2024 International Conference on Advanced Robotics and Mechatronics (ICARM) | 979-8-3503-8572-4/24/$31.00 ©2024 IEEE | DOI: 10.1109/ICARM62033.2024.10715993

2D Hand Pose Estimation from A Single RGB Image through Flow Model*

Ying Hao, Shaochen Wang, and Zhen Kan

***Abstract*— Hand pose estimation is a meaningful yet challenging task, primarily due to the intricate nature of hands and common issues such as self-occlusion. Traditional approaches often struggle with effectively mapping image features to a manageable base distribution, frequently neglecting the vital relationship between local and global features in pose estimation. To address these challenges, we propose 2DHandFlow, a flow-based framework that processes RGB images to output the 2D pose of hands. Our approach leverages visual transformers to embed image patches and uses a transformer encoder to extract global features. The 2DHandFlow learns to convert visual features from RGB images into a tractable distribution, enhancing the likelihood of accurately recognizing keypoints. Our model adopts a 2D normalizing flow model, which employs fully convolutional networks and a 2D loss function to represent the relationships among hand keypoints. By integrating visual transformers for feature extraction, our method has achieved high accuracy in 2D hand pose estimation. Impressively, our approach achieves 0.97 and 0.92 PCK on the Stereo Hand Pose Tracking Benchmark (STB) and the Rendered Hand Dataset (RHD), respectively. These results indicate superior performance. The project website is: https://github.com/HYing268/2dhandflow.**

I. Introduction

Hand pose estimation, alongside hand recognition techniques [1], plays a pivotal role in many fields such as sign language recognition [2], human-computer interaction [3], and virtual reality [4]. While algorithms such as Convolutional Pose Machine (CPM) [5] and Single-stage multiperson Pose Machine (SPM) have established body pose estimation, they can't be directly applied to hand pose estimation due to hands' complex structure, higher dexterity, and self-occlusion. Hand pose estimation requires explicit modeling of the structural relationships between hand keypoints.

Due to low implementation cost and easy access to data acquisition, 2D hand pose estimation is a recent research focus. Leveraging depth sensors, hand pose estimation has shifted from relying on hardware-based solutions like data gloves to computer vision based technology [6]. Nevertheless, depth sensors [7] generally suffer from limited resolution and sensitivity to lighting [8]. Existing methods face issues such as low recognition accuracy, the need for costly hand segmentation masks and the challenge of recognizing gestures across different scales. Recently, the advances of Deep Convolutional Neural Networks (DCNNs) [9] have spurred research into hand pose estimation using RGB images. While DCNNs can learn highly effective feature representations, they do not explicitly capture spatial relationships between joints or keypoints, leading to joint inconsistency [10].

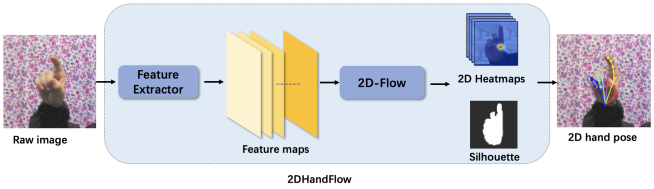

Fig. 1. 2DHandFlow transforms the features of an input image to a standard normal distribution, where the mean indicates the features of the keypoints.

Generative model is a promising approach for hand pose estimation [11]. By modeling the statistical correlation between the 2D hand pose space and the associated input data space, hand poses can be estimated in a semi/self-supervised manner. For instance, Depth-image Guided GAN (DGGAN) [12] produces lifelike depth maps from the input RGB image for the regularization of 3D hand pose estimation model. Cycle-consistent Generative Adversarial Network (Cycle-GAN) [13] generates a depth map with color-coded predictions for the fingertips, wrist, and palm conditioned on the actual depth image. However, GANs based approaches lead to increased training complexity due to the adversarial training. Recent works [14], [15] use a trainable method that optimizes the log-likelihood of image features incorporated into a standard normal distribution to facilitate keypoint localization and pose estimation. Nevertheless, the initial 1D normalizing flow model requires the flattening of 2D input features into a 1D vector for distribution estimation, which results in the loss of the intrinsic spatial positioning of the 2D image. This limitation restricts the model's ability to fully exploit the rich spatial structure and correlations that are crucial for accurate estimation of hand pose from 2D images.

To address this problem, we extend normalizing flow to 2D space and propose the 2DHandFlow, which preserves the spatial positional information of the 2D image for more accurate hand pose representation. This design improves hand keypoint recognition accuracy by better understanding hand pose variations and nuances. Additionally, by capturing the complex spatial dependencies between keypoints, 2DHandFlow can effectively model the correlations and interactions among different parts of the hand, which enhances the accuracy and robustness of hand keypoint estimation. In hand pose estimation, existing studies concentrated on CNN

*This work was supported in part by National Key R&D Program of China under Grant 2022YFB4701400/4701403 and the National Natural Science Foundation of China under Grant U2013601.

Ying Hao (email: sa22010013@mail.ustc.edu.cn), Shaochen Wang (email: wangsam39@gmail.com), and Zhen Kan are with the Department of Automation, University of Science and Technology of China, Hefei 230026, China.

Zhen Kan is the corresponding author (phone: 63602835; email: zkan@ustc.edu.cn).

networks like ResNet for feature extraction. In contrast, we employ Visual Transformer (ViT) [16], [17], which offers a global receptive field enhancing the processing of both global and local image information. Its self-attention mechanisms can help understand the relationships between all feature pairs within an image, which allows ViT to preserve semantic information at various depths, an aspect where CNNs tend to fall short due to fixed-weight convolutional filters. As shown in Fig. 1, our approach uses a feature extractor to derive visual features, which are then fed into the 2DHandFlow model for probability density estimation. This approach, bolstered by the transformer's enhanced analysis of keypoint relationships and hand-environment interactions, leads to more accurate hand joint estimations. Extensive experiments show that the 2DHandFlow achieves top-tier performance, with PCK scores of 0.92 on the RHD and 0.97 on the STB.

The main contributions of this paper are outlined as follows. First, we introduce 2DHandFlow, a two-dimensional, flow-based framework specifically designed for 2D hand pose estimation. Second, the 2DHandFlow is versatile, functioning as a plug-and-play solution compatible with a variety of feature extractors. A significant advancement is the integration of the vision transformer module, which supersedes traditional CNNs. In addition, in practical applications, 2DHandFlow has demonstrated state-of-the-art performance in 2D hand pose estimation. It achieves remarkable results on two widely recognized datasets.

II. Method

A. Problem Formulation

To explain the idea behind 2DHandFlow, we first examine the regression problem through the lens of maximum likelihood estimation (MLE). Given an input image $\mathcal{I}$, the regression model forecasts a distribution $P_\theta(\mu|\mathcal{I})$ that represents the likelihood of the true value occurring at the location μ, with θ representing the model parameters to be learned. Given the intrinsic uncertainties and potential errors in the labeling process, the labeled location μ_g can be regarded as an approximation sampled close to the true value the human annotator determined. The learning process' goal is to adjust the model parameters θ to make the observed label μ_g as probable as possible. The loss function for MLE process is defined as

$$\mathcal{L}_{\text{mle}} = -\log P_\theta(\mu|\mathcal{I}) \mid_{\mu=\mu_g} . \tag{1}$$

As shown in [18], only when we have prior knowledge of the target variable distribution can we better construct a loss function to facilitate the learning of model parameters. As a result, improving the accuracy of the density function is crucial. However, due to the unknown analytical expression of the underlying distribution, regressing multiple values and creating the density function directly is infeasible.

To address this challenge and enhance human pose estimation, we introduce a new method that utilizes the power of normalizing flow to estimate the underlying distribution. Normalizing flow is a reversible generative model that map an initial distribution to a known probability distribution through learned transformations. By ensuring reversible transformations, the model can estimate the probability density function of a given input, which we use to model the distribution of hand joint location and learn the likelihood estimation of 2D hand poses. Since the main notion of flow is the distribution transformation formula, we define a reversible function $f : X \to Z$ to map image features x to the hidden variable z, i.e.,

$$\begin{aligned} p_X(x) &= p_Z(f(x)) \left|\det\left(Jf(x)\right)\right|, \\ p_Z(z) &= p_X(f^{-1}(z)) \left|\det\left(Jf^{-1}(z)\right)\right|. \end{aligned} \tag{2}$$

where $\det(\cdot)$ indicates the determinant and$Jf(x)$ is Jacobian of $f(x)$. In a flow model, each flow layer is represented by a reversible function. Given a sample from the initial distribution, we can gradually transform the initial distribution into the target probability distribution by sequentially applying these reversible functions. This invertibility allows for implicit learning of the inverse mapping. By repeatedly drawing samples from a straightforward, tractable distribution, approximating the complete posterior distribution is possible. Based on prior research, we employ and expand upon their frameworks to model the complete posterior distribution of plausible 2D hand pose given a RGB image.

Problem The aim of this work is to develop a hand pose estimation mechanism, namely 2DHandFlow, which leverages normalizing flows to train a model to accurately map the input RGB image data $\mathcal{I}$ to 2D heatmaps of hand keypoints' position probability distribution $\mathcal{H}_{2D} \in \mathbb{R}^{M\times H\times W}$ and silhouette representing the outline of the hand $\mathcal{S} \in \mathbb{R}^{H\times W}$ of different hand pose, where M = 21 means the number of hand joints and (H;W) = (64;64) indicates the 2D heatmaps' resolution and silhouette.

B. Feature Extractor

Feature extraction involves transforming raw data into higher-level representations. In our method, we utilize ResNet or ViT to extract representative features from the input image. The ViT [16] involves dividing the input image into a series of fixed-size image patches. Each patch is flattened through the patch embedding layer, and positional information is added through position embedding. The patch is treated as a sequence of input vectors and multiple layers of transformer encoders are used to process the input. Each encoder layer includes of a multi-head self-attention mechanism and a feed-forward neural network, aiming to capture the relationship between image patches and obtain global contextual information. The global image feature representation is obtained by performing pooling on the output of the last layer. The ResNet model [19] is a deep convolutional neural network composed of multiple residual blocks, which is used to learn residual feature representations. The input image in ResNet is initially handled by initial convolutional layer to generate low-level visual patterns. Subsequently, the residual blocks consist of successive convolutional operations which help in capturing increasingly complex and abstract features. When using ViT, we specifically

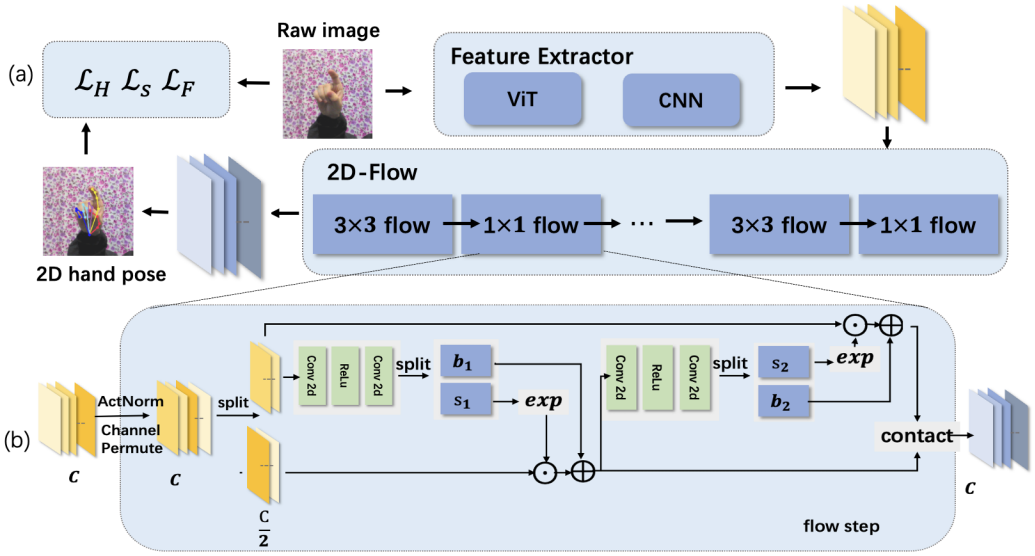

Fig. 2. (a) The whole pipeline for 2D hand pose estimation composing of feature extractor and 2DHandFlow. Either CNN or ViT can be used as the feature extractor. The 2DHandFlow is alternatively stacked by the "3 × 3" and "1 × 1" flow. (b) An example flow step of our 2DHandFlow, where the "Conv 2d" can respectively be a 3 × 3 or 1 × 1 convolution layer for 3 × 3 or 1 × 1 flow.

use the features from a certain layer as it allows us to better capture the connection between global feature and local patches. When using ResNet as the feature extractor, we utilize the features from the last layer of the first three blocks directly and feed them to united 2DHandFlow models.

C. 2DHandFlow Model Architecture

Normalizing flow is used to map data to a latent space where the distribution is tractable by applying a sequence of invertible transformations to the input data. Let $Z \in \mathbb{R}^d$ be a random variable with a base distribution, such as standard multivariate Gaussian distribution $p_Z(z)$, and $X \in \mathbb{R}^d$ denote a random variable with any arbitrary complex distribution $p_X(x)$, with a bijective invertible mappingïŒ the normalizing flows project X into Z. As to the bijection function, the rule of change-of-variable determines the model distribution on X by

$$p_X(x) = p_Z(z)\left|\det\left(\frac{\partial z}{\partial x}\right)\right| = p_Z(f_\theta(x))\left|\det\left(\frac{\partial f_\theta(x)}{\partial x}\right)\right|. \tag{3}$$

where z is sampled from a standard normal distribution, $\frac{\partial f_\theta(x)}{\partial x}$ is the Jacobian of a bijective invertible flow model with $z = f_\theta(x)$ and $x = f_\theta^{-1}(z)$, and θ is the 2D flow model's parameter. We can estimate the log likelihoods of image features from $p_Z(z)$ by

$$\log p_X(x) = \log p_Z(f_\theta(x)) + \log\left|\det\left(\frac{\partial f_\theta(x)}{\partial x}\right)\right|. \tag{4}$$

In implementation, to obtain a complex distribution with expressive mapping, our flow model f_{2d} is constructed by chaining a sequence of multiple invertible transformation blocks f_i. Flow models build complex distribution mappings by concatenating blocks f_i, where each block is composed of an activation normalization layer and an affine coupling layer. Each block maps the input data from one representation space to another, and through the successive concatenation of these blocks, the data is transformed into the target probability distribution. The relationship between input X and latent representation Z becomes:

$$\begin{aligned} X \xrightarrow{f_1} H_1 \xrightarrow{f_2} H_2 \xrightarrow{f_3} \cdots \xrightarrow{f_K} Z, \\ X \xleftarrow{f_1^{-1}} H_1 \xleftarrow{f_2^{-1}} H_2 \xleftarrow{f_3^{-1}} \cdots \xleftarrow{f_K^{-1}} Z, \end{aligned} \tag{5}$$

where the 2D flow model is $f_{2d} = f_1 \circ f_2 \circ f_3 \circ \cdots \circ f_K$ with K transformation blocks. The log-likelihoods of X can be further written as

$$\log p_X(x) = \log p_Z(f_\theta(x)) + \sum_{i=1}^{K} \log\left|\det\left(\frac{\partial f_i}{\partial f_{i-1}}\right)\right|. \tag{6}$$

And each transformation block f_i consists of multiple steps. As shown in Fig. 2, Actnorm serves as an activation normalization layer which utilizes a scale and bias with initialization dependent on the data. Normalizing the activation values for each mini-batch sample helps alleviate the problem of gradient vanishing. The affine coupling layer splits the input into two segments. One segment can undergo an affine transformation influenced by the other segment, which is kept unchanged for an easy reverse transformation. In this way, the affine coupling layer can achieve a non-linear transformation of the input data while maintaining the dimensionality of the input data. This is particularly useful for modeling complex probability distributions. Following [20], we apply affine coupling layers in each block, which each step can be defined as

$$\begin{aligned} y_a, y_b &= split(y), \\ y_a^{'} &= y_a, \\ y_b^{'} &= s(y_a) \odot y_b + b(y_a), \\ y^{'} &= concat(y_a^{'}, y_b^{'}), \end{aligned} \tag{7}$$

where $s(y_a)$ and $b(y_a)$ are outputs of the two neural networks. The $split(y)$ function performs splitting and

$concat(y_a^{'}, y_b^{'})$ function performs concatenation along the channel dimension. And $\odot$ represents the element-wise multiplication. To avoid multiplication by zero and ensure the block's invertibility, the exponential function is employed.

The original normalizing flow model uses fully connected networks for the two subnets, $s(y_a)$ and $b(y_a)$. However, by flattening and squeezing the input visual features from 2D to 1D, the spatial positional relationships within the feature map are destroyed. To address this limitation and enable a 2D approach in the normalizing flow model, we employ two-dimensional convolution layers in the default subnet. This allows us to preserve the spatial information within the flow model. This approach utilizes a fully convolutional network that alternates between 3×3 convolutions and 1×1 convolutions. This preserves the spatial information in the flow model by capturing the local patterns and adjusting the loss function accordingly. By adopting this fully convolutional network, the flow model is ensured to retain the spatial relationship in the input visual features.

To mitigate instabilities arising from the exponential function in the coupling block, the parameterized soft clamping mechanism [21] is employed. This scale coefficients s is applied as the last layer to the outputs s_1 and s_2 by the activation function

$$\sigma_\alpha(s) = \frac{2\alpha}{\pi} \arctan \frac{s}{\alpha}. \tag{8}$$

where s refers to network input. This limits the values to the interval $(-\alpha, \alpha)$, thereby preventing the scaling components from reaching excessively large magnitudes.

Loss Terms The loss function of 2DHandFlow is defined as $\mathcal{L}_{2D} = \lambda_H \mathcal{L}_H + \lambda_S \mathcal{L}_S + \lambda_F \mathcal{L}_F$ with tuning parameter $\lambda_H, \lambda_S, \lambda_F \in \mathbb{R}^+$, where

$$\mathcal{L}_H = \frac{1}{M} \sum_{m=1}^{M} \sum_{i=1}^{H} \sum_{j=1}^{W} \left(\mathcal{H}_{2D}^m (i,j) - \hat{\mathcal{H}}_{2D}^m (i,j) \right)^2, \tag{9}$$

$$\begin{aligned} \mathcal{L}_S = & \sum_{i=1}^{H} \sum_{j=1}^{W} \left(\mathcal{S}_{gt} (i,j) \log(\mathcal{S}_{pred} (i,j)) \right. \\ & \left. + (1 - \mathcal{S}_{gt} (i,j)) \log (1 - \mathcal{S}_{pred} (i,j)) \right), \end{aligned} \tag{10}$$

$$\mathcal{L}_F = \frac{1}{K} \sum_{k=1}^{K} \left(\frac{1}{2} \sum \| p_Z (z_k) \|^2 - \log \left| \det \left(\frac{\partial z_k}{\partial x_k} \right) \right| \right). \tag{11}$$

First, the heatmaps loss, denoted as $\mathcal{L}_H$, is defined as the mean squared error (MSE) calculated on a per-pixel basis between the prediected and actual 2D heatmaps. The actual 2D heatmap $\mathcal{H}_{2D}^{(m)}$ of keypoint m is created by placing 2D Gaussian distributions at the keypoint annotations of m with a standard deviation of $\sigma = 1.5$. Secondly, the silhouette loss $\mathcal{L}_S$ is computed as the cross-entropy loss between the estimated silhouette $\mathcal{S}_{pred}$ and the actual silhouette $\mathcal{S}_{gt}$. The heatmaps loss focuses on improving the accuracy of keypoint localization, while the silhouette loss aims to enhance object contour recognition. Thirdly, the flow loss $\mathcal{L}_F$ aims to lower the model's complexity and improve its generalization by penalizing the sum of squares of the model output and the logarithm of the Jacobian matrix's determinant.

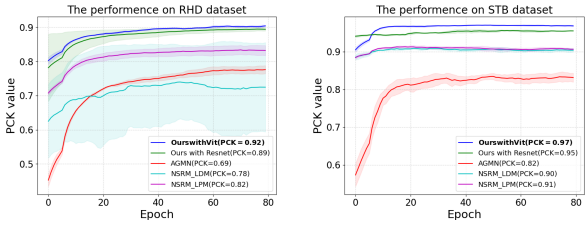

Fig. 3. Quantitative results of 2DHandFlow in accordance PCK. The comparisons results with sota methods for 2D hand pose estimation on dataset RHD and STB.

III. Experiment

In this section, 2D hand pose estimation is evaluated quantitatively and qualitatively against state-of-the-art results. Ablation studies are performed to highlight the effectiveness of each model component.

Implementation Details We implement our model using PyTorch and train it on NVIDIA GeForce RTX 3090 graphic cards. Table I presents the details of feature extractor's structure. In our approach, we only utilize the feature maps from a specific layer of the vision transformer. Finally, we take the average value as the final result. In 2DHandFlow, we employ 9-step flows for all feature extractors. The hyper-parameters are set to balance different types of supervision. Our model is trained using the Adam optimizer with a learning rate of 1e-3 and weight decay of 1e-5. The training follows a 100-epoch schedule, and the batch size is set to 32.

Datasets We train and score this method on two public datasets: the Rendered Hand Dataset (RHD) [22] and the Stereo Hand Pose Tracking Benchmark (STB) [23]. For training purposes, we selected 10 sequences from the STB dataset, while the remaining 2 sequences were reserved for testing, following the approach described in [22]. Similar to [24], we shifted the root joints in STB from the palm to the wrist, ensuring consistency with the RHD. In both datasets, the left hand samples were mirrored to the right hand.

Metrics We employed the percentage of correct 2D keypoints (PCK), which was measured at distance thresholds spanning from 20 mm to 50 mm, to assess the algorithm performance.

Results For quantitative analysis, we parabole our approach in accordance PCK with other methods [25]–[29]

TABLE I
SPECIFIC DETAILS OF DIFFERENT FEATURE EXTRACTORS

Backbone	Input Size	Block Index	Feature Size
CaiT-M48-distilled	448	40	28
DeiT-base-distilled	384	7	24
ResNet18	256	[1,2,3]	[64,32,16]
Wide-ResNet50-2	256	[1,2,3]	[64,32,16]

TABLE II
COMPARISONS WITH STATE-OF-THE-ART METHODS IN ACCORDANCE PCK

Method		Ours(ResNet)	Ours(ViT)	AGMN [25]	NSRM(LDM) [26]	NSRM (LPM) [26]	FreiHand [27]	KCR [28]	SRHandNet [29]
PCK	RHD	0.89	0.92	0.69	0.78	0.82	0.77	0.83	0.46
	STB	0.95	0.97	0.82	0.90	0.91	0.78	0.75	0.53

on both RHD and STB. Fig. 3 shows the comparison of performance on the above mentioned datasets and Table II summarizes numerical results. On STB dataset, our 2DHandFlow reaches 0.95 PCK using Resnet as feature extractor and 0.97 PCK using ViT, achieving an absolute PCK improvement of 15.66% comparing with AGMN training with unary branch and approximately 4.44% in contrast with NSRM with different basic representation (LDM or LPM). On RHD dataset, 2DHandFlow achieves 0.89 PCK using Resnet and 0.92 PCK using ViT, outperforming all the parabole methods. The results in two different datasets indicate that our model can adapt to various hand pose estimation settings. Utilizing ViT for feature extraction, as opposed to ResNet, can yield superior results due to its stronger global and local modeling capabilities. Since the RHD contains more variation, all results are lower than STB datasets. As shown in Fig. 3, we preserve the pre-trained parameters of feature extractors network to improve convergence speed.

Fig. 4 and Fig. 5 show the qualitative results through the generated keypoints. The results with RHD dataset especially demonstrate the effectiveness of our hand pose estimation model under various challenging conditions, including cluttered backgrounds, different environments, similar color variations and so on. Our method consistently achieves good results in these scenarios.

A. Ablation Study

Ablation study is conducted to show the effect of main components within our method. These experiments are completed on STB dataset because STB provides consistent image quality and annotations facilitating data analysis and algorithm complexity. The result aligns with our expectation that using 2D flow model can significantly enhance estimating 2D hand pose, and using ViT as feature extractor can further build up the experimental capability.

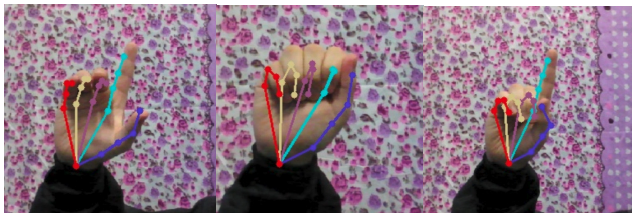

Fig. 4. Visualization of forecasted hand pose for samples from STB dataset.

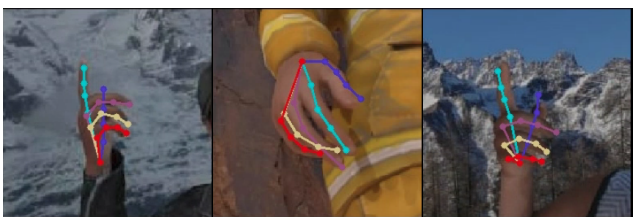

Fig. 5. Visualization of forecasted hand pose for samples from RHD dataset.

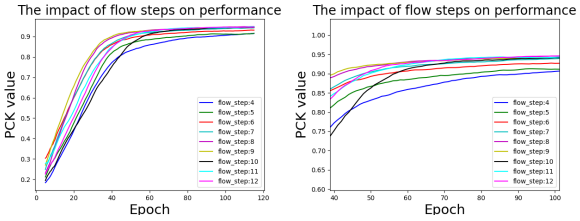

Fig. 6. The impact of flow steps on flow model performance.

Flow Step To investigate the effect of network configurations on Flow model performance, we conduct experiments by varying the number of flow steps K. We started with a baseline 2DHandFlow architecture that includes 9 flow steps and then systematically manipulated the network structure by adding or removing steps. As shown in Fig. 6, increasing the number of flow steps can improve the final accuracy, but at step 9, we achieved a balance between efficiency and performance.

Dimension of probability estimation Compared to using 2D convolutional layers, we flatten and squeeze the input visual features from ResNet in Fig. 7(a). The experiment reveals that enabling a 2D approach in the normalizing flow model can obtain improved performance because the spatial information is preserved without dimension loss.

Alternating network structure Based on the experiment in Fig. 7(b), it is clear that convolution kernel selection in subnet is a factor that influences the experimental results. In order to strike a balance between accuracy and inference speed, we utilize alternating 3×3 and 1×1 convolution kernels for large model capacities backbone networks such as DeiT, CaiT, and Wide-ResNet50-2. However, we rely solely on the 3×3 convolution layer for the backbone network with smaller model capacities like ResNet18.

IV. Conclusion

In this paper, a lightweight 2D flow model is developed, which projects the feature distribution of regular images onto a standard normal distribution. In testing, the probabilities obtained from this projection serve as the confidence score for each hand joint. Extensive experiments performed on the RHD and STB show the superiority of 2DHandFlow over state-of-the-art methods regarding accuracy and reasoning efficiency.

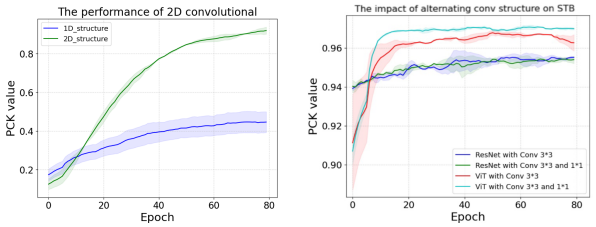

Fig. 7. (a) The performance of 2DHandFlow using 2D convolutional layers (b) The performance using alternating convolution kernel design with large model capacities

REFERENCES

[1] Y. Zhang, C. Cao, J. Cheng, and H. Lu, "Egogesture: A new dataset and benchmark for egocentric hand gesture recognition," *IEEE Trans Multimedia*, vol. 20, no. 5, pp. 1038–1050, 2018.

[2] D. Avola, M. Bernardi, L. Cinque, G. L. Foresti, and C. Massaroni, "Exploiting recurrent neural networks and leap motion controller for the recognition of sign language and semaphoric hand gestures," *IEEE Trans Multimedia*, vol. 21, no. 1, pp. 234–245, 2018.

[3] S. Wang, Z. Zhou, B. Li, Z. Li, and Z. Kan, "Multi-modal interaction with transformers: bridging robots and human with natural language," *Robotica*, vol. 42, no. 2, pp. 415–434, 2024.

[4] S. Wang, W. Zhang, Z. Zhou, J. Cao, Z. Chen, K. Chen, B. Li, and Z. Kan, "What you see is what you grasp: User-friendly grasping guided by near-eye-tracking," in *IEEE Int. Conf. Dev. Learn., ICDL*. IEEE, 2023, pp. 194–199.

[5] S.-E. Wei, V. Ramakrishna, T. Kanade, and Y. Sheikh, "Convolutional pose machines," in *Proc IEEE Comput Soc Conf Comput Vision Pattern Recognit*, 2016, pp. 4724–4732.

[6] A. Elboushaki, R. Hannane, K. Afdel, and L. Koutti, "Improving articulated hand pose detection for static finger sign recognition in rgb-d images," *Multimedia Tools Appl*, vol. 79, no. 39-40, pp. 28 925–28 969, 2020.

[7] K. Chen, S. Wang, B. Xia, D. Li, Z. Kan, and B. Li, "Tode-trans: Transparent object depth estimation with transformer," in *Proc IEEE Int Conf Rob Autom*. IEEE, 2023, pp. 4880–4886.

[8] D. Mehta, S. Sridhar, O. Sotnychenko, H. Rhodin, M. Shafiei, H.-P. Seidel, W. Xu, D. Casas, and C. Theobalt, "Vnect: Real-time 3d human pose estimation with a single rgb camera," *ACM Trans Graphics*, vol. 36, no. 4, pp. 1–14, 2017.

[9] Y. Xia, S. Wang, and Z. Kan, "A nested u-structure for instrument segmentation in robotic surgery," in *IEEE Int. Conf. Adv. Robot. Mechatronics, ICARM*, 2023, pp. 994–999.

[10] L. Ke, M.-C. Chang, H. Qi, and S. Lyu, "Multi-scale structure-aware network for human pose estimation," in *Lect. Notes Comput. Sci.*, 2018, pp. 713–728.

[11] S. Baek, K. I. Kim, and T.-K. Kim, "Weakly-supervised domain adaptation via gan and mesh model for estimating 3d hand poses interacting objects," in *Proc IEEE Comput Soc Conf Comput Vision Pattern Recognit*, 2020, pp. 6121–6131.

[12] L. Chen, S.-Y. Lin, Y. Xie, Y.-Y. Lin, W. Fan, and X. Xie, "Dggan: Depth-image guided generative adversarial networks for disentangling rgb and depth images in 3d hand pose estimation," in *Proc. - IEEE Winter Conf. Appl. Comput. Vis., WACV*, 2020, pp. 411–419.

[13] F. Farahanipad, M. Rezaei, A. Dillhoff, F. Kamangar, and V. Athitsos, "A pipeline for hand 2-d keypoint localization using unpaired image to image translation," in *ACM Int. Conf. Proc. Ser.*, 2021, pp. 226–233.

[14] T. Wehrbein, M. Rudolph, B. Rosenhahn, and B. Wandt, "Probabilistic monocular 3d human pose estimation with normalizing flows," in *Proc IEEE Int Conf Comput Vision*, 2021, pp. 11 199–11 208.

[15] A. Zanfir, E. G. Bazavan, H. Xu, W. T. Freeman, R. Sukthankar, and C. Sminchisescu, "Weakly supervised 3d human pose and shape reconstruction with normalizing flows," in *Lect. Notes Comput. Sci.* Springer, 2020, pp. 465–481.

[16] 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," *ICLR - Int. Conf. Learn. Represent.*, 2020.

[17] S. Wang, Z. Zhou, and Z. Kan, "When transformer meets robotic grasping: Exploits context for efficient grasp detection," *IEEE Robot. Autom.*, vol. 7, no. 3, pp. 8170–8177, 2022.

[18] J. Li, S. Bian, A. Zeng, C. Wang, B. Pang, W. Liu, and C. Lu, "Human pose regression with residual log-likelihood estimation," in *Proc IEEE Int Conf Comput Vision*, 2021, pp. 11 025–11 034.

[19] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc IEEE Comput Soc Conf Comput Vision Pattern Recognit*, 2016, pp. 770–778.

[20] L. Dinh, D. Krueger, and Y. Bengio, "Nice: Non-linear independent components estimation," *Int. Conf. Learn. Represent., ICLR - Workshop Track Proc.*, 2014.

[21] L. Ardizzone, C. Lüth, J. Kruse, C. Rother, and U. Köthe, "Guided image generation with conditional invertible neural networks," *arXiv*, 2019.

[22] C. Zimmermann and T. Brox, "Learning to estimate 3d hand pose from single rgb images," in *Proc IEEE Int Conf Comput Vision*, 2017, pp. 4903–4911.

[23] J. Zhang, J. Jiao, M. Chen, L. Qu, X. Xu, and Q. Yang, "A hand pose tracking benchmark from stereo matching," in *Proc. Int. Conf. Image Process. ICIP*. IEEE, 2017, pp. 982–986.

[24] Y. Cai, L. Ge, J. Cai, and J. Yuan, "Weakly-supervised 3d hand pose estimation from monocular rgb images," in *Lect. Notes Comput. Sci.*, 2018, pp. 666–682.

[25] D. Kong, Y. Chen, H. Ma, X. Yan, and X. Xie, "Adaptive graphical model network for 2d handpose estimation," *Br. Mach. Vis. Conf. , BMVC*, 2019.

[26] Y. Chen, H. Ma, D. Kong, X. Yan, J. Wu, W. Fan, and X. Xie, "Nonparametric structure regularization machine for 2d hand pose estimation," in *Proc. - IEEE Winter Conf. Appl. Comput. Vis., WACV*, 2020, pp. 381–390.

[27] C. Zimmermann, D. Ceylan, J. Yang, B. Russell, M. Argus, and T. Brox, "Freihand: A dataset for markerless capture of hand pose and shape from single rgb images," in *Proc IEEE Int Conf Comput Vision*, 2019, pp. 813–822.

[28] W. Li, R. Du, and S. Chen, "Keypoint-based contextual representations for hand pose estimation," *Multimedia Tools Appl*, vol. 83, no. 10, pp. 28 357–28 372, 2024.

[29] Y. Wang, B. Zhang, and C. Peng, "Srhandnet: Real-time 2d hand pose estimation with simultaneous region localization," *IEEE Trans Image Process*, vol. 29, pp. 2977–2986, 2019.