectively, leading the aspect ratio of $length : height = 2 : 1$. We resize our original image and optical flow with the nearest interpolation scheme to maintain required aspect ratio. Then, we use simple projection techniques given by $(\phi = 2 \times \pi \times u, \cos\theta = 2 \times v - 1)$ for unit sphere to perform forward projection (i.e., perspective to spherical projection) followed by restoration using backward projection (i.e., spherical projection to backward projection).

As we discussed, issues regarding projecting perspective optical flow directly into sphere requires a correction factor. We apply this correction factor separately

for u and v component of original perspective flow. The idea behind these factors is to scale displacement magnitude to be fair all over the points in spherical representation. For example, u is corrected by scaling each row with the ratio of central circumference (corresponding to the actual width w in perspective plane) and circumference $w_i = 2\pi r_i$ at each row. Regarding calculation of r_i, see Fig.3.4, where radius of a pixel-row i at distance $h_i = |R - i|$ from center can be calculated using simple law of triangle $r_i^2 = R^2 - h_i^2$ where $R = \frac{w}{2\pi}$, where $i \in (-R, R)$. We finally define function $\Omega_{(x,y)}$ to perform perspective to spherical projection, $\omega_{(r,\theta,\phi)}$ to perform back projection and ζ as correction function. Now we present image augmentation I as $I' = \omega(\Omega(I))$ and optical flow Δ as $\Delta' = \omega(\Omega(\zeta(\Delta))$. We present spherical data augmentation algorithm in Algorithm-2.

Algorithm 2 : Spherical Data Augmentation

$\Delta_{I_1 \to I_2} \leftarrow input()$;
$(I_1, I_2) \leftarrow I \leftarrow input()$;
$(h, w) \leftarrow dim(I_1)$;
$(r_w, r_h) \leftarrow (\frac{h}{4\pi}, \frac{w}{2\pi})$;
for *each* $i \in [-h/2, h/2]$ **do**
  $\Delta_u = \Delta_{I_1 \to I_2}[i, :] \times \frac{2\pi\sqrt{r_w^2 - |r_w - i|^2}}{w}$;
end
for *each* $j \in [-w/2, w/2]$ **do**
  $\Delta_v = \Delta_{I_1 \to I_2}[:, j] \times \frac{2\pi\sqrt{r_h^2 - |r_h - j|^2}}{h}$;
end
$\Delta'_{I_1 \to I_2} \leftarrow \omega(\Omega(\Delta))$;
$I' \leftarrow (I'_1, I'_2) \leftarrow \omega(\Omega(I))$;

The training objective is to minimize end point error $||\Delta' - \Delta''||$ between predicted flow Δ'' and ground truth flow Δ' in conjunction with brightness error $||(I'_1 + \Delta'_{I_1 \to I_2}) - I''_2||$ between the warped image and source image. We follow routine prescribed by [45] to train our network, but we limit our training process to significantly fewer amount of epochs compared to original implementation, as this is only a refinement process and network plateaus in terms of error rate. Our model is

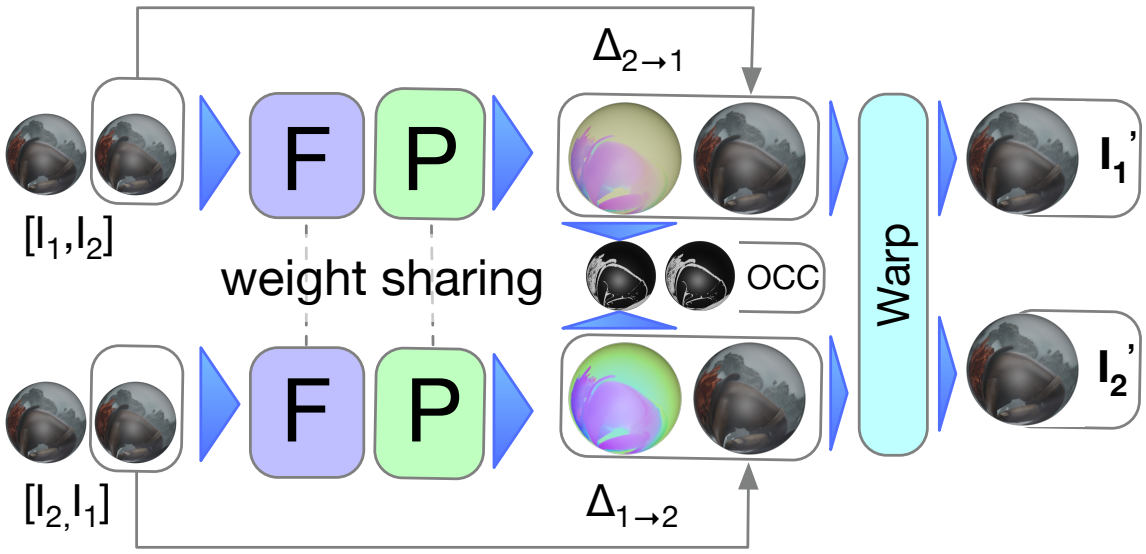


Figure 3.5: **Final refinement process.** Network from second stage is extended to have two parallel weight sharing architecture.

now ready to cope with the spherical domain, but we need to adapt our model to real-world data. To adapt our model to real-world data, we move into ultimate refinement stage.

Stage 3: Final Refinement

We replicate our initial network from stage-2 into two channel siamese network as shown in Fig. 3.5 to estimate forward $\Delta_{1\to2}$ and backward $\Delta_{2\to1}$ flow. We use forward and backward flow to estimate occlusion $\tilde{O} = (\tilde{O}_{2\to1}, \tilde{O}_{1\to2})$ using Eq.III.5 where $\epsilon \approx 10^{-2}$, $(i, j) = (1, 2)$ for forward flow and vice versa.

$$
\begin{aligned}
M_i &= \begin{cases} 0 \\ \\ 1 \quad if, |\Delta_{i\to j}| \leq \epsilon \end{cases} \\
\tilde{O}_{i\to j} &= M_i \odot ((1 - M_j) + \tilde{O}_{j\to i})
\end{aligned}
\tag{III.5}
$$

$$L_p = \sum_{i,j} \frac{\sum \psi(I_i - I_i') \odot (1 - O_{i\to j})}{\sum 1 - O_{i\to j}}$$

Similarly, we use predicted optical flow to warp target image. Apart from traditional warping techniques, we modify warping technique as shown in Algorithm-3. This warping technique is necessary to address the continuous nature of 360 images, i.e., whenever pixel displacement occurs beyond the boundary condition the pixel is displaced somewhere within the equirectangular plane. For example, if a pixel is displaced beyond the right boundary, the pixel will be displaced on the left side of the equirectangular plane. This is not true with perspective flow, where we consider this as a boundary condition and put the pixel into boundary. This is well preserved and more accurate assumption for smaller displacement in the border area.

Algorithm 3 : Boundary Condition

$(g_\theta, g_\phi) \leftarrow ([-180, +180], [-90, +90])$;
$G \leftarrow mesh\_grid(g_\theta, g_\phi)$;
$(\tilde{\Delta}_u, \tilde{\Delta}_v) \leftarrow \tilde{\Delta} \leftarrow G + \Delta_{1\rightarrow 2}$;
$\tilde{\delta_u} = \frac{\tilde{\Delta_u}}{|\tilde{\Delta_u}|}(|\tilde{\Delta_u}| - 360)$;
$\tilde{\delta_v} = \frac{\tilde{\Delta_v}}{|\tilde{\Delta_v}|}(180 - |\tilde{\Delta_v}|)$;

$$\tilde{\Delta}_u = \begin{cases} `\tilde{\Delta}_u, & \tilde{\Delta_u} \in [-180, 180] \\ -\tilde{\Delta}_u, & \tilde{\Delta_v} \notin [-90, 90] \\ \tilde{\delta_u}, & \tilde{\Delta_u} \notin [-180, 180] \end{cases};$$

$$\tilde{\Delta}_v = \begin{cases} \tilde{\Delta}_v, & \tilde{\Delta_v} \in [-90, 90] \\ \tilde{\delta}_v, & \tilde{\Delta_v} \notin [-90, 90] \end{cases};$$

$\tilde{\Delta} \leftarrow (\tilde{\Delta}_u, \tilde{\Delta}_v)$;

We present final refinement process as further training steps to adapt to the target domain. We use dataset from our ongoing work Egok360, an egocentric activity recognition dataset for 360 videos as target dataset. The training process is self-supervised based on photometric loss as shown in Eq.III.5 where $\psi = (|x| + \epsilon)^q, \epsilon \approx 10^{-2}, q \approx 1 \times 10^{-1}$.

Table 3.1: **Experimental results on Sintel360 dataset.**

Model	Data	#Layers	EPE	L_p^*
LiteFlowNet[45]	Sintel360	0	~ 6.35	~ 1.30
LiteFlowNet +[36]	Sintel360	> 4	≥ 17	≥ 3.06
Ours, Stage-2	Sintel360	4	~ 6.35	~ 0.70
Ours, Final	Sintel360	4	~ 3.95	~ 0.60

Results

We present our result mainly on augmented Sintel [2] dataset, which we termed as Sintel360. We performed spherical data augmentation on original Sintel training set, which we divided into 9:1 train-val set. This train-val set has ground truth optical flow information. We compared 4 different models as shown in Table 3.1 using commonly used end point error (EPE) metrics using validation set. To make comparision fair, we augmented original sintel test set. We used this test set to compute photometric loss (L_p), defined in Eq.III.5.

Quantitative Results. Table 3.1 summarizes our experiments. We found that exhaustive layer replacement task is unnecessary. The convergance rate dramatically decreases as we go deeper, as shown in Table 3.1, EPE is significanly higher (≥ 17) for more than 4 layers replacement. This creates a domino effect, which propagate errors in subsequent layers. We illustrate this effect in Fig.3.6. We can see that beyond layer 4, the output of transformed layers are different. Instead of reproducing the source CNN these layers learn nothing even after training for 30 epochs, using same techniques that was used to train previous layers.

Our model on Stage 2 performs on par with original implementation. Though original method seems fine, representation for optical flow in spherical video is not fair. We can explain the lower EPE on the original model with the large number of flow information correspondence between real and augmented data in central region

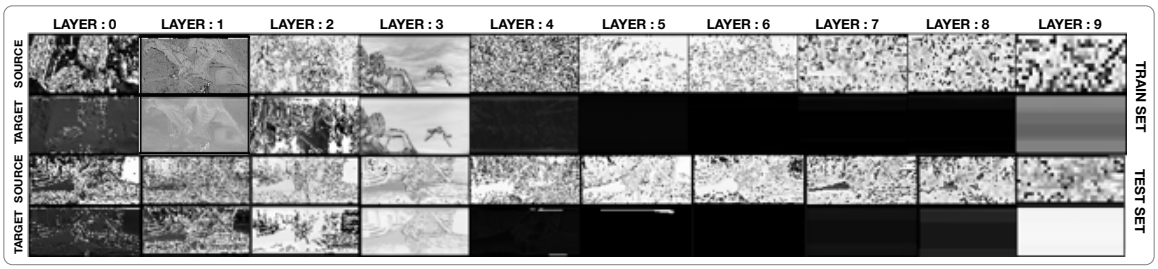


Figure 3.6: **Comparing activations in source and target CNN.** Showing randomly picked individual channel from output of different layers in source and target(stage 2) architectures. Shows source and transform CNNs on train (first row) and test (second row) set.

of equirectangular plane. With a final refinement stage, we improve our model significantly bringing EPE to 3.95 from 6.35 on val-set and photometric loss from 0.70 to 0.60 on a test set.

Qualitative Result. Fig.3.7 shows qualitative results from our experiment compared with baseline LiteFlowNet. We also present qualitative results on our target 360 video dataset. To understand the fairness of the flow predicted, we used flow information to predict the next frame. We observe that the warping of flow preserves the spherical nature. In another word, it preserves the artificat we introduced in original dataset (please note patches in different colors in target dataset shown in Fig.3.7). However, there are cases where none of these models work as expected. We show such case in the last row of estimated optical flow on target dataset. We believe this can be improved further by allowing the model to have longer training times with further hyperparameter exploration.

Conclusion

In this chapter, we presented a novel framework for 360 optical flow estimation, dubbed as LiteFlowNet360. This framework is an adaptation of existing best practices from both of the world, "optical flow estimation for perspective videos" and "spherical convolution for 360 videos/images". We presented our work as three

major subsequent stages, transformation stage, intermediate refinement stage and final refinement stage. We started with the process of transformation, which includes evolutionary learning of spherical convolution based on transformer network. Apart from the success of these methods in other field, we empirically showed that exhaustive layer transformation from source to target CNN is insignificant in the context of optical flow estimation. We present second stage to address the correct representation of 360 flow. This stage requires further training as a refinement task. To train our model, we introduce a lossy data augmentation techniques to exploit existing labelled datasets. This technique allowed us to introduce artifacts related to spherical distortion in perspective videos, meanwhile transforming optical flow information in a spherical domain. We presented final stage as a domain transfer stage, where we use unlabelled target 360 video data to train our model in a self-supervised manner. Empirical and qualitative results showed the potential of this work. We believe this work will inspire others to investigate this area of optical flow estimation.

Acknowledgements: This research was partially supported by NSF CSR-1908658 and NeTS-1909185. This article solely reflects the opinions and conclusions of its authors and not the funding agents.

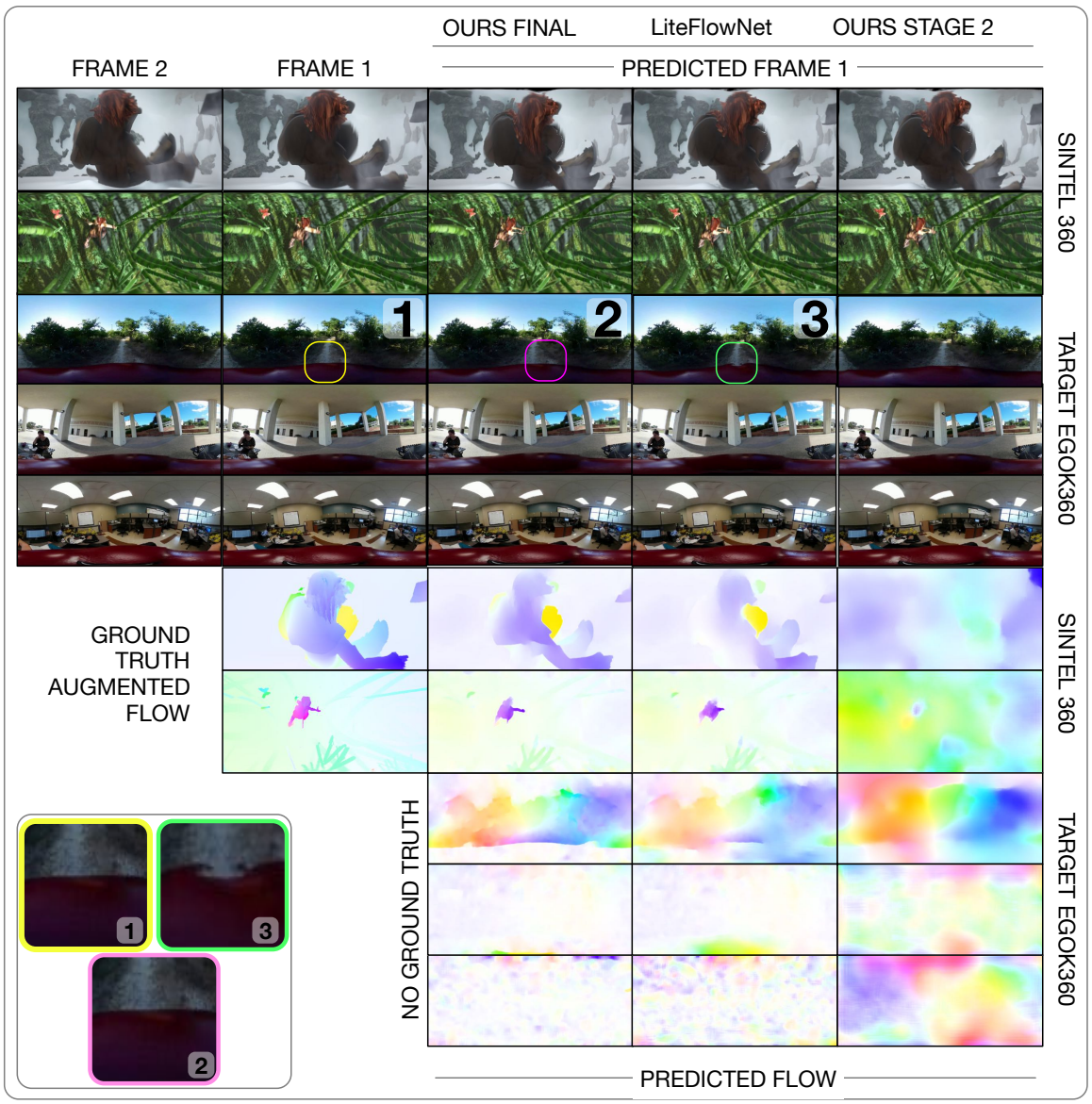


Figure 3.7: **Qualitative results from LiteFlowNet360.** Qualitative results on augmented Sintel 360 dataset and target video dataset. First two row represents randomly picked frames and second two row represents corresponding optical flow information. We predict frame-1 using forward flow from each architecture. We randomly pick patches from same location from predicted(patch-2,patch-3) and ground truth(patch-1) frame-1 as shown in bottom left corner. Patch 2,3 are from liteflownet360 and liteflownet respectively. We can see liteflownet360 results are comparatively better. Note: We encourage digital reader to zoom in for detail view.

IV. LEARNING OMNIDIRECTIONAL FLOW IN 360° VIDEOS VIA SIAMESE REPRESENTATION

Introduction

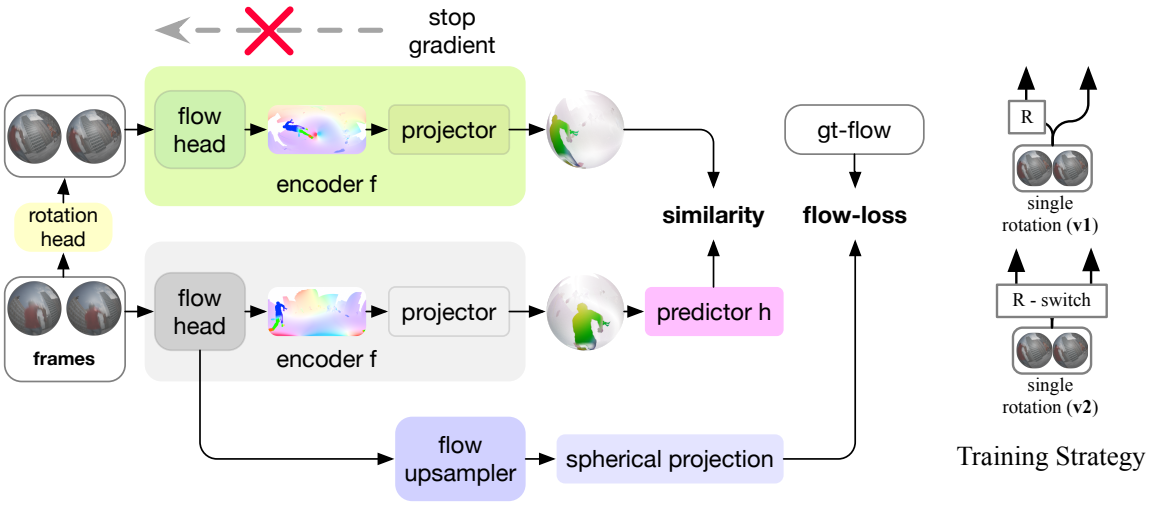


Figure 4.1: **Siamese representation learning for omnidirectional flow-(SLOF).** Pairs of frame sequence (w/ and w/o random rotation) are passed as inputs to encoder f (RAFT as a flow head backbone and a standard convolutional projector layer). A predictor layer h is an MLP layer. The entire framework is trained by fusing the pretraining and fine-tuning stage to combine the similarity and flow-loss in a single stage. The model maximizes the similarity between latent representations of flow information from two streams and minimizes the flow loss. **Training Strategy (right):** Here two different arrows$(left, right)$ represent siamese streams or input pathways to our model. **v1** and **v2** (randomly switching the rotational augmentation between two streams) are similar strategies achieving overall better performance.

Optical flow estimation, as a fundamental problem in computer vision, has been studied over decades by early works [43, 42] dated back to 80s. Before the era of modern deep learning, traditional optical flow estimation methods relied on hand-crafted features based optimizations [70, 71, 72], energy-based optimizations [73, 74, 75] and variational approaches [76, 77, 78]. Although deep learning-based approaches [79, 80, 81, 82, 32, 45] have shown great advantages over these classical approaches, most of them are specially tailored for perspective videos.

The availability of perspective optical flow datasets [2, 83, 84, 85, 86] heavily supports the advancement of these modern deep learning-based approaches. The optical flow datasets are difficult to obtain and requires the generation of naturalistic synthetic dataset like Sintel [2]. As these datasets mark the foundation for optical flow estimation research, the availability of reliable omnidirectional datasets is equally important to advance the omnidirectional flow estimation research. The need for the datasets brings up the first challenge: there is no such reliable (perceptually natural and complex) 360° or omnidirectional video dataset in the literature collected for omnidirectional optical flow estimation. Another challenge of omnidirectional optical flow estimation is that current perspective video-based deep networks fail to accommodate the nature of 360° videos. These perspective optical flow estimation methods inevitably require fine-tuning due to the presence of radial distortion [87] on 360° videos. This fine-tuning task is effort-intensive and requires several transformation techniques to adapt the distortion [36, 88]. An intuitive solution is to fine-tune perspective-based deep networks under omnidirectional supervised data. However, this brute-force migration of perspective-based networks often requires enormous supervision and still leads to significant performance degradation [89].

We address the first challenge of reliable benchmark dataset shortage by proposing a new dataset named FLOW360. To the best of our knowledge, this is the first perceptually natural-synthetic 360° video dataset collected for omnidirectional flow estimation. Currently, existing omnidirectional datasets face two significant issues i.e., lack of full 360° FOV (field of view) and lack of perceptual realism. Specifically, OmniFlow[90] dataset only has 180° FOV failing to address the omnidirectional nature, while the dataset proposed in OmniFlowNet[91] lacks perceptual realism in scene and motion. Meanwhile, perspective optical flow datasets such as [86, 2, 83] have facilitated researchers in investigating perspective

optical flow estimation methods [45, 44, 32, 81, 80], where the availability of such omnidirectional videos dataset is essential to advance this particular field. It is worth noting that FLOW360 dataset can be used in various other areas such as continuous flow estimation in 3-frame settings with forward and backward consistency [50, 92, 93], depth [94, 95] and normal map estimation [96].

The accommodation to the omnidirectional nature generally requires modification of convolution layers and further refinements on the target dataset due to the presence of radial distortions [89], which is caused by projecting 360° videos (spherical) to an equirectangular plane. Existing works design various convolution layers to address the distortion problem, such as spherical convolution [36, 62, 48, 37], spectral convolution [46, 47] and tangent convolution [88]. Although these methods can achieve better performance than classical CNN convolutions, they require immense effort with layer-wise architecture design, which is impractical for high-demanding deployment in the real-world setting.

Instead of adding new convolution layers, we design a novel SLOF (**S**iamese representation **L**earning for **O**mnidirectional **F**low) framework (Fig. 4.1), which leverages the rotation-invariant property of omnidirectional videos to address the radial distortion problem. The term rotation-invariant here implies that 360° videos are rotated in a random projection such that the reverse rotation of such projection is equal to the original projection. This rotation-invariant property ensures that omnidirectional videos can be projected to a planar representation with infinite projections by rotating the spherical videos on three different axis (X, Y, Z), namely "pitch", "roll" and "yaw" operations preserving overall information. Specifically, we design a siamese representation learning framework for learning omnidirectional flow from a pair of consecutive frames and their rotated counterparts, assuming that the representations of these two cases are similar enough to generate nearly identical optical flow in the spherical domain. Besides, we design and compare different

combinations of rotational augmentation and derive guidelines for selecting the most effective augmentation scheme.

To summarize, we make three major contributions in this chapter: (**i**) we introduce FLOW360, a new optical flow dataset for omnidirectional videos, to fill the dataset's need to advance the omnidirectional flow estimation field. (**ii**) We propose SLOF, a novel framework for optical flow estimation in omnidirectional videos, to mitigate the cumbersome framework adjustments for omnidirectional flow estimation. (**iii**) We demonstrate a new distortion-aware error measure for performance analysis that incorporates the relative error measure based on distortion. Finally, we compare our method with existing omnidirectional flow estimation techniques via kernel transformation [36] to address radial distortions. The FLOW360 dataset, the SLOF framework, and our experimental results provide a solid foundation for future exploration in this important field.

Related Work

Optical Flow Datasets. Perspective datasets such as [97, 86, 98, 99, 100, 101] comprise synthetic image sequences along with synthetic and hand-crafted optical flow. However, these datasets fall short in terms of perceptual realism and complexities. Even though several optical flow datasets have been published recently in [102, 83, 84, 85], they are primarily used in automotive driving scenarios. The other relevant dataset in the literature was Sintel [2], which provided a bridge to contemporary optical flow estimation and synthetic datasets that can be used in real-world situations.

All datasets, as mentioned earlier, are introduced for perspective videos thus cannot be used for omnidirectional flow estimation. So to address this problem, LiteFlowNet360 [89] on omnidirectional flow estimation was released to augment the Sintel dataset by introducing distortion artifacts for the domain adaptation task.

Nevertheless, these augmented datasets are discontinuous around the edges and violate the 360° nature of omnidirectional videos. The closest datasets to ours are OmniFlow [90] and OmniFlowNet [91]. OmniFlow introduced a synthetic 180° FOV dataset, which is limited to indoor scenes and lacks full 360° FOV. Similarly, OmniFlowNet introduced a full 360° FOV dataset. However, both datasets lack complexities and evidence for perceptual realism. We show a detailed comparison of FLOW360, OmniFlow, and OmniFlowNet in Fig. 4.5. Compared to existing datasets in the literature, FLOW360 is the first perceptually natural benchmark 360° dataset and fills the void in current research.

Optical Flow Estimation. Advancements in optical flow estimation techniques largely rely on the success of data-driven deep learning frameworks. Flownet [44] marked one of the initial adoption of CNN- based deep learning frameworks for optical flow estimation. Several other works [32, 45, 53, 51, 52, 54, 55, 56] followed the footsteps with improved results. Generally, these networks adopt an encoder-decoder framework to learn optical flow in a coarse-to-fine manner. The current framework RAFT [80] has shown improvements with correlation learning.

The methods mentioned above are insufficient on omnidirectional flow field estimation as they are designed and trained for perspective datasets. One of the initial work [65] on omnidirectional flow estimation was presented as flow estimation by back-projecting image points to the virtually curved retina, thus called back-projection flow. It showed an improvement over classical algorithms. Similarly, another classical approach [103] relyed on spherical wavelet to compute optical flow on omnidirectional videos. However, these methods are limited to classical approaches as they are not relevant in existing deep learning-based approaches. One of the recent works, LiteFlowNet360 [89] tried to compute optical flow on omnidirectional videos using domain adaptation. This method utilized the kernel

transformer technique (KTN [36]) to adapt convolution layers on LiteFlowNet [45] and learn correct convolution mapping on spherical data. Similarly, OmniFlowNet [91] proposed a deep learning-based optical flow estimation technique for omnidirectional videos. The major drawback of these methods is the requirement to adapt convolution layers, which takes a substantial amount of time and makes portability a significant issue. For example, in LiteFlowNet360, each convolution layer in LiteFlowNet was transformed using KTN with additional training and adjustments. Similar to OmniFlowNet, every convolution layer in LiteFlowNet2 [104] was transformed using kernel mapping [105] based on different locations of the spherical image. These techniques incur computational overheads and limit the use of existing architectures. Such approaches demand explicit adaptation of convolution layers, which is hard to maintain when more up-to-date methods are published constantly. Contrary to these methods, we propose a Siamese Representation Learning for Omnidirectional Flow (SLOF) method to learn omnidirectional flow by exploiting existing architectures with designed representation learning objectives, significantly reducing the unnecessary effort of transforming or redesigning the convolution layer.

Siamese Representation Learning. Representation learning is a powerful approach in unsupervised learning. Siamese networks have shown great success in different vision-related tasks such as verification [106, 107] and tracking[108]. A recent approach [109] in siamese representation learning showed impressive results in unsupervised visual representation learning via exploiting different augmentation views of the same data. They presented their work in pre-training and fine-tuning stages, where the former being the unsupervised representation learning. We use the representation learning scheme on omnidirectional data via rotational augmentations, maximizing the similarity for latent representations and minimizing the flow loss.

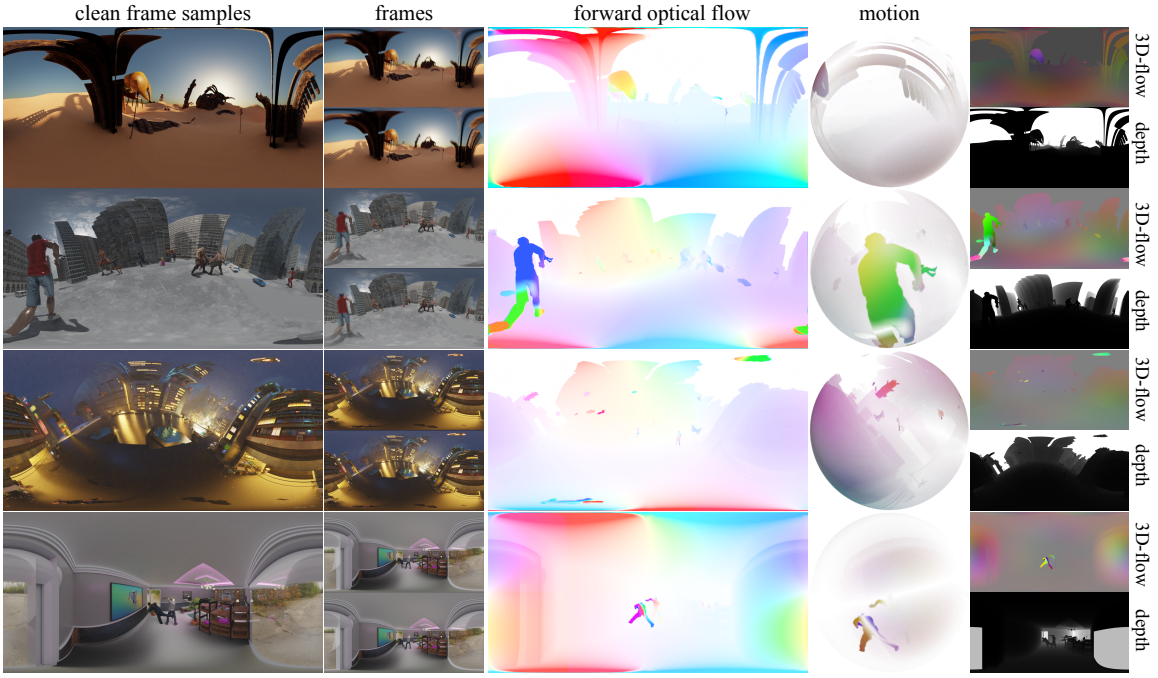


Figure 4.2: **The FLOW360 dataset.** Sample frames (first and second column, respectively) from some of the videos with corresponding forward optical flow and dynamic depth information. Motion in 3D Sphere (fourth column) is computed by transforming the motion vectors from Equirectangular plane (θ, ϕ) to unit sphere $f(x, y, z)$. Motion in the sphere is represented in RGBA color notation. RGB color representation (as suggested in Middlebury [86]) is encoded using (x, y) components, and the alpha color is encoded from z of a unit sphere. RGB encoding (fifth column) is an RGB color map of flow in 3D space. **Note**: flow fields are clipped for better visualization.

FLOW360 is an optical flow dataset tailored for 360° videos using Blender [110]. This dataset contains naturalistic 360° videos, forward and backward optical flow, and dynamic depth information. The dataset comprises 40 different videos extracted from huge 3D-World 'The Room', 'Modern', 'Alien Planet', and 'City Rush'. Due to their size, this 3D-World cannot be rendered at once in a single video. We render several parts of this 3D-World, which provides enough qualitative variation in motion and visual perception like 3D-assets, textures, and illuminations. The nature of this large and diverse animated world provides relatively enough

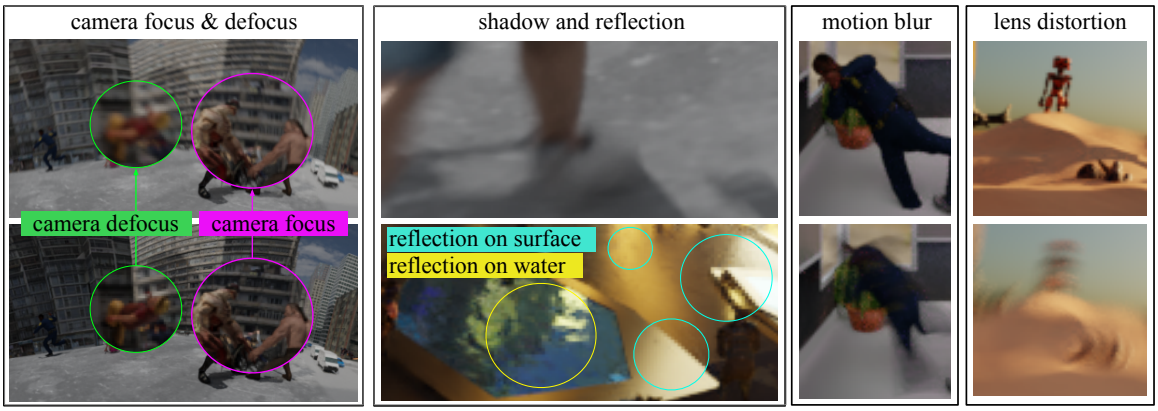


Figure 4.3: **Complexity of FLOW360 dataset.** Final frames in FLOW360 Dataset include complex characteristics like camera focus/defocus, motion blur, lens distortion, shadow, and reflections. Our dataset provides ambiance occlusion and environmental effects for a realistic visual appearance.

diversity to qualify for a standard benchmark dataset. The Fig. 4.4 shows some of the examples of motion and scene diversity of FLOW360. Similarly, samples from the dataset of different 3D-World are shown in Fig. 4.2. We build these 3D-World using publicly available 3D models [111, 112, 113] and 3D animated characters [114, 115, 116]. Meanwhile, we adopt Blender [110] for additional rigging and animation for the dataset.

FLOW360 contains 40 video clips extracted from different parts of huge 3D-World, 'The Room', 'Alien Planet', 'City Rush', and Modern'. The datasets also contain other information like depth maps and normal fields extracted from the 3D-World. The FLOW360 dataset has 4,000 video frames, 4,000 depth maps, and 3,960 flow fields. We divide the video frames into 2700/1300 train-test split. We render the video frames with the dimension of $(512, 1024)$ to save the rendering time. However, FLOW360 can be rendered with higher resolution, as 3D models and Blender add-ons will also be public.

Diversity. We design FLOW360 datasets to include a diverse situation that resembles the real world scenario as much as possible. The statistical validity of the datasets in terms of perceptual realism of scene and motion is presented in Fig. 4.5.

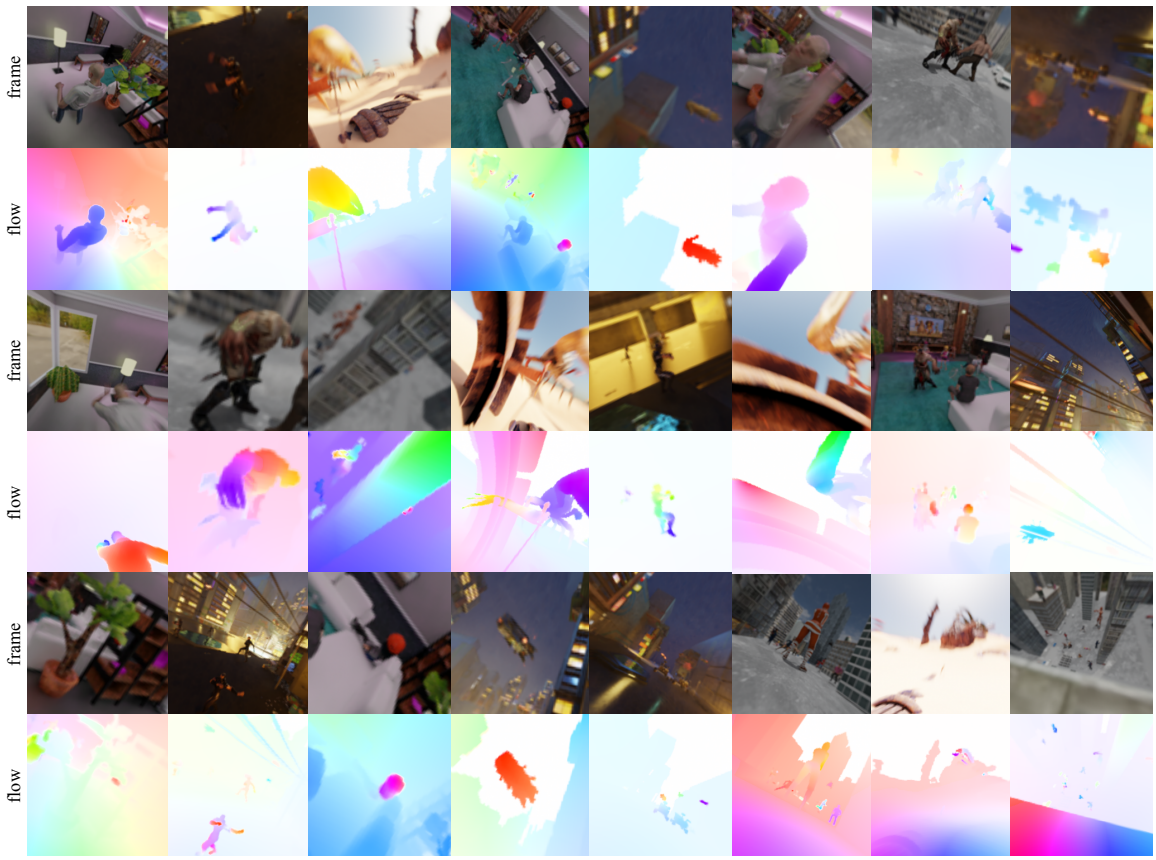


Figure 4.4: **Motion and scene diversity.** Samples from FLOW360 Dataset with random projection (pitch, roll, yaw, fov) showing scene and motion diversity. The FLOW360 dataset has a vast scene consisting of several lighting scenarios, textures, diverse 3D assets, and motion complexity in different regions.

The datasets contain a wide range of motion complexity from smaller to larger displacement, occlusion, motion blur, and similar complexities on the scene using camera focus-defocus, shadow, reflections, and several distortion combinations. As these complexities are quite common in natural videos, the FLOW360 provides similar complexities. Similarly, the datasets cover diverse scenarios like environmental effects, textures, 3D assets, and diverse illuminations. The qualitative presentation of these diversities and complexities are presented in Fig. 4.4 and Fig. 4.3 respectively.

Fairness. The FLOW360 dataset contains custom-tailored animated 360 videos. We plan to release the dataset with the 3D models and our custom Blender

add-ons to provide researchers a platform to create their custom optical flow datasets for all kinds of environments (perspective, 180° and 360° FOV). However, the release of 3D world scenes can raise questions regarding fairness. To mitigate this issue, we will perturb certain parts of 3D world scenes and not release any camera information related to the test set.

Flow-generator with Blender Add-ons. Flow-generator is a custom Blender add-on written for Blender-v2.92. The flow-generator serves two basic purposes. First, it creates a Blender compositor pipeline to collect frames, depth maps and optical flow information. This add-on can also collect additional information, such as normal maps. Second, it sets up a camera configuration for 360° FOV. We will describe details of the add-ons in supplementary material.

Render Passes. We exploit several modern features from Blender-v2.92 like advanced ray-tracing as a render engine along with render passes like vector, normal, depth, mist, and so on to produce realistic 3D scenes. Additionally, we incorporate features like ambient occlusion, motion blur, camera focus/defocus, smooth shading, specular reflection, shadow, and camera distortion to introduce naturalistic complexity (shown in Fig. 4.3) in our dataset. Besides optical flow information, the FLOW360 3D-world may be used to collect several other helpful information like depth, normal maps, and semantic segmentation.

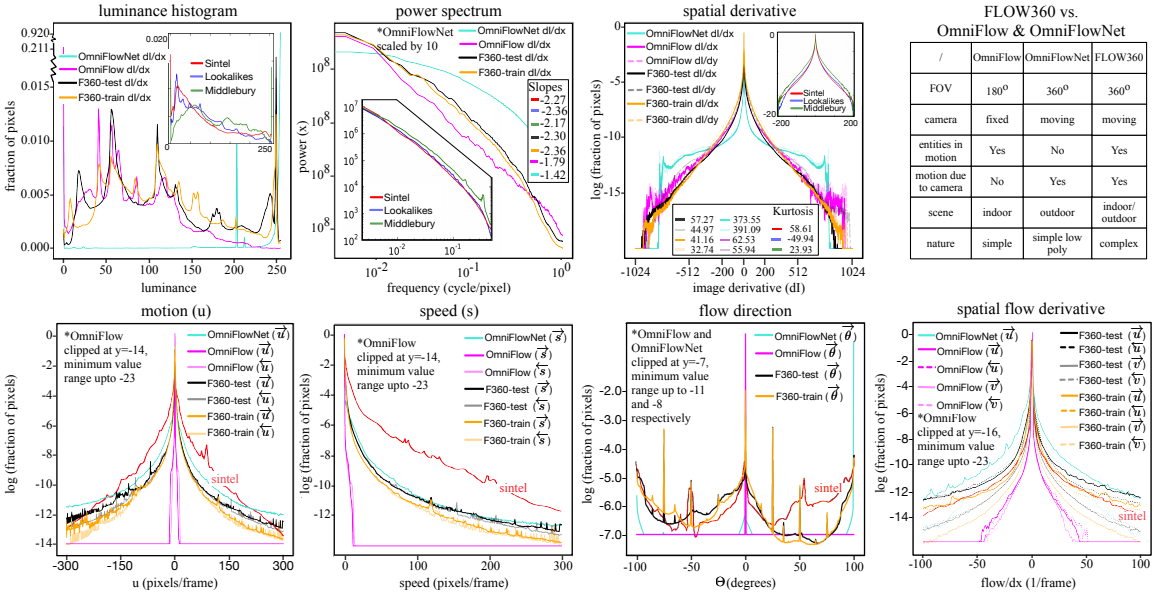
Dataset Statistics. We conduct a comprehensive analysis and compare our dataset with Sintel [2], Lookalikes (presented in the original Sintel paper to compare the image statistics with the simulated dataset), Middlebury [86], OmniFlow [90] and OmniFlowNet [91]. The analysis shown in Fig. 4.5 shows the image and motion statistics in the top and bottom rows, respectively.

Based on analysis from Sintel, we present frame statistics with three different analysis: luminance histogram, power spectrum, and spatial derivative. For luminance statistics, we convert the frames to gray-scale, $I(x, y) \in [0, 255]$ then we

compute histograms of gray-scale images across all pixels in the entire dataset. The luminance statistics show the FLOW360 has a similar distribution with the peak in the range between [0−100] and decreasing luminosity beyond that range. Similarly, we estimate power spectra from the 2D FFT of the 512×512 in the center of each frame. We compute the average of these power spectra across all the datasets. We present power spectra analysis separately for the training and test set in this analysis. The power spectra analysis closely resembles the Sintel, Lookalikes, and Middlebury datasets. Based on [101, 117], the real-world movies exhibit a characteristic of a power spectrum slope around -2, which is equivalent to a $1/f^2$ falloff. FLOW360 with the slope $(-2.30, -2.36)$ on test and training split shows such characteristics. We do not claim that FLOW360 is realistic, but it certainly exhibits perceptual similarity with natural movies. The spatial and temporal derivative analysis additionally supports this characteristic. The Kurtosis of frames spatial derivatives range from 32.74 to 57.27, peaked at zero. This characteristic shows that FLOW360 has a resemblance to natural scenes [101].

Regarding the flow field analysis we directly compare the distribution of motion $u(x, y)$, speed defined as $s(x, y)=\sqrt{u(x, y)^2 + v(x, y)^2}$, flow direction $\Theta(x, y)=\tan^{-1}(v(x, y)/u(x, y))$ and spatial flow derivative of u and v. The close resemblance of the flow field statistics between Sintel and FLOW360 suggests motion field resemblance with natural movies. Based on these comparisons, FLOW360 exhibits sufficient properties evident enough for its perceptual realism and complexities.

Comparison with OmniFlow and OmniFlowNet. OmniFlow [90] presents an omnidirectional flow dataset that is roughly similar to FLOW360. However, the major distinction between these datasets is the FOV. FLOW360 provides immersive 360° FOV, whereas OmniFlow provides only 180° FOV showing FLOW360 compared to OmniFlow is the true omnidirectional dataset. Similarly,



/	OmniFlow	OmniFlowNet	FLOW360
FOV	180°	360°	360°
camera	fixed	moving	moving
entities in motion	Yes	No	Yes
motion due to camera	No	Yes	Yes
scene	indoor	outdoor	indoor/ outdoor
nature	simple	simple low poly	complex

Figure 4.5: **Comparision of frames and flow statistics.** Top row represents the frames statistics and comparison with Sintel, Lookalikes, Middlebury, OmniFlow [90] and OmniFlowNet [91]. Bottom row represents flow statistics and comparison with Sintel (red), OmniFlow (magenta) and OmniFlowNet (turquoise). The table on the top-right shows a brief comparision of OmniFlow & OmniFlowNet with FLOW360 dataset. **Note**: ($\rightarrow$, $\leftarrow$) represents forward and backward flow fields, respectively.

OmniFlowNet [91] presents synthetic omnidirectional flow dataset with 360° FOV. However, this dataset contains low poly unnatural scenes, which can be explained by relatively larger kurtosis $(373.55, 391.09)$, characteristic of a power spectrum and luminance distribution (peaked at 255). The overall statistical analysis reveals FLOW360's better perceptual realism and diversity.

Applications. As we mentioned, the FLOW360 dataset contains frames and forward flow field and includes backward flow field, depth maps, and 3D-FLOW360 worlds, providing potential for applications like continuous flow-field estimation in 3 frames setting. Besides optical flow estimation, the FLOW360 dataset can be used in other applications such as depth and normal field estimation. Moreover, given 3D-FLOW360 animation data, the researcher can create as many optical flow datasets as needed.

SLOF

SLOF, as shown in Fig. 4.1, is inspired by the recent work on Siamese representation learning [109]. Since the method we rely on acts as a hub between several methods like contrastive learning, clustering, and siamese networks, it exhibits two special properties required for our case. First, this method has non-collapsing behavior. Second, it is useful when we have only positive discriminative cases. SLOF does not consider radial distortion mitigation via changing/transforming the convolution layers rather learns the equivariant properties of 360 videos via siamese representation. We claim that such transformation is trivial, based on the following fact. First, the omnidirectional videos are projected in angular domain, *w.r.t.*
polar(θ), **azimuthal**(ϕ); $\theta \in (-\frac{\pi}{2}, \frac{\pi}{2}), \phi \in (-\pi, \pi)$, so we can learn flow fields in these domains and convert these flow fields to spherical domain using planar to spherical transformations as shown in Eq. IV.1 and Eq. IV.2. Second, the intent of a convolution operator in optical flow architecture is relatively different from other applications like classification, detection, or segmentation network, where other tasks require convolution to learn relevant features (spatially consistent), the relevance of these features should stay consistent (strictly for better performance) throughout any spatial location of the images/videos. However, the convolution operation is dedicated to computing the pixel-wise displacement regardless of spatial inconsistency in the distorted region via equivariant representation learning [109]. Another important consideration of such a design is to make this method portable to any existing optical flow architecture. This design will eliminate the cumbersome architecture re-adjustments tasks and make it powerful and portable.

Mapping Flow Field to Unit Sphere. Input to our model are equirectangular images projected in angular domain **polar**(θ), **azimuthal**(ϕ), where

these angles are defined in radian as $\theta \in (-\frac{\pi}{2}, \frac{\pi}{2}), \phi \in (-\pi, \pi)$, thus the predicted optical flow is in (θ, ϕ). These flow fields can be converted to unit sphere using planar to spherical co-ordinate transformation as shown below:

$$(x_s, y_s, z_s) = (\sin\theta\cos\phi, \sin\theta\sin\phi, \cos\theta). \tag{IV.1}$$

We can compute sphere to catadioptric plane [118] projections to express the flow field in Cartesian co-ordinates as:

$$(x, y) = (\frac{x_s}{1 - z_s}, \frac{y_s}{1 - z_s}) = (\cot\frac{\theta}{2}\cos\phi, \cot\frac{\theta}{2}\sin\phi). \tag{IV.2}$$

Design. Given a pair of input image sequence X_1=(x_1, x_2), the rotation head (R) computes augmented view of this sequence as X_2=(x'_1, x'_2) with rotation r using a random combination of "pitch", "yaw" and "roll" operations. These two augmented views are passed as an input to an encoder network f, defined as f=$P(R'(\Theta(E(R(X, r)))))$ where E is a flow prediction module, RAFT [80] in our case, Θ is a mapping of 2D flow to unit sphere, R' is a reverse rotation operation and P is a convolution based down-sampling head. A prediction head presented as h (an MLP head), transforms the output from the encoder f from one stream to match the other stream. The illustration of this process shown in Equation. IV.3 as maximization of cosine similarity two views from siamese stream:

$$D(p^{left}, z^{right}) = -\frac{p^{left}}{||p^{left}||_2} \cdot \frac{z^{right}}{||z^{right}||_2}. \tag{IV.3}$$

Here, $p^{left} \triangleq h(f^{left}(X_1))$ and $z^{right} \triangleq f^{right}(X_2)$ denotes the output vectors to match from two different streams(f^{left}, f^{right}). This maximization problem can be viewed from another direction, with (p^{right}, z^{left}) as the second matching pair from siamese stream (f^{right}, f^{left}) respectively. Given two matching pairs, we can use following

(Eq. IV.4) symmetrized similarity loss function (note that z^{left} and z^{right} are treated as a constant term using stop-grad operations to prevent a degenerate solution due to model collapse [109]):

$$L_{sim} = \frac{1}{2}D(p^{left}, z^{right}) + \frac{1}{2}D(p^{right}, z^{left}). \tag{IV.4}$$

Similarly, the optical flow loss is computed as a sequence loss [80] over predicted flow field and ground truth. This loss (l_1 distance over predicted and ground truth flow f_{gt}) is computed and averaged over sequence of predictions iteraterively generated for the same pair of input frames $\{f_1, f_2, ..., f_n\}$=$E(R(X, r))$ as shown in Equation. IV.5, where γ=0.8^{n-i-1} served as weights over sequence loss. Note that (n, i) denotes number of prediction(n) in sequence and prediction id(i) in predicted flow sequences. The design of the weighted schemes ensures different levels of confidence on predicted flows over time.

$$L_{flow} = \sum_{i=1}^{n} \gamma ||R(f_{gt}, r) - f_i||. \tag{IV.5}$$

Given similarity loss(L_{sim}) and flow loss(L_{flow}) we implement a hybrid loss function L=L_{sim}+L_{flow}. The overall objective of this loss function is to maximize the similarity between latent representation of flow information while minimizing the loss between ground truth and predicted optical flow.

Experiments

We evaluate SLOF on the FLOW360 test set. We use pre-trained RAFT on Sintel [2] and fine-tune on FLOW360 as a comparison baseline. The fine-tuning process is done using training protocols suggested in [80]. Moreover, to make a fair comparison with traditional methods, we transform RAFT (pre-trained) to adapt spherical convolution using KTN [36]. KTN transforms the convolution kernel to

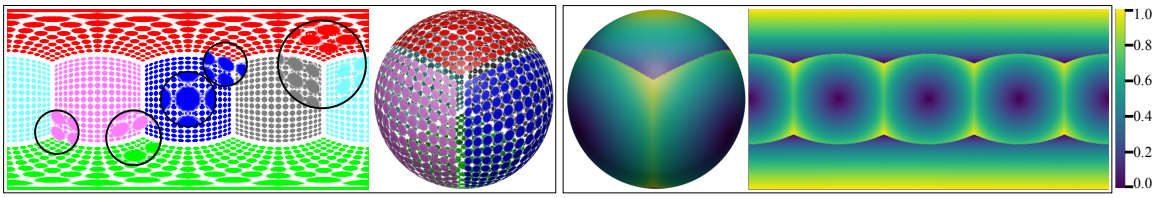


Figure 4.6: **Distortion density map**. Illustrating different distortion intensity due to equirectangular projections. Left: upper (red) and lower (green) part of projections shows higher distortion in central part where as the equatorial region (cyan, pink, blue, gray) exhibit higher distortion rate away from the center of tangential plane. Right: shows the distortion density from $(0, 1)$. This distortion density map is used to evaluate the distortion aware EPE (EPE$_d$). **Note**: Each circle patch in left spherical projection have same area.

mitigate the radial distortions via estimating the spherical convolution function. Additionally, we run ablation studies on different training strategies and propose a distortion-aware evaluation. We will present details of the training procedure in the supplemental material.

Scope. The scope of our experiments are two folds: First, create a baseline for future researchers to explore novel methodologies. Second, address the validity of our method based on the fair comparison with a flow network designed for a spherical dataset. We formulate our baseline experiment on perspective optical flow network RAFT and modified version of RAFT with KTN [36] to compare the performance. The RAFT+KTN architecture simulates a domain adaptation similar to approaches like [89, 91]. We choose KTN because of its success over alternative approaches like [46, 47, 38, 48, 37]. It is worth noting that the design of omnidirectional flow estimation can be extended to several techniques involving mitigation of radial distortions, making it practically impossible to cover all.

Augmentation Strategy. Given the nature of SLOF, we can train it using two different training strategies (**v1**,**v2**) as shown in Fig. 4.1(right). These strategies can be achieved by performing different rotational augmentation on the input sequences. The first strategy (**v1**) can be achieved by using set of inputs $(R(X_1, r_1), R(X_2, r_2))$ where r_1=$(0, 0, 0)$, i.e., X_1 does not have any rotational

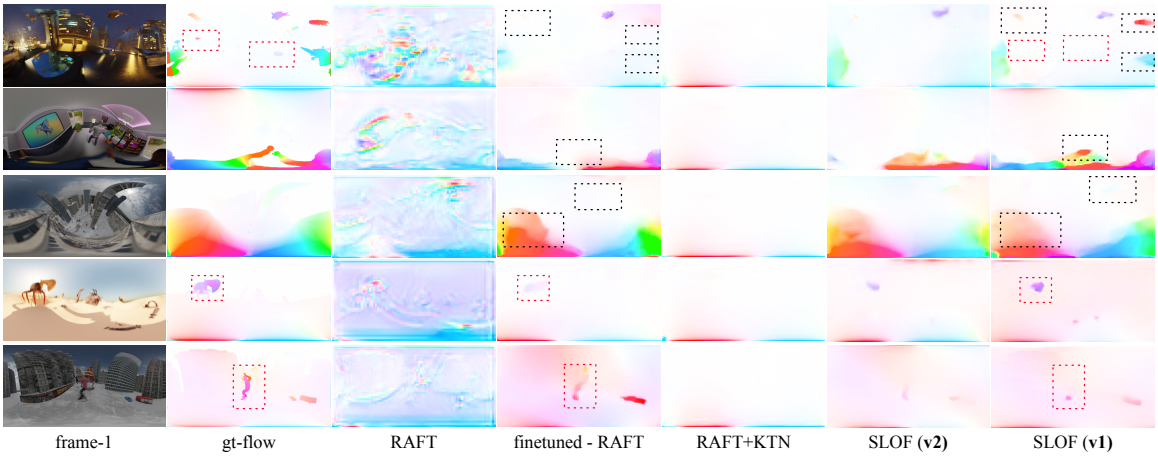


Figure 4.7: **Qualitative results on FLOW360 test set.** Qualitative results show our best model SLOF(**v1**) shows better results compared to fine-tuned RAFT trained with policy explained in [80]. The dotted (**black**) rectangle indicates the comparative improvements of our model over fine-tuned RAFT. RAFT+KTN method fails to predict flow-field correctly; instead, it only predicts shallow flow fields from camera motion. The weakness of our model can be seen on dotted (red) rectangle where smaller motion segments are missing. **Note:** Flows information is clipped for better visualization.

augmentation, whereas $r_2 \neq (0, 0, 0)$ has rotation defined with random combinations of "pitch", "roll", and "yaw" operations. This setting is kept consistent throughout the training process. Alternatively, identical augmentation can be achieved by flipping this augmentation protocols. The second rotational scheme (**v2**) can be achieved by randomly switching rotation such that when r_1 is none, the r_2 is some random rotational augmentation and vice versa. This approach performs on par with **v1**.

$$\text{AE} = \arccos\left(\frac{u_e u_r + v_e v_r + 1}{\sqrt{u_r^2 + v_r^2 + 1}\sqrt{u_e^2 + v_e^2 + 1}}\right). \tag{IV.6}$$

$$\text{EPE} = \frac{1}{N}\sum_{i}^{N} ||f_{pred} - f_{gt}||_2. \tag{IV.7}$$

$$\text{EPE}_d = \frac{1}{N}\sum_{i}^{N} \frac{||f_{pred} - f_{gt}||_2}{1 - d}. \tag{IV.8}$$

Table 4.1: **Quantitave results on FLOW360 test set.** $*$ denotes that we use EPE_d/AE_d as the metrics; otherwise, the normal EPE and AE. Compared to baseline, SLOF achieves lower end-point-error and angular error on both distortion aware (EPE_d and AE_d) and normal scheme. In terms of end-point-error (lower the better) our model (**v1**,**v2**) outperforms all the baseline. Similarly in terms of angular error (lower the better) our models (**v1**, **v2**) perform comparatively similar and outperform all the baseline. Though RAFT+KTN achieves comparable normal EPE, the distortion aware (Weighted) metrics (EPE_d and AE_d) are significantly larger. **Note**: metrics in range (all, less than (5, 10, 20) and greater than 20) is computed as an average, based on the speed ($\mathbf{s}$(x,y)=$\sqrt{u(x,y)^2 + v(x,y)^2}$) only in the respective pixel regions.

Mehtod	Version	Metric	Weighted $\mathbf{s}\geq 0^*$	$\mathbf{s}\geq 0$	$\mathbf{s}<5$	$\mathbf{s}<10$	$\mathbf{s}<20$	$\mathbf{s}\geq 20$
Baselines	RAFT [80]	EPE	3.344	2.058	0.558	0.682	0.838	71.736
		AE	1.120	0.820	0.825	0.821	0.819	0.868
	Finetuned RAFT [80]	EPE	2.635	1.624	0.314	0.393	0.509	65.340
		AE	0.745	0.522	0.527	0.522	0.520	0.647
	RAFT + KTN [36]	EPE	3.899	2.222	0.598	0.742	0.924	76.426
		AE	2.020	0.912	0.912	0.910	0.911	1.0114
SLOF	Switch rotation (**v2**)	EPE	2.626	1.615	0.326	0.401	0.512	64.678
		AE	**0.691**	**0.485**	**0.489**	**0.484**	**0.482**	0.659
	Single rotation (**v1**)	EPE	**2.548**	**1.568**	**0.309**	**0.387**	**0.502**	62.476
		AE	0.708	0.497	0.501	0.497	0.495	**0.607**

Evaluation Strategy. We evaluate our method based on 2D-raw flow. Besides, using EPE (End Point Error in Eq. IV.7), i.e., Euclidean distance between the predicted flow and ground truth flow, as a single evaluation metric, we incorporate AE (Angular Error) as shown in Eq. IV.6 as the second measure. To explain the error in the omnidirectional setting, we introduce a distortion-aware measure called EPE_d as in Eq. IV.8. This metric penalizes the error in the distorted area based on the distortion density map.

As EPE_d, AE_d is calculated as $\frac{1}{N}\sum_i^N \frac{\text{AE}}{1-d}$ where, d represents the distortion density map illustrated in Fig. 4.6, f_{pred}=(u_e, v_e) represents predicted flow, and f_{gt}=(u_r, v_r) represents ground truth flow. Note that, to maintain lower metrics scale the distortion density is mapped between $[0.500, 1.000)$ from $(0.0, 1.0]$. Please refer to supplemental for additional details on distortion density map.

Results. Fig. 4.7, Fig. 4.8 and Table 4.1 summarize our experimental results.

The overall summary of qualitative results is presented in Fig. 4.7. SLOF performs better than baseline RAFT and kernel transformed RAFT+KTN methods. This result is evident enough to show that siamese representation learning can exploit the rotational properties of 360° videos to learn omnidirectional optical flow regardless of explicit architecture adjustments.

Our methods, SLOF (**v1**,**v2**) perform better than presented baselines.Among these methods **v1** has the best EPE score whereas, **v2** has better AE score. However, AE on both **v1** and **v2** are relatively similar, suggesting **v1** as our best method. This is clearly visible in qualitative results shown in Fig. 4.7.

By investigating distortion-aware EPE, we can see that RAFT with KTN achieves significantly higher EPE regardless of comparable normal EPE with the other methods. This clearly explains why RAFT+KTN methods could not predict the motion around the distorted area; instead, it predicts shallow flow fields due to camera motion only. Moreover, comparing qualitative results in Fig. 4.7 and EPE measure in different distortion ranges in Fig. 4.8, we can see that our best method can predict smoother flow fields compared to baseline methods. These fields in the polar region are comparatively better and have better motion consistency in the edge region. However, our model might fail to predict relatively smaller motion regions in some cases, which leaves room for future improvements based on the proposed method. This concludes that RAFT+KTN requires additional re-engineering and domain adaptation, which is out of the scope of current work.

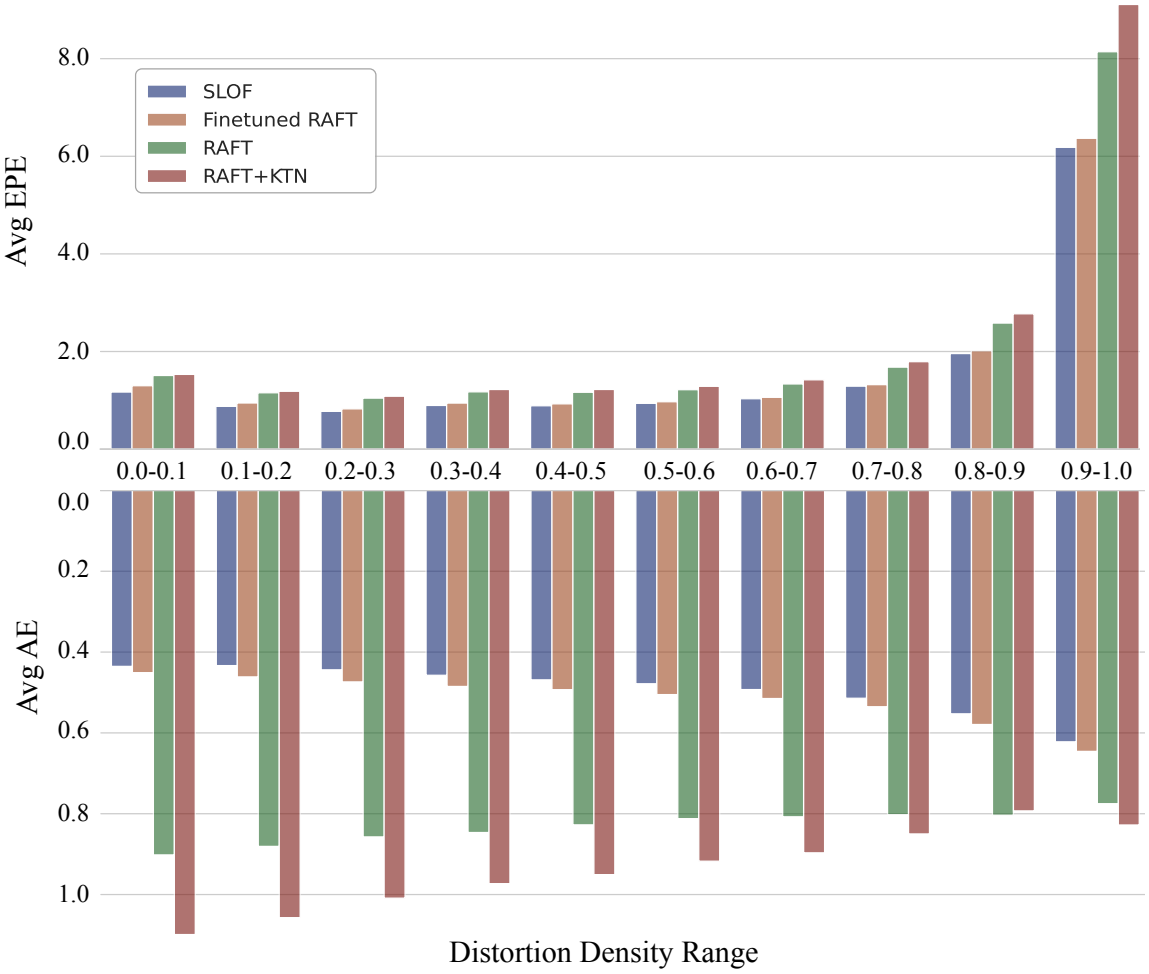


Figure 4.8: **Error distribution plot**. Illustrating error (EPE and AE) in different distortion density ranges. SLOF relatively performs better in all distortion density ranges.

Conclusion

Omnidirectional flow estimation remains in its infancy because of the shortage of reliable benchmark datasets and tedious tasks dealing with inescapable radial distortions. This chapter proposes the first perceptually natural-synthetic benchmark dataset, FLOW360, to close the gap, where comprehensive analysis shows excellent advantages over other datasets. Our dataset can be extended for other non-motion applications like segmentation and normal estimation task as well. Moreover, we introduce a siamese representation learning approach for

omnidirectional flow (SLOF) instead of redesigning the convolution layer to adapt omnidirectional nature. Our method leverages the invariant rotation property of 360° videos to learn similar flow representation on various video augmentations. Meanwhile, we study the effect of different rotations on the final flow estimation, which provides a guideline for future work. Overall, the elimination of network redesigns aids researchers in exploiting existing architectures without significant modification leading faster deployment in real world setting.

V. VIT360: EGOCENTRIC ACTIVITY RECOGNITION VIA SIAMESE REPRESENTATION LEARNING IN 360° VIDEOS

Introduction

The increased interests in wearable camera sensors open an interesting area of research in computer vision, commonly termed as Egocentric Activity Recognition (EAR) [119]. The activity recognition is critical in the video understanding domain with many real-world applications like surveillance, video retrieval, video summarization, and human-computer interaction. The growth of 360° videos reuslts in interesting extension of egocentric activity recognition domain commonly referred as EAR360 [120] meanwhile adding several challenges relating to adaption with complex nature of 360° videos.

Human activity recognition focuses on identifying characteristic activities performed by humans in a video sequence. This task, in general, is formulated as a multi-class classification problem of accurately predicting the activity labels. Hand-crafted features [121, 122, 123, 124] were used as the basis of activity recognition prior to the advent of deep learning. In recent years, Convolution Neural Networks (CNNs) have been widely used for activity recognition [125, 10, 126, 127, 128, 129, 130, 9, 131, 132]. However, these models are not easily transferable to egocentric activity recognition on 360° videos. The 360° videos or spherical videos exhibit a unique property of radial distortions when projected in a plane. Spherical video-based applications suffer a considerable performance loss in CNNs due to the distortion variant [133, 37] nature of convolution operations. Generally speaking, such issues can be solved by learning spherical representation [36] or learning spherical CNN [46, 47]. These increased overheads require several modifications of the existing architectures with additional domain adaptation [89]. Besides, the adaptation of 3D convolution imposes new

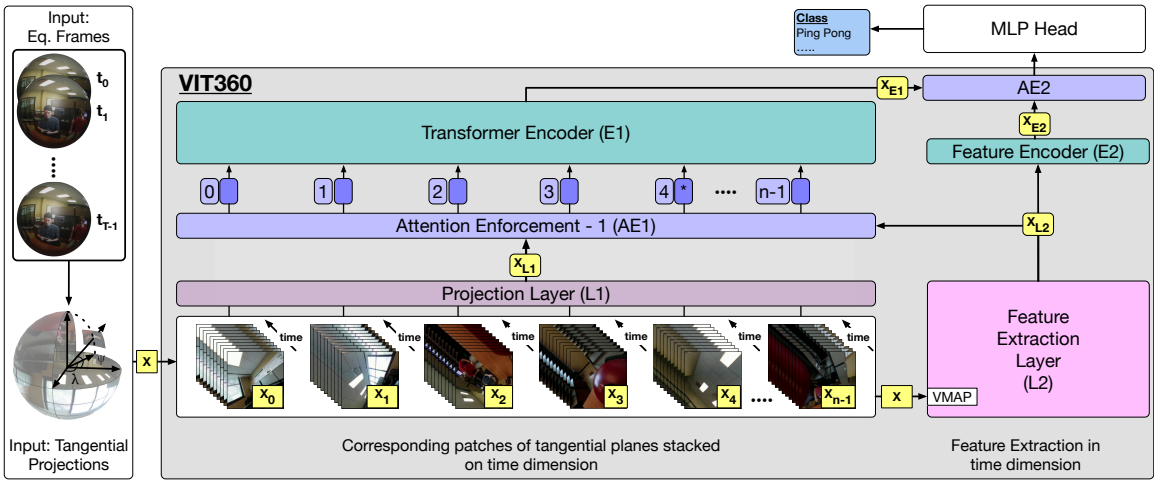


Figure 5.1: **VIT360 framework.** The sequence of ten consecutive frames (or optical flow) is sampled into six different tangential projections. These tangential projections are passed through Projection Layer(L_1) and Feature Extraction Layer (L_2) to compute features X_{L1} and X_{L2}. The First Attention Enforcement Layer (AE1) computed the attention map between these features and passed it to Transformer Encoder (E1) with positional embedding as an input. Final Attention Enforcement Layer (AE2) computes the attention maps between the output X_{E1} and X_{E2} and passes it as an input to MLP Head for activity classification.

challenges such as complex architecture design and computational bottleneck. Recent advancements in spherical convolution techniques have primarily focused on convolution on tangential patches [88], which demonstrates the potential of such methodologies in downstream computer vision tasks. On the other hand, attention-based frameworks [134, 135] are outperforming traditional CNNs on still image applications [136, 4] as well as video-based applications [137, 138, 139, 140]. In this chapter, we propose a novel egocentric activity recognition framework called VIT360 (shown in Fig. 5.1) by combining tangent view-based convolution [88] with vision transformer [4]. VIT360 advances the state-of-the-art research in EAR360 because: i) it follows a hybrid design pattern where we implement the framework of 3D convolution for feature extractions which are later fed into the transformer encoder for learning representations; ii) it eliminates the impact of radial distortions by limiting the convolution operations only to tangential planes; and iii) it provides

a unique way to exploit the advancement of transformers in egocentric activity recognition.

Research in egocentric activity recognition in 360° videos is relatively less explored. One such work proposed in EGOK360 [120] implements modified resnet [33] and I3D [9] based frameworks for action/activity recognition. Based on the results presented in EGOK360, these performance-centric methods achieve relatively higher accuracy on both action and activity classification tasks. However, given EGOK360 being a dataset collected in controlled settings, 360° videos in the dataset exhibit consistent fixed field-of-view. The prior results were derived from experiments with controlled settings (*i.e.*, the training and inference were performed using fixed field-of-view projections). Nevertheless, such controlled settings are rarely seen in real-world applications as the person/agent with wearable sensors is subjected to maneuver randomly, resulting in a random field-of-view. As a result, previous methods for egocentric activity recognition perform poorly on random projections [120], which makes them unusable in real-world settings. Our method solves this problem by exploiting siamese representation learning [109] to build a projection invariant framework for activity recognition. We achieve that by pre-training VIT360 to maximize latent representation similarity across different rotational augmentation of the same input data. VIT360 with rotation-invariant property provides an effective solution for real-world egocentric activity recognition in 360° videos.

Optical flow constitutes the temporal characteristic of motion between consecutive frames. Our VIT360-based model uses optical flow as input for the egocentric activity recognition to capture the temporal characteristic-based activity features. These optical flow information can be extracted using popular off-the-shelf frameworks as [80, 45, 44, 3, 104, 32]. However, these architectures are designed for optical flow estimation of perspective videos only, which does not work for the

optical flow estimation in 360° videos. The main challenges are the radial distortion seen in the equirectangular projection of these 360° videos. Recently, several works [89, 91] are proposed for 360° optical flow estimation but they introduce significant overheads of domain adaptation, training, and fine-tuning on target videos. Given that egocentric activity recognition is the primary goal, we intend to use off-the-shelf architectures to extract optical flow in 360° videos. We demonstrate that 360° optical flow can be seamlessly integrated into VIT360 with improved performance. This is achieved by leveraging PerceiverIO [3], a recently proposed transformer-based optical flow estimation technique by DeepMind. We use a pre-trained PerceiverIO for optical flow inference only. This inference technique exercises the similar approach mentioned in VIT360.

To summarize, the nature of 360° videos imposes three significant challenges to EAR360 research: i) Performance deterioration due to random field-of-view projection; ii) Radial distortions caused by sphere-to-plane projections; and, iii) High overheads of model adaptation, transformation and refinement for 360° optical flow. This chapter proposes a novel rotation invariant egocentric activity recognition framework VIT360 for 360° videos. The design of VIT360 tackles challenges of processing 360° videos with a fully convolution-based framework while learning the rotational invariant representation. VIT360 is collectively inspired by the current trend in transformer-based techniques combined with siamese representation learning. Our contributions are three folds: i) Transformer based novel activity recognition framework, ii) Rotation invariant framework for 360° videos, and iii) Seamless integration of optical flow inference for 360° videos with existing frameworks.

Related Work

CNN based Activity Recognition. The convolution-based action/activity recognition framework has been the standard practice in the activity recognition field. Though 2D CNN shows massive success in still image-based applications, incorporating the temporal aspects for the downstream task is challenging because the CNN alone does not account for time. The initial work [125] shows a way to use CNN to incorporate temporal aspects for the video classification task. Similarly, the two-stream architecture [10] learns spatio-temporal features from input RGB and optical flow information for video classification. Many other approaches [126, 127, 128, 129] leverage the recurrent neural network to model long-term dependencies across input video frames. Following the first introduction of 3D CNN based video approach [130], several other variants of 3D CNN based approaches [9, 131, 132, 9] were proposed. Our transformer-based approach, a hybrid method, takes inspiration from these methods to extract initial video features.

Attention based Activity Recognition. Though initially introduced in NLP [134], the attention mechanism in deep learning has changed several aspects of computer vision research. Ranging from still image-based vision tasks [136] to video action classification task [141], the attention-based mechanism is drawing increasing interest in application areas like the activity recognition domain. The introduction of the paper, "Attention is all you need" [135] marks a significant turning point in transformer-based approaches. This attention framework led to the notable outcome of the transformer-based vision applications as Vision Transformer [4] and activity recognition frameworks [137, 138, 139, 140]. However, research in egocentric activity recognition in 360° videos is still in its infancy. Inspired by recent advancements in transformer-based vision tasks, we combine the potential of vision transformer with traditional convolution-based techniques for egocentric activity recognition.

Siamese Representation Learning. The siamese networks have succeeded in many vision-related tasks [106, 107]. Recently, these networks are attracting attention in unsupervised representation learning techniques. One of the notable work SimSiam [109] tries to learn augmentation invariance representation via training the networks with different augmentation views of the same input, resulting in improved performance. We apply this approach to exploit rotational invariance in 360° videos to train a rotational invariant activity recognition model for 360° videos.

360° optical flow. Following the success of Flownet [44], several CNN based deep learning frameworks [32, 45, 53, 51, 52, 54, 55, 56, 80] have shown consistent improvements over time. However, perspective video-based frameworks impose several challenges in estimating optical flow in 360° videos. Architecture designed for 360° videos based optical flow framework, LiteFlowNet360 [89] and OmniFlowNet [91] propose several modifications on existing CNN based architectures [45, 104]. These modifications include an adaptation of existing convolution layers to 360° using techniques as [36, 105]. As a secondary task in egocentric activity recognition, the overheads mentioned above in 360° optical flow estimation create considerable lag in the research. Instead of training, adapting, and finetuning the existing architectures, improvisation on optical flow inference can significantly alleviate such issues. The recent success on transformer-based techniques called PerceiverIO by DeepMind [3] takes several input patches of the input frames. It predicts intermediate optical flow, which can be later warped together to make a final flow. We modify the PerceiverIO inference process to adapt optical flow inference in 360° videos using the input patches as a different rotational view of the same 360° videos and later warping these intermediate flows to 360° representation.

Dataset. Although there are affluent datasets in the egocentric activity recognition literature, most of the published datasets [23, 24, 25, 26, 27] are limited

to the perspective field-of-view. The 360° videos based egocentric activity recognition datasets are relatively scarce. EGOK360 [120] is a recently published egocentric dataset with 360° field-of-view, which includes two confusing terminologies called activity and actions. The activity is defined as a collection of minor actions, where the actions are more fine-grained motions related to egocentric activities. We perform our experiments on these datasets focusing only on the activity recognition (twelve different classes).

Method

The overall design of egocentric activity recognition in 360° videos comprises four different components: (1) design of VIT360, (2) learning projection invariant representation of 360° videos, (3) computing 360° or omnidirectional aware motion features from off-the-shelf architecture, and (4) two-stream approach for activity classification. The following subsections will discuss each of these components in chronological order.

Design of VIT360

VIT360 requires input pre-processing which converts an equirectangular image into multiple tangential patches covering the entire field-of-view as shown in Fig. 4.1. Instead of processing raw image patches in the time dimension, these patches are processed first via the global projection layer (L1) to compute feature sequences. Compared with the original VIT [4] implementation, input features to VIT360 are relatively larger as the patches are fed in the time dimension. The projection layers transform the larger input feature space into a relatively smaller feature space by downsizing the transformed feature dimension. The feature extraction layer (L2) acts as a siamese stream for learning additional features from independent tangent patches in the time dimension to mitigate the low parametrization of features. Using

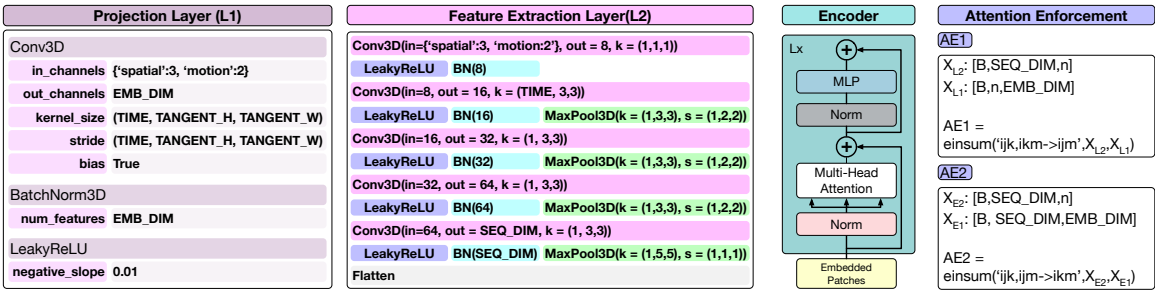


Figure 5.2: **Layers in VIT360.** VIT360 is composed of four major components: Projection Layer (L1) computes the initial flattened features of input sequences, and Feature Extraction Layer (L2) computes additional features using the siamese stream. Attention Enforcement Layers (AE1 and AE2) compute the attention maps computed across two different feature extraction layers and encoder streams, respectively. Finally, a pair of Transformer Encoder (E1 and E2) to learn temporal features for activity recognition.

a projection layer (L1) and feature extraction layer (L2) with stacked convolution architecture makes VIT360 a hybrid architecture.

Tangential Projections: VIT360 takes input of 360° video frames or optical flows in time dimension ($X \in \mathbb{R}^{C \times T \times H \times W}$), resulting in spatial modality (where C = 3) and motion modality (where C = 2) respectively. Note that T represents the number of consecutive frames denoting the time dimension. Each input at $time(t)$ are in fact projected in the equirectangular plane($X^t \in \mathbb{R}^{C \times H \times W}, 0 \leq t \leq T$), which is obtained by a sphere mesh unwrapped on a flat rectangular plane surface. This process maps sphere latitude and longitude to horizontal and vertical coordinate systems expressing the length and height of the plane in the range $(-\pi, \pi)$ and $(-\pi/2, \pi/2)$ respectively. In order to create input patches, tangential planes are sampled using a spherical to cartesian coordinate transformation system. This is shown in Eq. (V.1), where (λ, ψ) represents latitude and longitude such that $(\lambda_0, \psi_0) = (0, 0)$ represents the centre of the plane, and (c) represents the angular

distance of the $point(x, y)$ from the centre of the projection.

$$
\begin{aligned}
x &= \frac{\cos\psi \sin(\lambda - \lambda_0)}{\cos(c)}, \\
y &= \frac{\cos\psi_0 \sin\psi - \sin\psi_0 \cos\psi \cos(\lambda - \lambda_0)}{\cos c}, \\
\cos(c) &= \sin\psi_0 \sin\psi + \cos\psi_0 \cos(\psi) \cos(\lambda - \lambda_0).
\end{aligned}
\tag{V.1}
$$

Similarly, the inverse map from plane to sphere can be computed using the following equation as

$$
\begin{aligned}
\psi &= \sin^{-1}\left(\cos(c)\sin\psi_0 + \frac{y\sin(c)\cos\psi_0}{\rho}\right), \\
\lambda &= \lambda_0 + \tan^{-1}\left(\frac{x\sin(c)}{\rho\cos\psi_0\cos(c) - y\sin\psi_0\sin(c)}\right)
\end{aligned}
\tag{V.2}
$$

In the above equation, $\rho = \sqrt{x^2 + y^2}, c = \tan^{-1}\rho$. The projection layer samples n tangential planes ($\mathbb{R}^{C\times T\times h\times w}$) covering the entire 360° field-of-view, where (h, w) are the height and width of each tangential plane. These tangential planes are linearly stacked alongside the width to obtain a final output $x \in \mathbb{R}^{C\times T\times h\times nw}$. In summary, the tangential projection layers take a series of optical flows or frames in a video ($X \in \mathbb{R}^{C\times T\times H\times W}$) and transform it into a series of tangential plane projections ($x \in \mathbb{R}^{C\times T\times h\times nw}$) linearly stacked along the width.

Projection Layer (L1): An overview of the Projection Layer (L1) is shown in Fig. 5.2. This layer takes tangential patches ($x \in \mathbb{R}^{C\times T\times h\times nw}$) in time dimension as input. In contrast with the original VIT implementation, the input to VIT360 is a 3D image (considering the time dimension) which requires a modification of the original projection layer to 3D convolution-based architecture. The Projection Layer (L1) includes a 3D convolution with 3D batch-normalization and LeakyReLU layer as an activation function. The 3D convolution layer is parametrized with the 3D kernel size and stride as (TIME, TANGENT_H, TANGENT_W), where

TANGENT_H = h, TANGENT_W = w refers to the height and width of tangent patches and TIME = T refers to the number of frames in the time dimension. This formulation of L1 transformed input tangential patches ($x \in \mathbb{R}^{C \times T \times h \times nw}$) to ($x \in \mathbb{R}^{n \times EMBED\_DIM}$), where EMBED_DIM refers to the output feature dimension in Fig. 5.2.

Feature Extraction Layer (L2): The Projection Layer (L1) inspired from original VIT implementation maps a huge feature space ($x \in \mathbb{R}^{C \times T \times h \times nw}$) into relatively smaller feature ($x \in \mathbb{R}^{n \times EMBED\_DIM}$) embedding. This low parametrization of the feature space resulted in VIT360 performance degradation. To mitigate this issue, an additional Feature Extraction Layer (L2), as shown in Fig. 5.2 is designed to learn additional features. These additional features are later used for attention enforcement, resulting in improved performance. The Feature Extraction Layer (L2) is implemented as a siamese network to process individual input tangential patches. To achieve the siamese network, we use **vmap** (available in pytorch framework) operation per tangential patches. The Feature Extraction Layer (L2) takes an input ($x \in \mathbb{R}^{C \times T \times h \times nw}$), reshapes it to ($x \in \mathbb{R}^{n \times C \times T \times h \times w}$) and computes features ($x \in \mathbb{R}^{n \times SEQ\_DIM}$) where the SEQ_DIM is an output embeddings of the Feature Extraction Layer (L2).

Attention Enforcement (AE1 and AE2): The Encoder in the original implementation of VIT takes input from the linear projection layer from 16×16 image patches. Such design consideration in VIT360 is computationally challenging since the input for VIT360 is stacked patches in the time dimension. It is possible to generate a relatively larger number of tangential planes and subsequently larger embedding space from the Projection Layer (L1). Such a design is computationally infeasible because the input will be relatively larger than the original VIT. This tradeoff between the choice of the number of input patches and computational cost directly affects model performance. In addition, the Projection Layer (L1) reduces

the feature space significantly, as we discussed above, leading to lower parametrization and reducing the model's performance. In order to mitigate such issues and maintain the optimal input size, we introduce Attention Enforcement (AE1 and AE2) techniques. The first Attention Enforcement (AE1) computes an attention map between the output from L1(X_{L1}) and L2(X_{L2}) making the input ($X \in \mathbb{R}^{SEQ\_DIM \times EMB\_DIM}$) size of the encoder sufficiently optimal retaining the model performance. Similarly, the second Attention Enforcement(AE2) computes the attention maps ($X \in \mathbb{R}^{n \times SEQ\_DIM}$) between the output from E1($X_{E1} \in \mathbb{R}^{SEQ\_DIM \times EMB\_DIM}$) and E2($X_{E2} \in \mathbb{R}^{SEQ\_DIM \times n}$), keeping the number of parameter ($n \times EMB\_DIM \times NUM\_CLASSES$) in MLP Head fixed. The overview of Attention Enforcements is presented in Fig. 5.2.

Encoder Layer (E1 and E2): The Transformer Encoder layer E1 and E2 contains similar implementation as presented in the original VIT implementation. The E1-encoder contains two layers of encoder architecture, whereas E2 contains only one layer of encoder architecture. E1 takes an input ($X_{AE1} \in \mathbb{R}^{SEQ\_DIM \times EMB\_DIM}$) from the Attention Enforcement Layer (E1) where EMBED_DIM is the input dimension of the E1-encoder. Similarly, E2 takes an input ($X_{L2} \in \mathbb{R}^{n \times SEQ\_DIM}$) from the Feature Extraction Layer (L2) where SEQ_DIM is the input dimension of the E2-encoder. Both E1 and E2 computes output feature ($X_{E1} \in \mathbb{R}^{SEQ\_DIM \times EMB\_DIM}, X_{E2} \in \mathbb{R}^{n \times SEQ\_DIM}$) vectors similar to the input dimension.

MLP Head: In contrast to the original VIT implementation, we introduce a single layer MLP head as a classification head to predict the activity classes. The Attention Enforcement Layer (AE2) output is passed as an input to the final MLP Head.

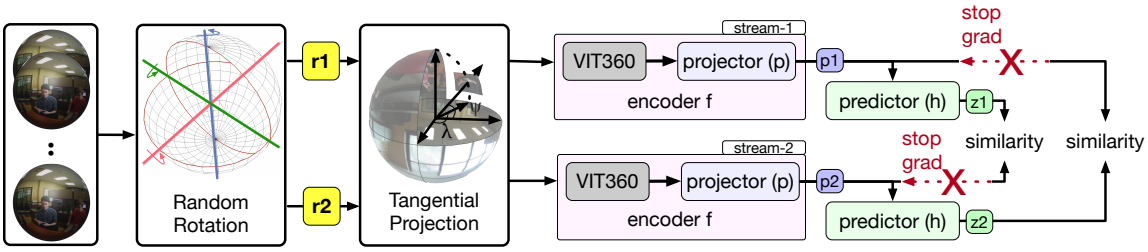


Figure 5.3: **Siamese representation learning for egocentric activity recognition.** The random Rotation head (R) computes two different rotational views of the same input video sequences. These augmented views are passed as an input to VIT360 using tangential projections. The siamese network of VIT360 computes latent representation, p_i and z_i, using projector(p) and predictor (h) respectively. The training policy maximizes the similarity between latent representations (p_1, z_2) and (p_2, z_1) in unsupervised manner. The pre-trained network with this policy is then subjected to training for egocentric activity recognition.

Projection Invariant Representation

360° videos in comparison with perspective videos exhibit a unique property of unlimited field-of-view, resulting in infinite projections. Such infinite projections can be achieved by rotating the 360° videos on three different axes (X, Y, Z), namely "pitch", "roll", and "yaw" operations. Regardless of these projections, the overall information of 360° videos remains intact. This rotation-invariant property of 360° videos is crucial while designing deep learning architecture for 360° videos based applications. In practice, the ideal architectures should be rotationally invariant or projection invariant. To achieve this goal, we perform pretraining of VIT360 to learn projection invariant representation.

Pre-training of VIT360 for projection invariant representation is loosely based on contrastive learning approach called SimSiam [109]. The overview of pre-training stage is illustrated in Fig. 5.3. This representation learning based on SimSiam is completed using siamese representation learning, formulated as a pair of weight sharing VIT360 streams (stream-1, stream-2) to maximize the similarity between two different representations of the same 360° videos information (both frames and

optical flow). Given a sequence of frames $X = (x_0, x_1, ..., x_{T-1})$, where T is the number of frames in sequence in time, the rotation head (R) computes a pair of rotational augmentation $X_i = R(X, r_i)$ using $r \in (r_1, r_2)$ defined as a random configuration of "pitch", "yaw" and "roll" operations. These rotationally augmented representations are now passed as an input to an encoder network $f = \mathbf{h}(\mathbf{p}(VIT360(R(X, r))))$, where $\mathbf{p}$ and $\mathbf{h}$ are MLP layers, defined as the projector and predictor head respectively. A projector head ($\mathbf{p}$) appends the MLP Head (shown in Fig. 4.1) with additional MLP layers to create a bottleneck MLP module, which consitutes the final VIT360 architecture. Similarly, a predictor head ($\mathbf{h}$) transforms the output ($p_i \in \mathbb{R}^{n \times EMBED\_DIM}$) from the encoder ($f$) to ($z_i \in \mathbb{R}^{n \times EMBED\_DIM}$) where $i = 1, 2$ represents the first and second stream.

Given output (p_i, z_i) from respective streams, we formulate a pretraining stage to maximize cosine similarity between these representations across siamese streams. The illustration of the maximization process is shown in Eq. (V.3).

$$D(p_1, z_2) = -\frac{p_1}{||p_1||_2} \cdot \frac{z_2}{||z_2||_2}, \quad L_{sim} = \frac{1}{2}D(p_1, z_2) + \frac{1}{2}D(p_2, z_1). \tag{V.3}$$

Here, $p_1 = f(X_1)$ and $z_2 = h(f(X_2))$ denote latent representations computed at different levels for the same input X using rotationally augmented different view (X_1, X_2). Siamese VIT360 is trained so that both projector and predictor layers are subjected to learn similar latent representation of the given input across two different sets of inputs. This representation learning aims to maximize the similarity between these two outputs at different levels across the siamese streams. The maximization problem can be considered in both directions, another being the maximization between (p_2, z_1). Given output pairs (p_1, z_2) and (p_2, z_1), we use the symmetrized similarity loss function (L_{sim}) as shown in Eq. (V.3). As discussed in SimSiam [109], we treat (p_1, p_2) as a constant via stop-grad operations to prevent a

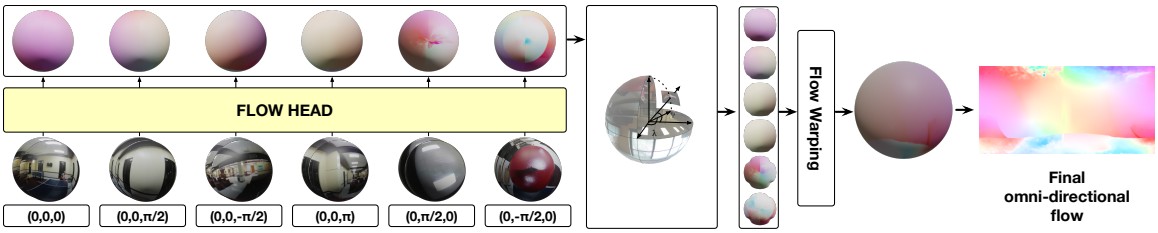


Figure 5.4: **360° flow inference using PercieverIO.** The pairs of consecutive frames are projected with six different field-of-view covering the entire 360° field. The PercieverIO [3] is implemented as Flow Head, which computes the optical flow in each pair of projected field-of-view. The computed flow fields are then projected using tangential projections to extract the central field-of-view. This central field-of-view is then warped in a spherical representation The inference technique avoids pre-crop to avoid the missing of motion features calculation beyond the padding zone.

degenerate solution due to model collapse.

Omnidirectional Aware Optical Flow

The overview of the optical flow inference for 360° videos using PerceiverIO is shown in Fig. 5.4. Similar to VIT360, this approach also considers different projections of pair of input 360° frames for calculating the final flow between the frames. These different projections are considered an input patch to PercieverIO, where it calculates motion information in each pair. Later these motions in patches are warped into one 360° field of view to compute the final 360° optical flow. In order to maintain the computational size, we limit the number of patches to six and perform six different projections to cover the entire 360° field-of-view. Compared to the original PercieverIO implementation, we do not crop these different projections but rather resize these projections into input size as defined in PercieverIO. We do not limit the flow computation across the neighboring patches by avoiding cropping beyond the padding zone. These intermediate flow represents optical flow per projections over entire equirectangular plane. Given optical flow computed in equirectangular plane, flow information in the highly distorted area (like polar

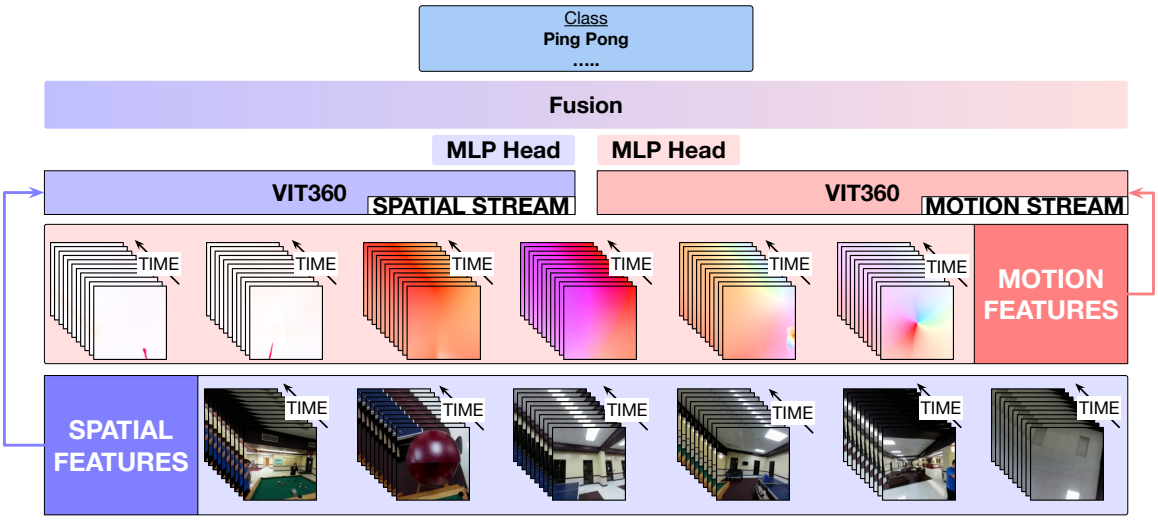


Figure 5.5: **Two stream architecture.** The motion and spatial stream constitute the component of two-stream architecture. The spatial stream receives RGB frames, whereas the motion streams receive the pre-computed optical flow. These two streams are trained independently and later fused to incorporate two different modalities via average and concatenation based late fusion techniques. The fusion techniques include average and concatenation-based methods. The average fusion follows straightforward averaging of logits. The concatenation technique involves the concatenation of fully connected layers and introduction of additional fully connected layer for further training and refinement.

regions) suffers the impact of distortions. However, the central part of equirectangular plane $(\lambda_0, \psi_0) = (0, 0)$ is comparatively least distorted region from where the tangential patches are sampled with $\frac{\pi}{2}$ field-of-view. These sampled patches cover the entire 360° field-of-view. Finally, these patches are warped into global 360° flow for input frames.

Two Stream Architecture

Our egocentric activity recognition framework (shown in Fig. 5.5) is based on recent success on action/activity recognition task [10] loosely based on two stream hypothesis [142]. One stream is designed as a spatial stream responsible for object recognition, and another is designed as a motion stream responsible for motion recognition. Both streams of this framework implement VIT360, as discussed above, for different input modalities (e.g. frames for spatial and optical flow for motion).

These streams are trained independently and later fused (late fusion) to make a joint prediction based on spatial and motion information. Following the best practices in previous work [120], we implement two different fusion techniques - average and concatenation. The average fusion technique computes the average of the scores from the last layer and computes the model confidence, and does not require additional training. The concatenation-based techniques require concatenation of the last MLP head and the introduction of additional fully connected layers matching the output class dimension. This technique requires additional fine-tuning and achieves marginally better results.

Experiment

In this work, we focus our experiments mainly on representation learning based egocentric activity recognition on 360° videos. We run our experiments on EGOK360 [120] dataset, an egocentric activity recognition dataset for 360° videos. We present a series of experiments that demonstrate the efficacy of pre-training VIT360 with a siamese representation approach maximizing the representation across the different view of the same input. In addition, we study the impact of using 360° or omnidirectional aware optical flow compared to traditional perspective videos based on optical flow on 360° videos for motion-based egocentric activity recognition tasks.

The first experiment includes egocentric activity recognition (using a two-stream approach) without the representation learning scheme. Similarly, the follow-up experiments include the representation learning based egocentric activity recognition, which follows two different consecutive experiments of pre-training and two streams approach for classification. Similarly, we conducted two additional experiments based on optical flow and omnidirectional optical flow. In addition to these, we also make a comparative study of qualitative results on proposed

Table 5.1: **Quantitave results.** The Top-1 accuracy (shown as %) of VIT360 compared with baseline methods in EGOK360 [120] achieves consistent performance regardless of projection mode, achieving less than 0.40% accuracy difference. The baseline method shows performance gaps between 5-14%. 360° optical flow computed using our inference techniques boost the peformance of VIT360 by almost 5%. (Note: We use Perspective Flow (RAFT [80]) for baseline experiments.)

Version	Random Projection	Flow	Motion	Spatial	Fused Avg	Fused Concat
ResNet	YES	Perspective	42.53	59.67	56.18	62.79
	NO	Perspective	56.43	68.95	66.79	69.32
I3D	YES	Perspective	48.43	63.61	60.17	65.39
	NO	Perspective	62.19	69.78	67.37	72.68
VIT360 (Ours)	YES	Perspective	56.52	67.14	65.18	70.89
	NO	Perspective	56.57	67.23	65.23	70.79
	YES	360°(Ours)	60.34	71.29	70.66	75.87
	NO	360°(Ours)	**60.43**	**71.13**	**70.29**	**75.59**

PercieverIO-based 360° optical flow inference techniques with pre-trained perspective video based optical flow technique, RAFT [80]. Note that our comparisons based on the two-stream architecture are reported in EGOK360 [120] dataset only. Since the experimental results presented on EGOK360 are based on a random train/test split of entire datasets, we also follow a similar approach. However, to make a fair test case, we sample train and test frames with 9:1 split from each clip from the entire training data. We achieve marginally similar baseline results on EGOK360 compared to experimental results reported in the original paper. In addition, we only consider activity recognition on EGOK360 dataset as experimental results are sufficient to establish core concepts of rotationally invariant egocentric activity recognition.

Experiment Configuration: VIT360 training follows the similar approach we have seen in other related works [9, 10, 89]. We choose Adam [143] as our optimizer with the initial learning rate of 10^{-3}, and StepLR as our learning rate scheduler for smooth training. We consider ten frames as one training input and six tangential patches as input discussed as (T, n) in the method section. The experiments are conducted on three 2080Ti NVIDIA GPUs with a batch size of 16,

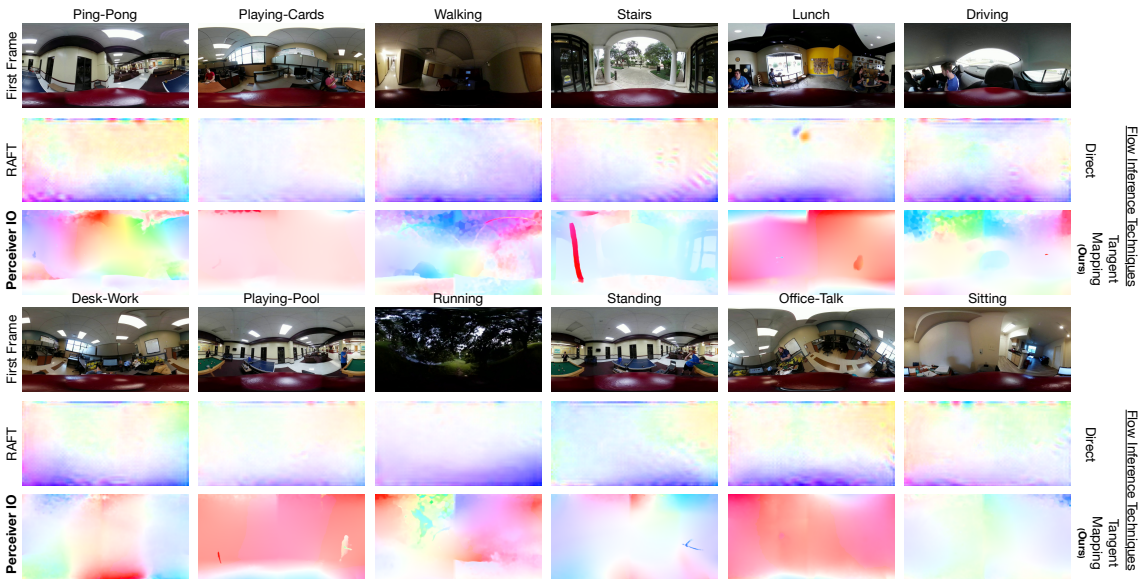


Figure 5.6: **Qualitative results.** We compare 360° flow inferred using our techniques and perspective video-based technique using RAFT [80]. The superiority of our method can be seen in optical flow with smooth motion, distinct motion region, and flow continuity across the edges.

both in the training and testing phase. The training and testing speed approaches 2 seconds/iterations and 1.5 seconds/iterations, respectively, achieving a throughput of nearly 80 and 106 frames per second.

Discussion on Activity Recognition Results: Table 5.1 summarizes our experimental results on egocentric activity recognition on EGOK360 datasets. From these results, we can observe that the baseline method achieves marginally similar accuracy as reported in [120] on a fairly sampled new test set. Though these results are promising for controlled projections, the weaknesses of the baseline algorithm quickly appear when we perform random rotation during testing. Its accuracy drops significantly in both motion and spatial stream. However, considering the VIT360 for a similar effect, we obtain consistent results during fixed and random rotations. These achievements of VIT360 are useful considering the nature of 360° videos in the wild. In addition, we see a significant boost in performance using optical flow features computed using our inference techniques on 360° videos. The experimental

results of VIT360 compared to the baseline on the same test set have shown an impressive performance boost, which provids strong evidences for the efficacy of VIT360 in egocentric activity recognition on 360° videos.

Discussion on Optical Flow Results: Based on results from Table 5.1, the proposed optical flow inference technique is comparatively better than the traditional techniques. To understand this in details, we also present qualitative results of our techniques and compare them with perspective flow-based techniques in Fig. 5.6. The qualitative results show more accurate optical flow for 360° videos using VIT360, which can be further explained by smooth displacement field, motion boundaries, and flow continuity around edges.

Limitation and Societal Impact: The experimental results are based on EGOK360 [120] datasets. The original paper discusses activity recognition as a two-tier task: activity, and action, where activity constitutes a series of more minor actions. In our experiments, we only focus on the activity recognition task. The experimental results on activity recognition support the claim regarding the efficacy of VIT360. However, we do not know the impact of our design on more fine-grained classifications at the action level. Similarly, as progress is made in this domain and 360° videos based activity recognition and framework becomes increasingly pervasive, privacy and security concerns are compounded. Using the video input from wearable devices, these architectures can be used maliciously for mass monitoring raising serious privacy, legal, and ethical concerns. On the other hand, this can also be used to provide prompt response and assistance in case of emergencies.

Conclusion

In this chapter, we propose VIT360 - a vision transformer based network pretrained with siamese representation - to achieve rotational invariance in 360°

videos for egocentric activity recognition. VIT360 performs consistently regardless of the projection mode, achieving less than 0.4% accuracy gap whereas baseline methods drop in performance by 5% to 14% between fixed and random field-of-view projections. Our 360° flow inference technique integrates seamlessly with VIT360 to boost the performance by almost 5%. VIT360 is more suitable for activity recognition in real-world scenarios due to its rotational invariant properties.

VI. CONCLUSIONS

In this dissertation, we have addressed several computer vision problems associated with activity recognition and motion understanding in 360° videos, i.e., lack of activity recognition datasets, challenges in omnidirectional motion estimation, and challenges in activity recognition in 360° videos.

Activity recognition is one of the critical areas of computer vision and has several potential considering its implementation on 360° videos. However, there are several challenges ranging from lack of egocentric activity recognition datasets to 360° domain-specific deep learning-based computer vision framework. We explore the first challenge, the lack of an egocentric activity recognition dataset in Chapter II. We proposed Egok360, A 360° Egocentric Kinetic Human Activity Video Dataset. The Egok360 is the first of its kind in the literature for egocentric activity recognition in 360° videos (EAR360).

Following the progress of the EgoK360, we focused on understanding an essential feature for activity recognition, optical flow for 360° videos or omnidirectional flow. In chapter III we propose LiteFlowNet360, a domain adaptation framework for transforming existing off-the-shelf optical flow models for omnidirectional flow estimation. This framework addresses challenges like radial distortions and mitigating such issues via transforming existing CNNs layers with learnable spherical convolution layers.

Though LiteFlowNet360 shows satisfactory results on the augmented dataset, it imposes two significant challenges. Lack of performance evidence on benchmark datasets and issues regarding the design like difficulty in portability, poor generalization, and over parametrization. In chapter IV we address this issue by proposing a benchmark dataset (FLOW360) and omnidirectional flow estimation framework (SLOF) by exploiting the nature of 360° videos.

Finally, following the success of SLOF in Chapter IV and recent advancements in transformer-based models, we introduced VIT360, a vision transformer-based egocentric activity recognition for 360° videos. VIT360 utilizes a similar principle to SLOF to achieve rotational invariance via siamese representation learning.

In summary, we targeted two significant aspects in 360° video understanding, (i) Activity recognition, and (ii) Motion understanding. The overall design principle of exploiting the rotational invariance properties of 360° videos reveals a crucial aspect in 360° video understanding for future research. The major contributions of this work focuses on representation learning for motion (SLOF) and activity understanding (VIT360) which is really a powerful concepts along with important dataset contributions like EgoK360 and FLOW360 to advance the motion understanding field in 360° videos.

APPENDIX SECTION

APPENDIX A: PUBLICATIONS

This dissertation consists of the following publications:

- Chapter II:
  - **Bhandari, K.**, DeLaGarza, M. A., Zong, Z., Latapie, H., & Yan, Y. (2020, October). Egok360: A 360 Egocentric Kinetic Human Activity Video Dataset. In 2020 IEEE ICIP (pp. 266-270). IEEE.
- Chapter III:
  - **Bhandari, K.**, Zong, Z., & Yan, Y. (2021, January). Revisiting Optical Flow Estimation in 360 Videos. In 2020 25th ICPR (pp. 8196-8203). IEEE.

The following papers have been accepted and are in the process for publication at the moment of writing this Ph.D. thesis:

- Chapter IV:
  - **Bhandari, K.**, Duan, B., Liu, G., Latapie, H., Zong, Z., & Yan, Y. Learning Omnidirectional Flow in 360° Video via Siamese Representation. In ECCV 2022.

The following papers have been sumbitted and are under review at the moment of writing this Ph.D. thesis:

- Chapter V:
  - **Bhandari, K.**, Aryal, B., Duan, B., Ngu Anne, HH., Zong, Z., & Yan, Y. VIT360: Egocentric Activity Recognition via Siamese Representation Learning in 360° Videos. In NeurIPS 2022.

REFERENCES

[1] https://microsites.lomography.com, "Spinner 360°, history." https://microsites.lomography.com/spinner-360/history/.

[2] D. J. Butler, J. Wulff, G. B. Stanley, and M. J. Black, "A naturalistic open source movie for optical flow evaluation," in *ECCV*, 2012.

[3] A. Jaegle, S. Borgeaud, J.-B. Alayrac, C. Doersch, C. Ionescu, D. Ding, S. Koppula, D. Zoran, A. Brock, E. Shelhamer, *et al.*, "Perceiver io: A general architecture for structured inputs & outputs," *arXiv preprint arXiv:2107.14795*, 2021.

[4] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[5] S. Hecker, D. Dai, and L. Van Gool, "End-to-end learning of driving models with surround-view cameras and route planners," in *ECCV*, September 2018.

[6] C. Häne, L. Heng, G. Lee, F. Fraundorfer, P. Furgale, T. Sattler, and M. Pollefeys, "3d visual perception for self-driving cars using a multi-camera system: Calibration, mapping, localization, and obstacle detection," *IVC*, vol. 68, pp. 14–27, 2017.

[7] K. Soomro, A. Roshan Zamir, and M. Shah, "UCF101: A Dataset of 101 Human Actions Classes From Videos in The Wild," *arXiv e-prints*, p. arXiv:1212.0402, Dec 2012.

[8] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre, "Hmdb: a large video database for human motion recognition," in *ICCV*, pp. 2556–2563, IEEE, 2011.

[9] J. Carreira and A. Zisserman, "Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset," *arXiv e-prints*, p. arXiv:1705.07750, May 2017.

[10] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," *NeurIPS*, vol. 27, 2014.

[11] W. Kay, J. Carreira, K. Simonyan, B. Zhang, C. Hillier, S. Vijayanarasimhan, F. Viola, T. Green, T. Back, P. Natsev, M. Suleyman, and A. Zisserman, "The Kinetics Human Action Video Dataset," *arXiv e-prints*, p. arXiv:1705.06950, May 2017.

[12] S. Singh, C. Arora, and C. V. Jawahar, "Generic action recognition from egocentric videos," in *NCVPRIPG*, pp. 1–4, Dec 2015.

[13] S. Singh, C. Arora, and C. Jawahar, "First person action recognition using deep learned descriptors," in *CVPR*, pp. 2620–2628, 2016.

[14] L. s, I. Gori, J. K. Aggarwal, and M. S. Ryoo, "Robot-centric activity recognition from first-person rgb-d videos," in *WACV*, pp. 357–364, IEEE, 2015.

[15] Y. J. Lee, J. Ghosh, and K. Grauman, "Discovering important people and objects for egocentric video summarization," in *CVPR*, pp. 1346–1353, June 2012.

[16] Y. Zhu, G. Zhai, and X. Min, "The prediction of head and eye movement for 360 degree images," *Signal Processing: Image Communication*, vol. 69, pp. 15–25, 2018.

[17] Y. Rai, J. Gutiérrez, and P. Le Callet, "A dataset of head and eye movements for 360 degree images," in *Proceedings of the 8th ACM on Multimedia Systems Conference*, pp. 205–210, ACM, 2017.

[18] W. Lo, C. Fan, J. Lee, C. Huang, K. Chen, and C. Hsu, "360 video viewing dataset in head-mounted virtual reality," in *Proceedings of the 8th ACM on Multimedia Systems Conference*, pp. 211–216, ACM, 2017.

[19] H. Hu, Y. Lin, M. Liu, H. Cheng, Y. Chang, and M. Sun, "Deep 360 pilot: Learning a deep agent for piloting through 360 sports videos," in *CVPR*, pp. 1396–1405, IEEE, 2017.

[20] C. Wu, Z. Tan, Z. Wang, and S. Yang, "A dataset for exploring user behaviors in vr spherical video streaming," in *Proceedings of the 8th ACM on Multimedia Systems Conference*, pp. 193–198, ACM, 2017.

[21] H.-T. Cheng, C.-H. Chao, J.-D. Dong, H.-K. Wen, T.-L. Liu, and M. Sun, "Cube padding for weakly-supervised saliency prediction in 360 videos," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[22] E. J. David, J. Gutiérrez, A. Coutrot, M. P. Da Silva, and P. L. Callet, "A dataset of head and eye movements for 360 videos," in *Proceedings of the 9th ACM Multimedia Systems Conference*, pp. 432–437, 2018.

[23] D. Damen, H. Doughty, G. Maria Farinella, S. Fidler, A. Furnari, E. Kazakos, D. Moltisanti, J. Munro, T. Perrett, W. Price, and M. Wray, "Scaling egocentric vision: The epic-kitchens dataset," in *ECCV*, 2018.

[24] S. Bambach, S. Lee, D. J. Crandall, and C. Yu, "Lending a hand: Detecting hands and recognizing activities in complex egocentric interactions," in *ICCV*, 2015.

[25] Y. Li, M. Liu, and J. M. Rehg, "In the eye of beholder: Joint learning of gaze and actions in first person video," in *ECCV*, 2018.

[26] G. A. Sigurdsson, A. Gupta, C. Schmid, A. Farhadi, and K. Alahari, "Charades-ego: A large-scale dataset of paired third and first person videos," *arXiv preprint arXiv:1804.09626*, 2018.

[27] H. Pirsiavash and D. Ramanan, "Detecting activities of daily living in first-person camera views," in *CVPR*, 2012.

[28] K. Singh, K. Fatahalian, and A. Efros, "Krishnacam: Using a longitudinal, single-person, egocentric dataset for scene understanding tasks," in *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 1–9, IEEE, 2016.

[29] S. Song, V. Chandrasekhar, N. Cheung, S. Narayan, L. Li, and J. Lim, "Activity recognition in egocentric life-logging videos," in *ACCV*, pp. 445–458, Springer, 2014.

[30] Y. Lee, J. Ghosh, and K. Grauman, "Discovering important people and objects for egocentric video summarization," in *CVPR*, pp. 1346–1353, IEEE, 2012.

[31] Z. Lu and K. Grauman, "Story-driven summarization for egocentric video," in *CVPR*, pp. 2714–2721, 2013.

[32] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox, "Flownet 2.0: Evolution of optical flow estimation with deep networks," in *CVPR*, 2017.

[33] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *CVPR*, 2016.

[34] M. Luo, X. Chang, Z. Li, L. Nie, A. G. Hauptmann, and Q. Zheng, "Simple to Complex Cross-modal Learning to Rank," *arXiv e-prints*, p. arXiv:1702.01229, Feb 2017.

[35] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Learning Deep Features for Discriminative Localization," *arXiv e-prints*, p. arXiv:1512.04150, Dec 2015.

[36] Y.-C. Su and K. Grauman, "Kernel transformer networks for compact spherical convolution," in *CVPR*, 2019.

[37] Y.-C. Su and K. Grauman, "Learning spherical convolution for fast features from 360°imagery," in *NeurIPS*, 2017.

[38] Z. Zhang, Y. Xu, J. Yu, and S. Gao, "Saliency detection in 360 videos," in *ECCV*, 2018.

[39] A. Nguyen, Z. Yan, and K. Nahrstedt, "Your attention is unique: Detecting 360-degree video saliency in head-mounted display for head movement prediction," in *Proceedings of the 26th ACM international conference on Multimedia*, pp. 1190–1198, 2018.

[40] H.-N. Hu, Y.-C. Lin, M.-Y. Liu, H.-T. Cheng, Y.-J. Chang, and M. Sun, "Deep 360 pilot: Learning a deep agent for piloting through 360 sports videos," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1396–1405, IEEE, 2017.

[41] Y. Xu, Y. Dong, J. Wu, Z. Sun, Z. Shi, J. Yu, and S. Gao, "Gaze prediction in dynamic 360 immersive videos," in *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5333–5342, 2018.

[42] B. Horn and B. Schunck, "Techniques and applications of image understanding," 1981.

[43] B. D. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *IJCAI*, vol. 2, 1981.

[44] 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 *ICCV*, 2015.

[45] T.-W. Hui, X. Tang, and C. C. Loy, "Liteflownet: A lightweight convolutional neural network for optical flow estimation," in *CVPR*, 2018.

[46] T. S. Cohen, M. Geiger, J. Koehler, and M. Welling, "Spherical cnns," *arXiv*, 2018.

[47] C. Esteves, C. Allen-Blanchette, A. Makadia, and K. Daniilidis, "Learning so (3) equivariant representations with spherical cnns," in *ECCV*, 2018.

[48] B. Coors, A. P. Condurache, and A. Geiger, "Spherenet: Learning spherical representations for detection and classification in omnidirectional images," in *ECCV*, 2018.

[49] Y. Bengio, J. Louradour, R. Collobert, and J. Weston, "Curriculum learning," in *Proceedings of the 26th Annual International Conference on Machine Learning*, 2009.

[50] P. Liu, M. Lyu, I. King, and J. Xu, "Selflow: Self-supervised learning of optical flow," in *CVPR*, 2019.

[51] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Deep end2end voxel2voxel prediction," in *CVPRW*, 2016.

[52] A. Ahmadi and I. Patras, "Unsupervised convolutional neural networks for motion estimation," in *ICIP*, 2016.

[53] J. Wulff and M. J. Black, "Efficient sparse-to-dense optical flow estimation using a learned basis and layers," in *CVPR*, 2015.

[54] J. Y. Jason, A. W. Harley, and K. G. Derpanis, "Back to basics: Unsupervised learning of optical flow via brightness constancy and motion smoothness," in *ECCV*, 2016.

[55] D. Teney and M. Hebert, "Learning to extract motion from videos in convolutional neural networks," in *ACCV*, 2016.

[56] A. Ranjan and M. J. Black, "Optical flow estimation using a spatial pyramid network," in *CVPR*, 2017.

[57] D. Gadot and L. Wolf, "PatchBatch: a Batch Augmented Loss for Optical Flow," *arXiv e-prints*, 2015.

[58] C. Bailer, K. Varanasi, and D. Stricker, "Cnn-based patch matching for optical flow with thresholded hinge embedding loss," in *CVPR*, 2017.

[59] H.-Y. Tung, H.-W. Tung, E. Yumer, and K. Fragkiadaki, "Self-supervised learning of motion capture," in *Advances in Neural Information Processing Systems*, 2017.

[60] H.-N. Hu, Y.-C. Lin, M.-Y. Liu, H.-T. Cheng, Y.-J. Chang, and M. Sun, "Deep 360 Pilot: Learning a Deep Agent for Piloting through 360° Sports Video," *arXiv e-prints*, 2017.

[61] W.-S. Lai, Y. Huang, N. Joshi, C. Buehler, M.-H. Yang, and S. B. Kang, "Semantic-driven Generation of Hyperlapse from 360° Video," *arXiv e-prints*, 2017.

[62] W. Boomsma and J. Frellsen, "Spherical convolutions and their application in molecular modelling," in *NeurIPS*, 2017.

[63] H.-T. Cheng, C.-H. Chao, J.-D. Dong, H.-K. Wen, T.-L. Liu, and M. Sun, "Cube Padding for Weakly-Supervised Saliency Prediction in 360° Videos," *arXiv e-prints*, 2018.

[64] R. Khasanova and P. Frossard, "Graph-Based Classification of Omnidirectional Images," *arXiv e-prints*, 2017.

[65] O. Shakernia, R. Vidal, and S. Sastry, "Omnidirectional egomotion estimation from back-projection flow," in *CVPRW*, 2003.

[66] A. Radgui, C. Demonceaux, E. M. Mouaddib, D. Aboutajdine, and M. Rziza, "An adapted lucas-kanade's method for optical flow estimation in catadioptric images," in *The 8th Workshop on Omnidirectional Vision, Camera Networks and Non-classical Cameras-OMNIVIS*, 2008.

[67] A. Radgui, C. Demonceaux, E. Mouaddib, M. Rziza, and D. Aboutajdine, "Optical flow estimation from multichannel spherical image decomposition," *Computer Vision and Image Understanding*, 2011.

[68] C. Kirisits, L. F. Lang, and O. Scherzer, "Decomposition of optical flow on the sphere," *GEM-International Journal on Geomathematics*, 2014.

[69] B. Alibouch, A. Radgui, M. Rziza, and D. Aboutajdine, "Optical flow estimation on omnidirectional images: an adapted phase based method," in *International Conference on Image and Signal Processing*, Springer, 2012.

[70] M. Menze, C. Heipke, and A. Geiger, "Discrete optimization for optical flow," in *GCPR*, 2015.

[71] C. Bailer, B. Taetz, and D. Stricker, "Flow fields: Dense correspondence fields for highly accurate large displacement optical flow estimation," in *ICCV*, 2015.

[72] Q. Chen and V. Koltun, "Full flow: Optical flow estimation by global optimization over regular grids," in *CVPR*, 2016.

[73] B. K. Horn and B. G. Schunck, "Determining optical flow," *Artificial intelligence*, vol. 17, no. 1-3, pp. 185–203, 1981.

[74] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert, "High accuracy optical flow estimation based on a theory for warping," in *ECCV*, 2004.

[75] F. Steinbrücker, T. Pock, and D. Cremers, "Large displacement optical flow computation withoutwarping," in *ICCV*, 2009.

[76] T. Brox and J. Malik, "Large displacement optical flow: descriptor matching in variational motion estimation," *TPAMI*, vol. 33, no. 3, pp. 500–513, 2010.

[77] R. Garg, A. Roussos, and L. Agapito, "A variational approach to video registration with subspace constraints," *IJCV*, vol. 104, no. 3, pp. 286–314, 2013.

[78] P. Weinzaepfel, J. Revaud, Z. Harchaoui, and C. Schmid, "Deepflow: Large displacement optical flow with deep matching," in *ICCV*, 2013.

[79] S. Jiang, D. Campbell, Y. Lu, H. Li, and R. Hartley, "Learning to estimate hidden motions with global motion aggregation," *arXiv*, 2021.

[80] Z. Teed and J. Deng, "Raft: Recurrent all-pairs field transforms for optical flow," in *ECCV*, 2020.

[81] S. Zhao, Y. Sheng, Y. Dong, E. I. Chang, Y. Xu, *et al.*, "Maskflownet: Asymmetric feature matching with learnable occlusion mask," in *CVPR*, 2020.

[82] D. Sun, X. Yang, M.-Y. Liu, and J. Kautz, "Models matter, so does training: An empirical study of cnns for optical flow estimation," *TPAMI*, vol. 42, no. 6, pp. 1408–1423, 2019.

[83] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *IJRR*, vol. 32, no. 11, pp. 1231–1237, 2013.

[84] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? the kitti vision benchmark suite," in *CVPR*, 2012.

[85] M. Menze and A. Geiger, "Object scene flow for autonomous vehicles," in *CVPR*, 2015.

[86] S. Baker, S. Roth, D. Scharstein, M. J. Black, J. Lewis, and R. Szeliski, "A database and evaluation methodology for optical flow," in *ICCV*, 2007.

[87] R. Azevedo, N. Birkbeck, F. Simone, I. Janatra, B. Adsumilli, and P. Frossard, "Visual distortions in 360-degree videos," *TCSVT*, vol. 2019, no. 8, pp. 2524–2537, 2020.

[88] M. Eder, M. Shvets, J. Lim, and J.-M. Frahm, "Tangent images for mitigating spherical distortion," in *CVPR*, 2020.

[89] K. Bhandari, Z. Zong, and Y. Yan, "Revisiting optical flow estimation in 360 videos," in *ICPR*, 2021.

[90] R. Seidel, A. Apitzsch, and G. Hirtz, "Omniflow: Human omnidirectional optical flow," in *CVPR*, 2021.

[91] C.-O. Artizzu, H. Zhang, G. Allibert, and C. Demonceaux, "Omniflownet: a perspective neural network adaptation for optical flow estimation in omnidirectional images," in *ICPR*, 2021.

[92] S. Meister, J. Hur, and S. Roth, "Unflow: Unsupervised learning of optical flow with a bidirectional census loss," in *AAAI*, 2018.

[93] J. Hur and S. Roth, "Mirrorflow: Exploiting symmetries in joint optical flow and occlusion estimation," in *ICCV*, 2017.

[94] N. Zioulis, A. Karakottas, D. Zarpalas, and P. Daras, "Omnidepth: Dense depth estimation for indoors spherical panoramas," in *ECCV*, 2018.

[95] B. Y. Feng, W. Yao, Z. Liu, and A. Varshney, "Deep depth estimation on 360° images with a double quaternion loss," in *3DV*, 2020.

[96] R. Wang, D. Geraghty, K. Matzen, R. Szeliski, and J.-M. Frahm, "Vplnet: Deep single view normal estimation with vanishing points and lines," in *CVPR*, 2020.

[97] J. L. Barron, D. J. Fleet, and S. S. Beauchemin, "Performance of optical flow techniques," *IJCV*, vol. 12, no. 1, pp. 43–77, 1994.

[98] B. McCane, K. Novins, D. Crannitch, and B. Galvin, "On benchmarking optical flow," *CVIU*, vol. 84, no. 1, 2001.

[99] M. Otte and H.-H. Nagel, "Optical flow estimation: advances and comparisons," in *ECCV*, 1994.

[100] C. Liu, W. T. Freeman, E. H. Adelson, and Y. Weiss, "Human-assisted motion annotation," in *CVPR*, 2008.

[101] D. J. Field, "Relations between the statistics of natural images and the response properties of cortical cells," *Josa a*, vol. 4, no. 12, pp. 2379–2394, 1987.

[102] S. Meister, B. Jähne, and D. Kondermann, "Outdoor stereo camera system for the generation of real-world benchmark data sets," *Optical Engineering*, vol. 51, no. 2, p. 021107, 2012.

[103] C. Demonceaux and D. Kachi-Akkouche, "Optical flow estimation in omnidirectional images using wavelet approach," in *CVPRW*, 2003.

[104] T.-W. Hui, X. Tang, and C. C. Loy, "A lightweight optical flow cnn -revisiting data fidelity and regularization," *TPAMI*, vol. 43, no. 8, pp. 2555–2569, 2021.

[105] C. Fernandez-Labrador, J. M. Facil, A. Perez-Yus, C. Demonceaux, J. Civera, and J. J. Guerrero, "Corners for layout: End-to-end layout recovery from 360 images," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1255–1262, 2020.

[106] Y. Taigman, M. Yang, M. Ranzato, and L. Wolf, "Deepface: Closing the gap to human-level performance in face verification," in *CVPR*, 2014.

[107] J. Bromley, J. W. Bentz, L. Bottou, I. Guyon, Y. LeCun, C. Moore, E. Säckinger, and R. Shah, "Signature verification using a "siamese" time delay neural network," *IJPRAI*, vol. 7, no. 04, pp. 669–688, 1993.

[108] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. Torr, "Fully-convolutional siamese networks for object tracking," in *ECCV*, 2016.

[109] X. Chen and K. He, "Exploring simple siamese representation learning," in *CVPR*, 2021.

[110] Blender. https://www.blender.org/.

[111] A. Goralczyk, "Nishita sky demo," 2020. Creative Commons CC0 (Public Domain) - Blender Studio - cloud.blender.org.

[112] M. Wolinski, "City - 3d model." sketchfab.com.

[113] S. V. Hulle, "Bcon19," 2019. 2019 Blender Conference - cloud.blender.org.

[114] Turbosquid. https://www.turbosquid.com.

[115] Sketchfab. https://sketchfab.com/.

[116] Adobe, "Mixamo." https://www.mixamo.com/.

[117] E. P. Simoncelli and B. A. Olshausen, "Natural image statistics and neural representation," *Annual review of neuroscience*, vol. 24, no. 1, pp. 1193–1216, 2001.

[118] C. Geyer and K. Daniilidis, "A unifying theory for central panoramic systems and practical implications," in *ECCV*, 2000.

[119] A. Núñez-Marcos, G. Azkune, and I. Arganda-Carreras, "Egocentric vision-based action recognition: A survey," *Neurocomputing*, vol. 472, pp. 175–197, 2022.

[120] K. Bhandari, M. A. DeLaGarza, Z. Zong, H. Latapie, and Y. Yan, "Egok360: A 360 egocentric kinetic human activity video dataset," in *ICIP*, 2020.

[121] T. Ojala, M. Pietikainen, and D. Harwood, "Performance evaluation of texture measures with classification based on kullback discrimination of distributions," in *ICPR*, 1994.

[122] T. Ojala, M. Pietikäinen, and D. Harwood, "A comparative study of texture measures with classification based on featured distributions," *PR*, vol. 29, no. 1, pp. 51–59, 1996.

[123] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *IJCB*, vol. 60, no. 2, pp. 91–110, 2004.

[124] H. Bay, T. Tuytelaars, and L. V. Gool, "Surf: Speeded up robust features," in *ECCV*, 2006.

[125] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-scale video classification with convolutional neural networks," in *CVPR*, 2014.

[126] J. Donahue, L. Anne Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell, "Long-term recurrent convolutional networks for visual recognition and description," in *CVPR*, 2015.

[127] Z. Li, K. Gavrilyuk, E. Gavves, M. Jain, and C. G. Snoek, "Videolstm convolves, attends and flows for action recognition," *CVIU*, vol. 166, pp. 41–50, 2018.

[128] J. Y.-H. Ng, M. J. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici, "Beyond short snippets: Deep networks for video classification. corr abs/1503.08909 (2015)," 2015.

[129] S. Sharma, R. Kiros, and R. Salakhutdinov, "Action recognition using visual attention," *arXiv preprint arXiv:1511.04119*, 2015.

[130] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3d convolutional networks," in *ICCV*, 2015.

[131] N. Hussein, E. Gavves, and A. W. Smeulders, "Timeception for complex action recognition," in *CVPR*, 2019.

[132] X. Wang and A. Gupta, "Videos as space-time region graphs," in *ECCV*, 2018.

[133] Y.-C. Su and K. Grauman, "Learning spherical convolution for 360 recognition," *TPAMI*, 2021.

[134] D. Bahdanau, K. Cho, *et al.*, "Neural machine translation by jointly learning to align and translate. arxiv preprint arxiv: 1409.0473," *arXiv preprint*, 2014.

[135] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *NeurIPS*, 2017.

[136] N. Parmar, A. Vaswani, J. Uszkoreit, L. Kaiser, N. Shazeer, A. Ku, and D. Tran, "Image transformer," in *ICML*, 2018.

[137] K. Gavrilyuk, R. Sanford, M. Javan, and C. G. Snoek, "Actor-transformers for group activity recognition," in *CVPR*, 2020.

[138] L. Yang, Y. Huang, Y. Sugano, and Y. Sato, "Stacked temporal attention: Improving first-person action recognition by emphasizing discriminative clips," *arXiv preprint arXiv:2112.01038*, 2021.

[139] X. Li, Y. Hou, P. Wang, Z. Gao, M. Xu, and W. Li, "Trear: Transformer-based rgb-d egocentric action recognition," *TCDS*, 2021.

[140] A. Bulat, J. M. Perez Rua, S. Sudhakaran, B. Martinez, and G. Tzimiropoulos, "Space-time mixing attention for video transformer," in *NeurIPS*, 2021.

[141] R. Girdhar, J. Carreira, C. Doersch, and A. Zisserman, "Video action transformer network," in *CVPR*, 2019.

[142] M. A. Goodale and A. D. Milner, "Separate visual pathways for perception and action," *Trends in neurosciences*, vol. 15, no. 1, pp. 20–25, 1992.

[143] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.