PoseFlow: A Deep Motion Representation for Understanding Human Behaviors in Videos

Dingwen Zhang[1,2†], Guangyu Guo[1†], Dong Huang[2†], Junwei Han[1*]
[1]Northwestern Polytechincal University, [2]Carnegie Mellon University
zdw2006yyy@mail.nwpu.edu.cn, gyguo95@gmail.com
dghuang@andrew.cmu.edu, junweihan2010@gmail.com

Abstract

Motion of the human body is the critical cue for understanding and characterizing human behavior in videos. Most existing approaches explore the motion cue using optical flows. However, optical flow usually contains motion on both the interested human bodies and the undesired background. This "noisy" motion representation makes it very challenging for pose estimation and action recognition in real scenarios. To address this issue, this paper presents a novel deep motion representation, called PoseFlow, which reveals human motion in videos while suppressing background and motion blur, and being robust to occlusion. For learning PoseFlow with mild computational cost, we propose a functionally structured spatial-temporal deep network, PoseFlow Net (PFN), to jointly solve the skeleton localization and matching problems of PoseFlow. Comprehensive experiments show that PFN outperforms the state-of-the-art deep flow estimation models in generating PoseFlow. Moreover, PoseFlow demonstrates its potential on improving two challenging tasks in human video analysis: pose estimation and action recognition.

1. Introduction

Understanding and characterizing human behavior in video have attracted tremendous research interests due to its potential to revolutionize daily life, health care and public security. In particular, applications such as video surveillance, Telerehabilitation, social robotics and autonomous driving, require automatically detecting and recognizing human body motion from cluttered background. To meet these demands, researchers have made great efforts in the last few years on pose estimation [3], tracking [6], human parsing [23], action recognition [18], person re-identification [22, 4]. In all these tasks, learning effective representation of the body motion plays a critical role for improving detection and recognition of human behavior in videos.

*Corresponding author. †The first three authors contribute equally.

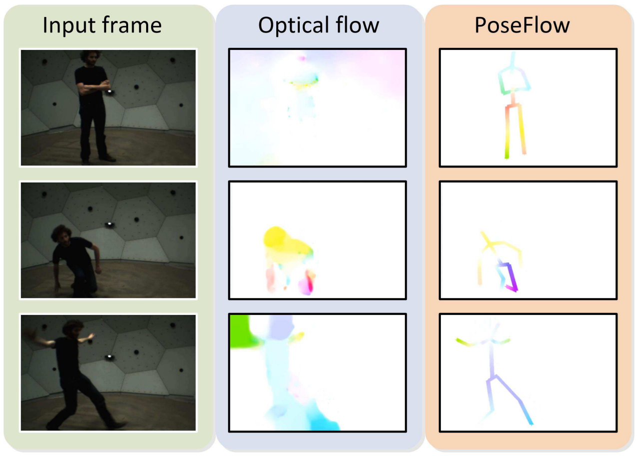

Figure 1. Optical flow and PoseFlow (our) computed on video frames. Pixels in the Optical flow and PoseFlow images are visualized as color-coded vector fields (**Best viewed in color**). PoseFlow reveals only human body motion in videos from the background motion (the 1st row), body motion blur (the 2nd row) and occlusion (the 3rd row).

One of the most widely used motion representations in video is optical flow [2]. Optical flow consists of the displacement vectors of correspondent pixels between video frames. Most video analysis tasks require precomputed optical flows. Without using the pose and action labels that correspond to human body motion, the displacement vectors of optical flow are produced not only on the interested human bodies but also on the undesired background (See the middle column in Fig. 1). Given cluttered background, body motion blur and occlusion in both video frames and optical flows, it is very challenging for latter algorithms to estimate body poses and recognize actions. Moreover, traditional methods to compute optical flow would become inaccurate especially when dealing with multi-scales and large displacement.

The flourishing advance of deep neural networks [24, 30, 25, 14] leads to faster and more accurate solutions for optical flow estimation. Among these approaches, the early effort [24] mainly focuses on end-to-end mapping with deep but simple network architectures. Unfortunately, learning without function-aware structures is inefficient to train, blindness to tune, and requires huge training data and GPUs. As one step towards structured flow learning, [25, 14] outperform previous approaches by explicitly performing pixel-wise matching: warping the former frame to the latter frame using the predicted flow field. However, such efforts are still insufficient for obtaining satisfactory flow estimation performance. As we know, network interpretation [8, 1] has become one of the emerging hot research fields in recent years. Instead of interpreting what has been learnt by the established DNNs, an alternative way for ease of understanding DNNs is to build them with the task-specific function-aware structures. When it comes to very complex learning tasks like the investigated PoseFlow estimation, the need for function-aware structures become more pressing.

To address the aforementioned issues, we present PoseFlow – a computationally efficient while highly informative motion representation of human body (see the right column of Fig. 1). Different from conventional optical flow, PoseFlow reveals human motion while suppressing other motion in the background. More specifically, our work leads to four main advantages over existing approaches: **(1)** Poseflow only computes motion on human body. It reflects the intrinsic body motion and can avoid external factors like person-specific body shape, illumination and cloth texture. **(2)** Flow fields on Poseflow are sparser. They have lower costs to store and compute, while already encoding enough rich patterns for understanding human behaviors. therefore could potentially reduce the computational cost (for the subsequent processes) and storage space comparing to conventional optical flow. **(3)** As an explicit representation of body motion, PoseFlow can be used to improve multiple tasks in understanding human behavior in video such as pose estimation and action recognition. Moreover **(4)** PoseFolw is learned using a functional-structured network: PoseFlow Net (PFN) (see Fig. 2). PFN consists of subnetworks explicitly modeling the spatial, temporal, and motion vector mapping functions towards body motion estimation. This design makes it possible to learn an informative PoseFlow representation without resorting to heavy neural networks.

In learning PoseFlow, we need to jointly solve two problems: localization and matching. Specifically, in the localization problem, human body regions (e.g., pixels of body joints and limbs) are extracted from the complex background of each video frame. In the matching problem, the corresponding regions on human bodies are found from the adjacent video frames. Finally, PoseFlow is computed as a pixel-wise vector field on human bodies. Fig. 2 shows PoseFlow Net (PFN), which is an unified end-to-end deep learning framework proposed for learning PoseFlow. The network inputs are two adjacent video frames and the outputs are the flow maps of PoseFlow. The ground-truth PoseFlow maps are generated by connecting annotated body joints into skeletons and computing pixel-wise displacement on skeletons. PFN simultaneously learns to localize the human body skeleton and match the corresponding pixels on skeleton between adjacent frames (see Fig. 2). As shown in Fig. 2, PFN is composed of multiple encoder-decoder branches, each of which corresponds to a computational component derived from the classic flow field estimation model [2]. Comparing to the classic flow field estimation, PFN learns optimal spatial and temporal filters to achieve highly nonlinear mapping from images to flow fields. Comparing to the deep neural networks for optical flow [9, 25, 14], PoseFlow Net (PFN) is explicitly structured with functional objectives. This enables PFN to advance the performance of high-level semantic problem, e.g., pose estimation and action recognition, with smaller model size and lower computational cost than using some previous deep models.

To sum up, this paper presents three-fold contributions:

- PoseFlow–a novel deep motion representation that captures human body motion in video while suppressing background and motion blur, and being robust to occlusion.
- A functionally structured deep neural network, called PoseFlow Net (PFN), which explicitly models spatial and temporal mapping functions for estimating PoseFlow. Comprehensive experiments demonstrate effectiveness and efficiency of PFN as compared with other state-of-the-art flow estimation networks.
- Quantitative and qualitative experiments in application study also demonstrate the benefits of using PoseFlow on pose estimation and action recognition in videos.

2. Related Works

2.1. Flow Estimation

A number of DNN-based approaches estimate motion in video as optical flow. Fischer et al. [9] proposed the FlowNet, which directly predicts optical flow given two input frames using either a cascade or a parallel network. In training, FlowNet was supervised by ground-truth optical flow. After several convolutional and deconvolutional layers, FlowNet predicts multi-channel flow field representations that are used to form the optical flow maps. The channel number depends on a pre-defined maximum displacement parameter. This is a pioneer work towards end-to-end learning of optical flow directly using a DNN model. Tran et al. [30] presented a deep 3D convolutional architecture that models the flow estimation problem as a "Voxel2Voxel" prediction problem. Teney and Hebert [28] developed a rotationally invariant network architecture from signal processing perspectives, which takes raw pixels as input and produced features representing evidence for motion at various speeds and orientations. Ranjan and Black [25] combined the spatial-pyramid formation with DNN for estimating large displacement in a coarse-to-fine approach. More recently, Ilg et al. [14] improved FlowNet in both quality and speed. They handled large displacement by warping the second input frame with the intermediate optical flow before stacked with the first frame and handled the small displacement using an standard FlowNet sub-network. Note this small step towards implicit temporal matching outperforms the "structureless" FlowNet. The warping technique in [14] relies on flow field estimated over the entire image to achieve implicit temporal matching, therefore, does not apply to the body-only flow field in PoseFlow.

All these DNN architectures were designed to overcome intuitive factors in flow estimation such as resolution, small or large displacement. In contrast, our PFN is designed by functional structures of flow estimation [2]. PFN explicitly models the temporal derivative reasoning, spacial derivative reasoning, and motion vector reasoning sub-network branches to help learning the computational components of flow estimation.

2.2. Pose Estimation in Videos

Pose estimation aims to localize a set of human body joints in visual scene. Most of the previous works were focused on still images. Recently several attempts have been made to use temporal information in videos [34, 33, 24, 13, 27]. Zuffi et al. [34] used Flowing puppets, an articulate body part model, to estimate body poses. It is done by matching Flowing puppets with pre-computed optical flows between frames. In contrast, PoseFlow estimates flow fields on body skeletons and is a high-level representation built for generally benefiting video-based human understanding tasks like pose estimation and action recognition. Two of the most recent achievements, Pfister et al. [24] and Song et al. [27], used a sequential spatial-temporal deep learning architecture. Given consecutive frames from videos, Pfister et al. [24] first estimated preliminary heatmaps of joints in individual frames, then aligned and pooled heatmaps before making the final pose estimation. Note that the heatmaps were aligned using optical flow precomputed using traditional optical flow algorithms. Moreover, the success of [24] relies on the accuracy of the preliminary joint heatmaps. Song et al. [27] improved the joint localization in [24] by an extra inference layer.

In above approaches, estimating human pose in videos requires motion cue of human bodies, while the precomputed optical flow contains background motion. PoseFlow addresses this problem by only estimating the flow on human body. Moreover, using our PFN network, PoseFlow estimation shares the same convolutional features for estimating joint coordinates, making it very efficient to leverage temporal information for pose estimation in video.

2.3. Action Recognition in Videos

Action recognition [15, 26, 32, 12, 7] aims at recognizing pre-defined human behavior categories from video. In the most recent DNN-based action recognition, deep representations of action are usually learned from raw pixel inputs and the precomputed optical flow maps. For example, Simonyan and Zisserman [26] proposed a two-stream DNN architecture: one spatial stream to learn appearance from raw image, and one temporal stream to learn motion from optical flows. Following this work, Feichtenhofer et al. [11] comprehensively studied multiple ways to fuse spatial and temporal subnetworks for the best combination of spatio-temporal information. Besides learning from raw pixels, there are also several approaches that explicitly extract poses before learning representation for action recognition. [15, 19, 31, 29] extended the conventional 2D CNN to the CNNs with 3D spatio-temporal convolutions for improving their invariance to translations both in image plane and time.

In above approaches, the precomputed optical flow is widely used as the network inputs for action recognition. Since the goal of using optical flow is to characterizing human motion, it is not necessary to compute background motion. Our approach replaces the conventional optical flow with the PoseFlow. The obtained action recognition approach works more effectively since PoseFlow encodes salient human motion while suppresses the noisy background motions.

3. PoseFlow

For simplicity of presentation, we assume all video frames are of the same size. The t^{th} frame $\mathbf{I}_t \in \Re^{w \times h \times c}$ is an image of w pixels in width, h pixels in height and c channels. PoseFlow of frame $\mathbf{I}_t$ is a 2D vector field $\mathbf{V}_t \in \Re^{w \times h \times 2}$ (see Fig. 1 for an example of PoseFlow). This 2D flow field $\mathbf{V}_t$ describes the horizontal and vertical body motion from the t^{th} to the $(t+1)^{th}$ frame. $\mathbf{V}_t$ is non-zero **only** at the pixels overlapping with human body skeletons, and is zero at remaining pixels.

The learning problem of PoseFlow $\mathbf{V}$ involves two mutually dependent subtasks: localizing human body skeletons and matching the skeletons between frames. The localization task needs temporal information to overcome ambiguity in single frame, while the matching task requires accurate localization in individual frames. If these subtasks are

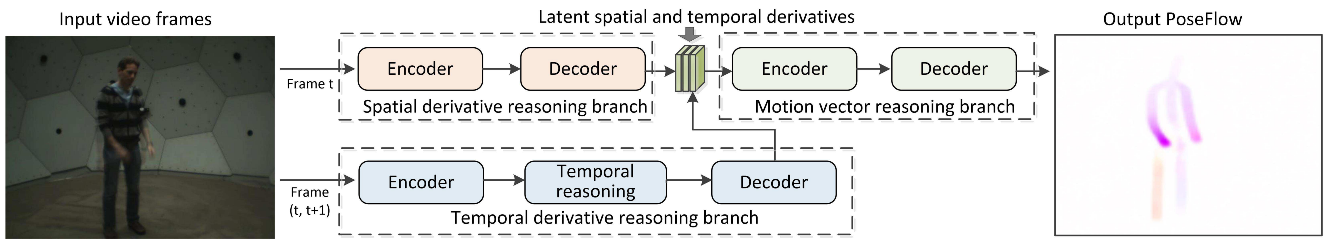
Input video frames
Latent spatial and temporal derivatives
Output PoseFlow
Frame t
Encoder
Decoder
Spatial derivative reasoning branch
Encoder
Decoder
Motion vector reasoning branch
Frame (t, t+1)
Encoder
Temporal reasoning
Decoder
Temporal derivative reasoning branch

Figure 2. PoseFlow net: The network architecture to learn PoseFlow. The output PoseFlow contains motion vectors **only** on human body. The pixels in the output map is color-coded (the same visualization for standard optical flow).

solved independently or sequentially, the learning error in either task will propagate and accumulate to the other.

PFN jointly localizes and matches body parts using two parallel deep neural network branches, i.e., a spatial derivative reasoning branch and a temporal derivative reasoning branch, and a motion vector reasoning branch. This network architecture is developed following the classic optical flow estimation approaches.

3.1. Flow Estimation Revisited

Denote $I(x,y,t)$ is the intensity at pixel (x,y) in the t^{th} frame. The classic flow estimation approach [2] estimates the temporal motion vector of pixels from t to $t+\Delta t$ under the brightness constancy constraint:

$$I(x,y,t) = I(x+\Delta x, y+\Delta y, t+\Delta t). \tag{1}$$

Assuming the movement to be small, the pixel intensity at $(t+\Delta t)$ is estimated by its first order approximation of Taylor series. Then we can get

$$\frac{\partial \mathbf{I}}{\partial x}\Delta x + \frac{\partial \mathbf{I}}{\partial y}\Delta y + \frac{\partial \mathbf{I}}{\partial t}\Delta t = 0. \tag{2}$$

Dividing Δx and Δy with respect to Δt, we obtain motion vector $\mathbf{v}_x = \frac{\Delta x}{\Delta t}$ and $\mathbf{v}_y = \frac{\Delta y}{\Delta t}$.

$$\frac{\partial \mathbf{I}}{\partial x}\mathbf{v}_x + \frac{\partial \mathbf{I}}{\partial y}\mathbf{v}_x + \frac{\partial \mathbf{I}}{\partial t} = 0, \tag{3}$$

which leads to

$$\left[\frac{\partial \mathbf{I}}{\partial x}\frac{\partial \mathbf{I}}{\partial y}\right]^T \begin{bmatrix} \mathbf{v}_x \\ \mathbf{v}_y \end{bmatrix} = -\frac{\partial \mathbf{I}}{\partial t}. \tag{4}$$

Given the spatial derivatives $\left[\frac{\partial \mathbf{I}}{\partial x}, \frac{\partial \mathbf{I}}{\partial y}\right]$ and temporal derivatives $\frac{\partial \mathbf{I}}{\partial t}$, the motion vector $[\mathbf{v}_x, \mathbf{v}_y]$ can be solved by linear programing. Note that in Eq. (4) the spatial and temporal derivatives are usually computed as image gradients as in the traditional flow estimation approaches, which are essentially the convolution of original frames and **fixed** template. The linear equations used in such traditional flow estimation approaches are difficult to model the highly nonlinear relation between non-rigid body and cluttered background. Furthermore, using the traditional image gradients, the calculated spatial and temporal derivatives will mainly appear at the body contours, whereas the more important motions on the body skeletons are hard to capture due to the lack of sufficient textures on them.

To address the aforementioned issues, this paper proposes a novel network architecture, i.e., the PoseFlow Net (PFN), to estimate PoseFlow i.e., the flow on body skeleton. PFN consists of multiple convolutional and deconvolutional network layers that learn the **body skeleton-aware** deep features and desired convolution templates for inferring the latent representations of the spatial and temporal derivatives $\left[\frac{\partial \mathbf{I}}{\partial x}, \frac{\partial \mathbf{I}}{\partial y}\right]$ and $\frac{\partial \mathbf{I}}{\partial t}$. Such representations are then fed to another network branch to estimate the motion vector $[\mathbf{v}_x, \mathbf{v}_y]$. All network parameters are learnt in an end-to-end manner.

The proposed PFN extends the classic flow estimation method to handle nonlinearities: replacing $\left[\frac{\partial \mathbf{I}}{\partial x}\frac{\partial \mathbf{I}}{\partial y}\right]$ with a spatial derivative reasoning branch, replacing $\frac{\partial \mathbf{I}}{\partial t}$ with a temporal derivative reasoning branch and replacing linear transformation in Eq. (4) with a motion vector reasoning branch. Denote $\mathbf{I}_t$ is an image of the t^{th} frame. $I(x,y,t)$ is the intensity at pixel (x,y). We generalize the classic flow estimation function Eq. 4 to convolution networks:

$$f_s(\mathbf{W}_{xy}, \mathbf{I}_t) \cdot \begin{bmatrix} \mathbf{v}_x \\ \mathbf{v}_y \end{bmatrix} = f_t(\mathbf{W}_t, [\mathbf{I}_t, \mathbf{I}_{t+1}]). \tag{5}$$

where $f_s(\cdot)$ denotes the spatial (de)convolution operators, i.e., convolution in image coordinates $\{x,y\}$, $\mathbf{W}_{xy}$ is the learnable parameters in $f_s(\cdot)$; Similarly, $f_t(\cdot)$ denotes the temporal (de)convolution operators, i.e., convolution in time t, and $\mathbf{W}_t$ is the learnable parameters in $f_t(\cdot)$.

3.2. PoseFlow Net

The basic architecture of the proposed PFN is shown in Fig. 2. Observe that PFN takes two adjacent video frames as the input and outputs the 2D PoseFlow maps $\{\mathbf{v}_x, \mathbf{v}_y\}$. Specifically, the first part of the PFN is two parallel network branches, i.e., the spatial derivative reasoning branch and the temporal derivative reasoning branch. The spatial derivative reasoning branch computes the spatial derivative

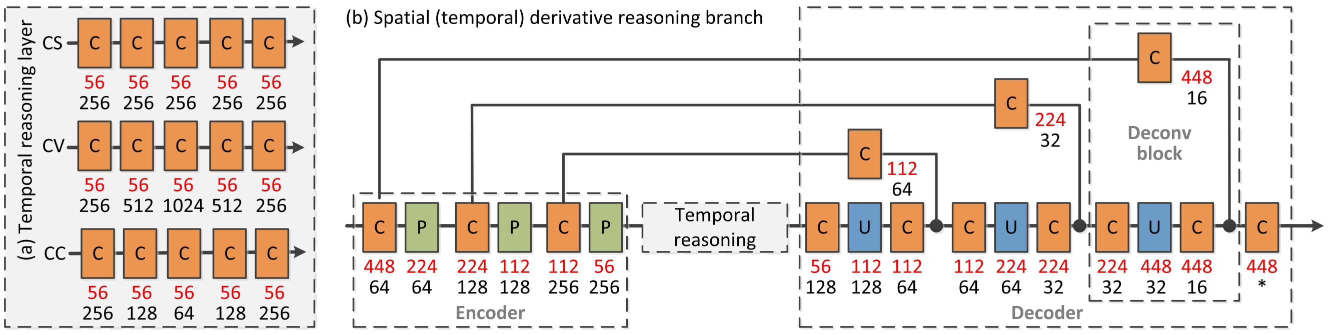

Figure 3. The detailed network architecture of the spatial/temporal derivative reasoning branch of our PFN network. The orange, green, and blue rectangles indicate the convolution, pooling, and up-sampling operations, respectively. The red numbers under each of the operation rectangle indicate the spatial size of the corresponding feature maps, while the black numbers indicate the corresponding channel number. Notice that the spatial derivative reasoning branch does not have the temporal reasoning block. A ReLU nonlinear activation function is used after each layer. The black dots in each deconv block indicates the concatenation operation of the feature maps.

on the human body skeleton in the t^{th} frame and outputs the estimated horizontal derivative representation $\frac{\partial \mathbf{I}}{\partial x}$ and vertical derivative representation $\frac{\partial \mathbf{I}}{\partial y}$. The corresponding network parameters are denoted as $\mathbf{W}_{xy}$. The temporal derivative reasoning branch computes the temporal derivative representation $\frac{\partial \mathbf{I}}{\partial t}$ on the human body skeleton from the t^{th} and $t+1^{th}$ frame, with network parameters $\mathbf{W}_t$. Afterwards, the derivative maps $\mathbf{D}_x, \mathbf{D}_y, \mathbf{D}_t$ formed by the estimated $\frac{\partial \mathbf{I}}{\partial x}$, $\frac{\partial \mathbf{I}}{\partial y}$, and $\frac{\partial \mathbf{I}}{\partial t}$ are combined together as the input of the motion vector reasoning branch with parameters $\mathbf{W}_m$. The motion vector reasoning branch models the mapping functions to produce the PoseFlow motion vector $\{\mathbf{v}_x, \mathbf{v}_y\}$.

Spatial Derivative Reasoning Branch: As shown in Fig. 2, the spatial derivative reasoning branch is an encoder-decoder network architecture with input as the t^{th} video frame. Fig. 3 (b) has displayed more concrete architecture of the spatial derivative reasoning branch. Specifically, the encoder network consists of six layers, i.e., three convolutional layers and three pooling layers. Then, the obtained feature map (with spatial size of 56 and channel number of 256) is fed into the decoder network, which contains three deconv blocks. As shown in Fig. 3 (b), each of the deconv block has three convolutional layers and an up-sampling layer, enabling the spatial size of the inferred feature maps to increase gradually. After the last deconv block, the network uses a convolutional layer to generate a two-channel estimation map with spatial size of 448, which corresponds to the desired horizontal derivative map $\mathbf{D}_x$ and vertical derivative map $\mathbf{D}_y$.

Temporal Derivative Reasoning Branch: As shown in Fig. 2, the temporal derivative reasoning branch is an encoder-temporal reasoning-decoder network architecture with input of the t^{th} and $t+1^{th}$ video frame. Its concrete network architecture can be also referred to in Fig. 3 (b). Specifically, it has the same encoder and decoder architecture as those in the spatial derivative reasoning branch. The main difference between the temporal derivative reasoning branch and the spatial derivative reasoning branch is that the former has the temporal reasoning layers with 1×1 kernel size to infer the temporal derivative representation. Here we propose three different architectures of the temporal reasoning module, which are named as the constant (CS) module, convex (CV) module, and concave (CC) module, respectively. As shown in Fig. 3 (a), the channel numbers of the temporal reasoning convolutional layers in the CS module are constantly set as 256. The channel number in the CV module gradually increases to 1024 and then decreases to 256. The channel number in the CC module gradually decreases to 64 and then increases to 256. After the temporal reasoning layers, the network uses the decoder network followed by a convolutional layer to generate a one-channel estimation map with size of 448, which corresponds to the desired temporal derivative map $\mathbf{D}_t$.

Motion Vector Reasoning Branch: The motion vector reasoning branch mainly follows the FlowNetS architecture [9] due to its success in the conventional flow estimation task. As shown in Fig. 2, it also consists of an encoder network and a decoder network. Basically, its encoder network propagates the input three-channel derivative map through 9 convolutoinal layers to obtain the feature map with spatial size of 7×7 and channel number of 1024. Then, the obtained feature maps is feed into the decoder network to obtain the final 2-channel PoseFlow estimation map with spatial size of 112×112.

Network Training: In training PFN, we follow the previous works [9, 14] to use the EndPoint Error (EPE), the standard error measure for the flow estimation tasks, as the loss function. EPE is computed as the average Euclidean distance over all pixels between the predicted motion vector and the ground truth.

To obtain the ground truth PoseFlow map, we leverage the ground truth human body joint locations provided by the existing benchmark datasets. Given the 2D ground truth joint locations in each frame, we first connect joints belonging to the same bone using straight lines and uniformly sample 10 pixels on each straight line. Then, we subtract 2D coordinates of the corresponding points between the $(t+1)^{th}$ and $(t)^{th}$ frame to compute motion vectors at the sampled pixels. Finally, we use the amplitudes of the computed motion vectors as the intensity of preliminary flow map, and smooth the preliminary map through a dilation process. More detailed description of the PoseFlow ground truth can be referred to in the supplementary material.

When training PFN, following FlowNetC [9], we choose the Adam as optimizer. The parameters of Adam is set as recommended in [20]. The PFN is trained through 30 epochs. The learning rate is set as 0.0001 for the first 15 epochs and reduced by a factor of 10 for the second 15 epochs. We use batch normalization for every bottom layer of concatenation operation (the black dots in Fig. 3). PFN is implemented in C++ and Matlab, and is based on the Caffe [16] and the publicly available implementation of FlowNet2. All of the experiments were run on a NVIDIA TitanXP GPU with 12GB memory.

4. Experiments

4.1. Datasets

We conducted experiments on three standard datasets:

CMU Panoptic dataset [17] records human behaviors in social interactions using an advanced multi-camera sensing system called the Panoptic Studio. Various games like *Ultimatum*, *Mafia*, and *Haggling* were played to evoke natural interactions among participants. In our experiments, we used the *Pose* subset. The body joints were annotated in each frame. After down-sampling, 8908 frames were used for training and 7924 frames for testing.

Poses in the Wild dataset [5] consists of 30 video sequences (totally 830 frames) extracted from Hollywood movies. It contains realistic poses in indoor and outdoor scenes, with background clutter, severe camera motion, non-frontal view, unusual poses and occlusions. Body joint are annotated for the upper human bodies. We followed [24] and use this dataset to test the generalization capacity of the flow estimation models trained on other datasets.

HMDB-51 dataset [21] consists of 766 videos, 51 different actions, three different splits of training and testing sets. We report the performance of PoseFlow and state-of-the-art flow estimation networks. Moreover, we compared PoseFlow-based video action recognition with state-of-the-art approaches based on optical flow. Note the actions in the HMDB-51 dataset have wide variations, making it very challenging for video action recognition.

4.2. PoseFlow Estimation

In this section, we conducted comprehensive evaluation of PFN on the *Pose* subset of the CMU Panoptic dataset. We used the standard error measure for flow estimation, i.e., the EndPoint Error (EPE). It is computed as the average pixel-wise Euclidean distance between the predicted flow vector and the ground truth.

We first compared seven baselines settings of PFN to analyze the contribution of each component. In Table 1, the baseline “PFN-C” denotes the PFN architecture, which inputs the concatenated adjacent video frames into a single forward encoder-decoder branch to infer $\frac{\partial \mathbf{I}}{\partial x}$, $\frac{\partial \mathbf{I}}{\partial y}$, $\frac{\partial \mathbf{I}}{\partial t}$ simultaneously. “PFN-D” denotes PFN shown in Fig. 2, where the input two video frames are decoupled as in two different forward branches. The next three baselines “PFN-D-T(CS)”, “PFN-D-T(CV)” and “PFN-D-T(CC)” include additional temporal reasoning layers with content, convex, and concave layer architecture, respectively. The sixth baseline “PFN-partial” uses the traditional image gradients to calculate $\frac{\partial \mathbf{I}}{\partial x}$, $\frac{\partial \mathbf{I}}{\partial y}$, and $\frac{\partial \mathbf{I}}{\partial t}$, and then directly feed the obtained $\frac{\partial \mathbf{I}}{\partial x}$, $\frac{\partial \mathbf{I}}{\partial y}$, and $\frac{\partial \mathbf{I}}{\partial t}$ into the middle layer of PFN during the training process. In this case, only the motion vector reasoning branch of PFN is trained to generate the final poseflow estimation. The last baseline “PFN-D-T(CS)-P” uses the $\frac{\partial \mathbf{I}}{\partial x}$, $\frac{\partial \mathbf{I}}{\partial y}$, and $\frac{\partial \mathbf{I}}{\partial t}$ obtained by the image gradient calculation as the supervision signals, pre-train the spatial derivative reasoning branch and the temporal derivative reasoning branch of “PFN-D-T(CS)”, and then followed by end-to-end poseflow learning.

The experimental results of all above baselines are shown in Table. 1. From the comparison between “PFN-C” and “PFN-D”, we obtained 1.9% and 1% performance gains in terms of the training and test error. This demonstrates the rationality of inferring the latent spatial and temporal motion derivative representations into two separate network branches over directly inferring them through a single network branch. It also shows that the inference of the $\frac{\partial \mathbf{I}}{\partial x}$, $\frac{\partial \mathbf{I}}{\partial y}$ and $\frac{\partial \mathbf{I}}{\partial t}$ relies on different hidden patterns rather than the shared ones.

From the comparison of “PFN-D-T(CS)”, “PFN-D-T(CV)” and “PFN-D-T(CC), we observe that the most effective way to implement the temporal reasoning is to use the convex layer architecture. The choice of different temporal reasoning architectures cause 1.2% and 0.7% performance variations in terms of training and test error, respectively. In addition, the comparison between “PFN-D-T(CV)” and “PFN-D” indicates that using the proper temporal reason block will produce extra 1.5% performance gain in testing. According to the above comparisons, we use **PFN-D-T(CV)** as our final PFN model due to its superior effectiveness.

The last two baselines “PFN-partial” and “PFN-D-

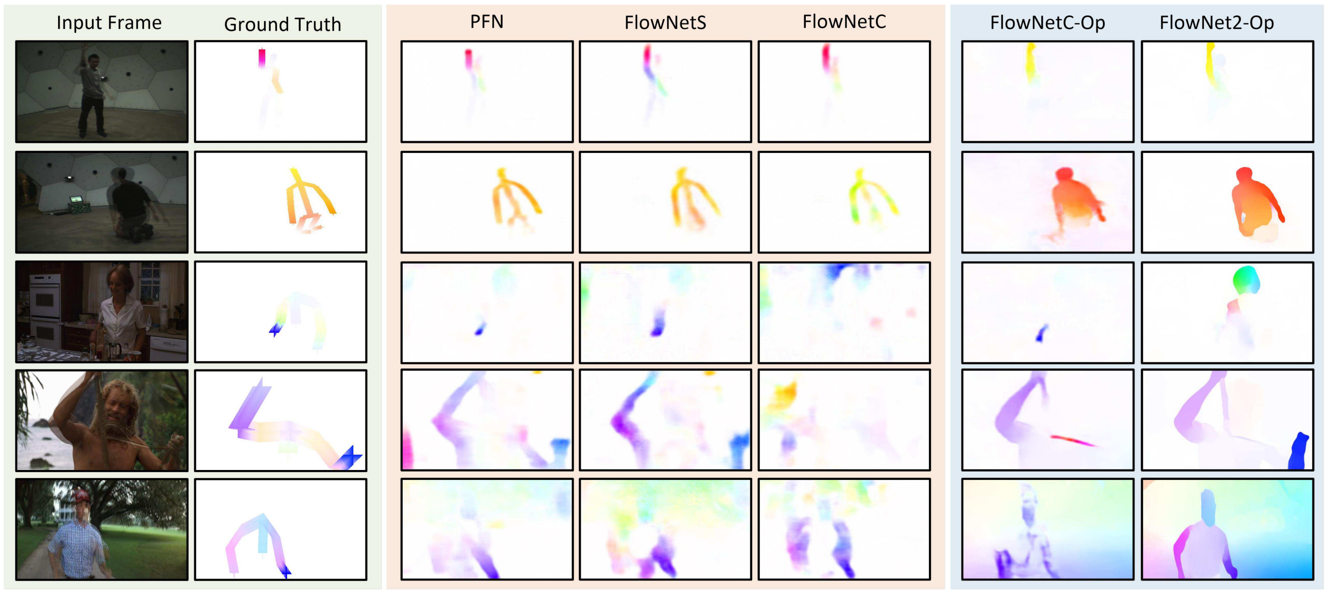

Figure 4. Some examples of the flow estimation results. The pink block indicates the predicted PoseFlow results by using different network architectures, while the blue block indicates the predicted optical flow results. The models in the pink block are trained on the pose CMU dataset with the PoseFlow ground-truth, while the models (FlowNetC-Op and FlowNet2-Op) in the blue block are trained on the traditional flow estimation dataset with the optical flow ground-truth.

Table 1. Comparison of PFN baseline models on the *Pose* subset of the CMU Panoptic dataset in terms of EPE.

Baseline name	Train error	Test error	Run time
PFN-C	0.243	0.346	**49ms**
PFN-D	0.224	0.336	59ms
PFN-D-T(CS)	0.233	0.328	60ms
PFN-D-T(CV)(final)	**0.223**	**0.321**	60ms
PFN-D-T(CC)	0.235	0.324	60ms
PFN-partial	0.531	0.539	174ms
PFN-D-T(CS)-P	0.226	0.331	60ms

Table 2. Comparison PFN with the state-of-the-art flow estimation models in PoseFlow estimation. EPE are reported on test sets of two databases [17] and [5].

Comparison methods	*pose* CMU dataset	Poses in the Wild dataset	Run time
FlowNetC	0.400	3.852	56ms
FlowNetS	0.430	1.818	**45ms**
FlowNet2	0.539	2.960	83ms
PFN(ours)	**0.321**	**1.661**	60ms

T(CS)-P" cause various degrees of performance degeneration comparing to **PFN-D-T(CV)**. Specifically, "PFN-partial" , about 20% worse than **PFN-D-T(CV)**, uses the $\frac{\partial \mathbf{I}}{\partial x}$, $\frac{\partial \mathbf{I}}{\partial y}$ and $\frac{\partial \mathbf{I}}{\partial t}$ obtained by the image gradient calculation to directly train the motion vector reasoning branch of PFN. "PFN-D-T(CS)-P", 1% worse than **PFN-D-T(CV)** in testing, pre-trains the spatial derivative reasoning branch and temporal derivative reasoning branch of PFN. This indicates that using $\frac{\partial \mathbf{I}}{\partial x}$, $\frac{\partial \mathbf{I}}{\partial y}$ and $\frac{\partial \mathbf{I}}{\partial t}$ obtained by the image gradient calculation will mislead the flow estimation even only using it for pre-training, which is the most critical bottle neck of the traditional flow estimation methods.

Finally, we compared PFN with three state-of-the-art DNN-based flow learning models: FlowNetS [9], FlowNetC [9] and FlowNet2 [14], respectively. We conducted comparison on both the *pose* subset of the CMU Panoptic dataset (short for "*pose* CMU") and the Poses in the Wild dataset. On the *pose* CMU dataset, we used the training and test split provided in the dataset. Then, to test the generalization capacity of different flow estimation approaches, we used the flow estimation models (including FlowNetS, FlowNetC, FlowNet2, and PFN) trained on the *pose* CMU dataset to predict the PoseFlow fields on all the video frames of the Poses in the Wild dataset. The visual comparisons are shown in Fig. 4 and the quantitative results are shown in Table 2. These comparisons show that PFN significantly outperforms other state-of-the-art deep flow models by $0.079 - 0.218$ EPE on the *pose* CMU dataset and $0.157 - 2.191$ EPE on the Poses in the Wild dataset. Based on our understanding, learning PoseFlow under human skeleton ground-truth would guide PFN to learn patterns on localizing human body and capturing the sparse flow fields. Whereas learning optical flow in the reviewer mentioned way can only capture the dense flow fields on all possible motion regions. Thus, PFN can better focus on human skeletons. Examples in Fig. 5 further demonstrate that PFN can better handle body occlusions, both in the simulated case and the real case. Besides, we also test

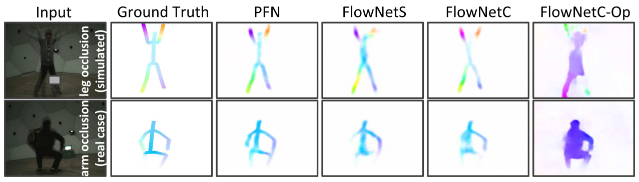

Figure 5. Quantitative comparisons between PFN and other approaches on occlusion cases.

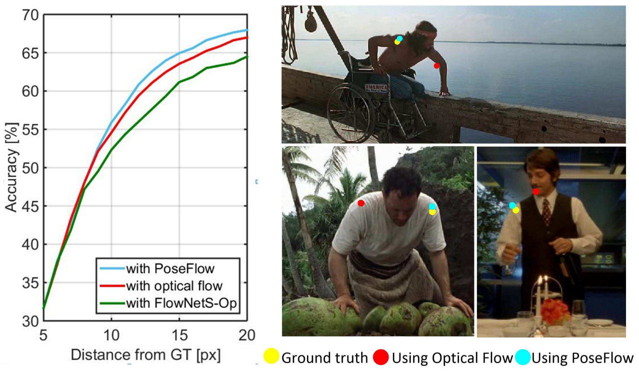

Figure 6. Quantitative and qualitative comparisons between the optical flow-based video pose estimation and PoseFlow-based video pose estimation results. We only show the joint locations of human shoulder in this figure.

the per-frame running times of using different flow estimation models. As shown in Table 2, the running times of these flow estimation models vary slightly. The proposed PFN require moderate running time, which is slightly slower than FlowNetS, approximately equal to FlowNetC, and faster than FlowNet2.

4.3. Pose Estimation

Existing approaches for video-based pose estimation used optical flow and the body joint locations in adjacent frames to help infer the body joints in the current frame. In this experiment, we explore the benefit of PoseFlow in this task. Specifically, we chose Pfister's method [24] as the baseline for localizing human shoulders in the Poses in the Wild dataset. The red curve in Fig. 6 shows accuracy vs distance from Ground Truth curve by [24]. The higher curve indicates better performance. Then, we used PoseFlow to replace the optical flow in [24] to generate the joint locations of human shoulders. The obtained accuracy vs distance from Ground Truth curve is shown in cyan in Fig. 6. We also show some qualitative comparison results in Fig. 6. From the comparisons shown in Fig. 6, we can observe the benefit of using PoseFlow in pose estimation: Even just adopting a simple warping strategy as it is in [24], the PoseFlow-based pose estimation can already obtain encouragingly better results especially for the examples shown in Fig. 6. More advanced strategies for introducing flow fields in pose estimation could lead to full advantages of PoseFlow, which will be considered in our future work.

Table 3. Velidation the effectiveness of PoseFlow for action recognition on the HBDM-51 dataset.

Methods	Accuracy
ST-ResNet w/o motion [10]	43.42%
ST-ResNet w optical flow (FlowNet2)	48.58%
ST-ResNet w Poseflow (FlowNetS)	50.68%
ST-ResNet w Poseflow (our PFN)	**51.74%**

4.4. Action Recognition

In this section, we demonstrate the effectiveness of PoseFlow in the action recognition task. Our experiment is based on one of the state-of-the-art action recognition framework [10]. This framework consists of a two-stream ConvNet model (One motion stream and one appearance stream) with residual connections. We report the recognition accuracy without any motion information as "ST-ResNet w/o motion" in Table 3. Then, we used the pre-trained FlowNet2, which is **the most state-of-the-art deep model for estimation optical flow** [14], to generate optical flow fields for introducing motion cues. This method, denoted by "ST-ResNet w optical flow (FlowNet2)" obtains 5.16% performance gain as compared with "ST-ResNet w/o motion". Finally, we replace the optical flow fields generated by FlowNet2 by the PoseFlow fields generated by the pre-trained PFN, denoted by "ST-ResNet w Poseflow (our PFN)". As shown in Table 3, with the informative motion vectors on human bodies, PoseFlow can further improve the action recognition accuracy by 3.16%. Note that the accuracies reported in Table 3 are the mean accuracy of all three train-test splits as reported in previous works.

5. Conclusion

PoseFlow is a rich and powerful representation that jointly describes body pose and motion. Comparing to the conventional motion representation based on optical flow, PoseFlow can suppress the background and motion blur, and is robust to occlusion. To learn such a rich representation without resorting to heavy neural networks, we developed PoseFlow Net (PFN), a functional-structured networks that explicitly models the spatial derivative reasoning, temporal derivative reasoning and motion vector reasoning. Experiments show that this cleaner representation, PoseFlow, resulted in substantial improvement on pose estimation and action recognition in videos. Moreover, PoseFlow does not rely on accurate body joint detection and dense flow tracking, therefore can be widely applied to video data with low resolution, video blur and small human body size.

Acknowledgement: This work was supported in part by the National Science Foundation of China under Grant 61473231, Grant 61522207, Grant 61373114, Grant 61661166011 and the National Science Foundation under the grants RI-1617953. The content is solely the responsibility of the authors and does not necessarily represent the official views of the funding agencies.

References

[1] D. Bau, B. Zhou, A. Khosla, A. Oliva, and A. Torralba. Network dissection: Quantifying interpretability of deep visual representations. *CVPR*, 2017.

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

[3] C.-H. Chen and D. Ramanan. 3d human pose estimation = 2d pose estimation + matching. In *CVPR*, 2017.

[4] D. Cheng, X. Chang, L. Liu, A. G. Hauptmann, Y. Gong, and N. Zheng. Discriminative dictionary learning with ranking metric embedded for person re-identification. In *IJCAI*, 2017.

[5] A. Cherian, J. Mairal, K. Alahari, and C. Schmid. Mixing body-part sequences for human pose estimation. In *CVPR*, 2014.

[6] G. G. Chrysos, E. Antonakos, P. Snape, A. Asthana, and S. Zafeiriou. A comprehensive performance evaluation of deformable face tracking in-the-wild. *IJCV*, pages 1–35, 2016.

[7] M. L. Chuang Gan, Y. Yang, Y. Zhuang, and A. G. Hauptmann. Exploring semantic interclass relationships (sir) for zero-shot action recognition. In *AAAI*, 2015.

[8] Y. Dong, H. Su, J. Zhu, and F. Bao. Towards interpretable deep neural networks by leveraging adversarial examples. *CVPR*, 2017.

[9] 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.

[10] C. Feichtenhofer, A. Pinz, and R. P. Wildes. Spatiotemporal residual networks for video action recognition. In *NIPS*, 2016.

[11] C. Feichtenhofer, A. Pinz, and A. Zisserman. Convolutional two-stream network fusion for video action recognition. In *CVPR*, 2016.

[12] C. Gan, M. Lin, Y. Yang, G. de Melo, and A. G. Hauptmann. Concepts not alone: Exploring pairwise relationships for zero-shot video activity recognition. In *AAAI*, 2016.

[13] G. Gkioxari, A. Toshev, and N. Jaitly. Chained predictions using convolutional neural networks. In *ECCV*, 2016.

[14] 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.

[15] S. Ji, W. Xu, M. Yang, and K. Yu. 3d convolutional neural networks for human action recognition. *TPAMI*, 35(1):221–231, 2013.

[16] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe:convolutional architecture for fast feature embedding. pages 675–678, 2014.

[17] H. Joo, H. Liu, L. Tan, and L. Gui. Panoptic studio: A massively multiview system for social motion capture. In *ICCV*, 2015.

[18] A. Kar, N. Rai, K. Sikka, and G. Sharma. Adascan: Adaptive scan pooling in deep convolutional neural networks for human action recognition in videos. In *CVPR*, 2017.

[19] 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.

[20] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015.

[21] H. Kuehne, H. Jhuang, R. Stiefelhagen, and T. Serre. Hmdb51: A large video database for human motion recognition. In *ICCV*, 2013.

[22] Z. Li, L. Yao, F. Nie, and D. Zhang. Multi-rate gated recurrent convolutional networks for video-based pedestrian reidentification. In *AAAI*, 2018.

[23] X. Liang, Y. Wei, L. Lin, Y. Chen, X. Shen, J. Yang, and S. Yan. Learning to segment human by watching youtube. *TPAMI*, 39(7):1462–1468, 2017.

[24] T. Pfister, J. Charles, and A. Zisserman. Flowing convnets for human pose estimation in videos. In *ICCV*, 2015.

[25] A. Ranjan and M. J. Black. Optical flow estimation using a spatial pyramid network. *arXiv preprint arXiv:1611.00850*, 2016.

[26] K. Simonyan and A. Zisserman. Two-stream convolutional networks for action recognition in videos. In *NIPS*, 2014.

[27] J. Song, L. Wang, L. Van Gool, and O. Hilliges. Thin-slicing network: A deep structured model for pose estimation in videos. *arXiv preprint arXiv:1703.10898*, 2017.

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

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

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

[31] G. Varol, I. Laptev, and C. Schmid. Long-term temporal convolutions for action recognition. *TPAMI*, 2017.

[32] C. Wang, Y. Wang, and A. L. Yuille. An approach to pose-based action recognition. In *CVPR*, 2013.

[33] D. Zhang and M. Shah. Human pose estimation in videos. In *ICCV*, 2015.

[34] S. Zuffi, J. Romero, C. Schmid, and M. J. Black. Estimating human pose with flowing puppets. In *ICCV*, 2013.