Pattern Recognition Letters 174 (2023) 137–144

Contents lists available at ScienceDirect

Pattern Recognition Letters

journal homepage: www.elsevier.com/locate/patrec

Lightweight 3D hand pose estimation by cascading CNNs with reinforcement learning

Mingqi Chen, Shaodong Li *, Feng Shuang, Xi Liu, Kai Luo, Wenbo He

Guangxi Key Laboratory of Intelligent Control and Maintenance of Power Equipment, School of Electrical Engineering, Guangxi University, Nanning 530004, China

ARTICLE INFO

Editor: Wen-Huang Cheng

Keywords:
Hand pose estimation
Feature exploitation
Reinforcement learning
Real-time performance

ABSTRACT

This paper proposes a novel strategy for lightweight 3D hand pose estimation. The strategy decomposes the estimation process into feature extraction and feature exploitation, where feature extraction performs dimension reduction on the original input and outputs feature vectors. Feature exploitation is further analyzed and considered as a path optimization problem, and reinforcement learning (RL) is proved to be capable of tackling the problem accurately. A framework cascading convolutional neural networks (CNNs) and RL is next introduced to validate the effectiveness of the proposed strategy, where two different backbones are used to extract features, and RL is extended into continuous space to enhance accuracy. Ablation studies and experiments are carried out on NYU and ICVL datasets using the proposed strategy with continuous RL. The results show that the accuracy of continuous RL exceeds discrete RL, and the rapidity and accuracy leads the backbones. Comparative studies show the strategy achieves leading rapidity and accuracy in single-view depth-based methods.

1. Introduction

3D hand pose estimation remains a key issue in vision-based human–robot interactions [1,2]. To further enhance estimation accuracy, multi-modal and multi-view fusion are introduced, which may deteriorate real-time performance. Cheng et al. [3] selects proper views to perform estimation and achieves leading accuracy, while its running speed is below 30 fps, which usually does not satisfy real-time usage in high-precision occasions. Therefore, a proper method is required to achieve better real-time performance of 3D hand pose estimation without sacrificing accuracy.

3D hand pose estimation aims at establishing a mapping from the images to 3D joint locations, which is achieved by a two-step process in most existing works [4–6]. The first step is reducing the dimension of input images to low-dimensional features. The second step is refining the features 3D hand joint estimations. Thus, inspired by traditional machine learning, 3D hand pose estimation can be achieved by a decomposed and modularized strategy containing a feature extraction module and a feature exploitation module. The feature extraction module performs dimension reduction and outputs feature maps or vectors from numerous high-dimensional data. The feature exploitation module then refines the feature vectors to corresponding estimations. Better feature extraction enhances feature representation, and ensures accuracy, while feature exploitation takes more effect on running speed. Our aim in this paper is improving real-time performance without losing overall accuracy. Thus, in this paper, CNNs are used to ensure feature extraction accuracy, while a more proper strategy is further discussed to achieve better tradeoff between rapidity and accuracy.

Meanwhile, most existing feature exploitation modules follow bottom-up strategy to estimate 3D joint locations, where ergodic search is performed on all regions of the feature map, as shown in Fig. 1(a). The strategy leads to redundancy on region searching and is usually time-consuming [7]. Top-down strategy is introduced to exploit features in pixel-wise detection issues [8], and is considered to be a better way to improve searching speed. The strategy finds optimal directions sequentially to search the corresponding feature region, while weak-ergodic search needs to be performed in each step to obtain the optimal region proposal, as shown in Fig. 1(b). In fact, both exploitation strategies need CNNs to perform dimensional reduction on feature maps. However, in 3D hand pose estimation, the extracted features can be low-dimensional, and the exploitation process can be considered as sequential point-wise translations of hand joints in pixel or camera coordinates, as shown in Fig. 1(c). Thus, in this paper, the feature exploitation process of 3D hand pose estimation is then considered as a point-wise path optimization issue without ergodic search.

* Corresponding author.
E-mail addresses: 2012401009@st.gxu.edu.cn (M. Chen), lishaodong@gxu.edu.cn (S. Li), fshuang@gxu.edu.cn (F. Shuang), 2112401019@st.gxu.edu.cn (X. Liu), 2212391032@st.gxu.edu.cn (K. Luo), 2212391012@st.gxu.edu.cn (W. He).

https://doi.org/10.1016/j.patrec.2023.09.004
Received 16 December 2022; Received in revised form 13 August 2023; Accepted 8 September 2023
Available online 18 September 2023

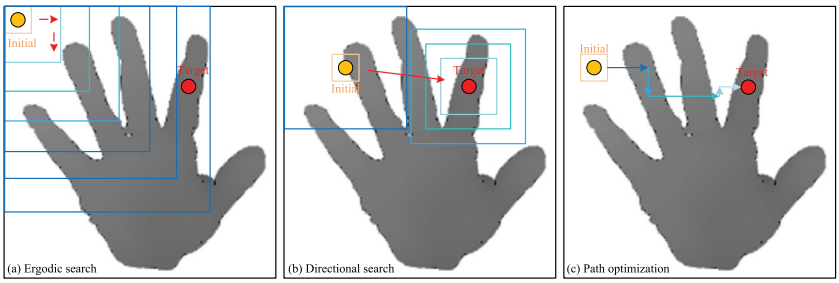


Fig. 1. A brief illustration of our feature exploitation process. Existing exploitation follows bottom-up (a) or top-down (b) strategy, which need ergodic search on the whole feature map or the region proposals. This paper considers the feature exploitation process as a path optimization issue (c) to avoid exhaustive search and further improves running rapidity.

Constraint-based optimization algorithms are usually used to perform path optimization, which are established mainly based on mathematical models such as PSO [9] and ICP [10]. These methods are sensitive to the initial states, and may be difficult to obtain global optimal solution. Meanwhile, constraints are difficult to define considering the point-wise path optimization issue when introducing constraint-based optimization. With the emerge and development of RL, the environment can be explored actively, and the optimal path can be obtained rapidly under various initial states in a model-free manner. Meanwhile, proper strategies have been proposed in RL to avoid local optimum, which ensures stability and accuracy. Thus, RL is a relatively optimal method in feature exploitation to improve real-time performance during testing, although training may be time-consuming to fully explore the environment.

In this paper, we further discuss RL to achieve better tradeoff between accuracy and running rapidity, and a continuous RL algorithm is extended from our previous work [11] in this paper to enhance feature exploitation accuracy. RL has been validated effective in improving running rapidity via a shallow DQN (S-DQN)-based feature exploitation module. The output features of the CNN backbones are used as the initial state of the agent, fixed-step, discretized translations are then performed sequentially via S-DQN to obtain an optimized estimation. The states of S-DQN are updated under the guidance of a reward function based on the variation of estimation error. The cascaded framework of CNNs and S-DQN can effectively avoid iterative refinement in CNN-based backbones, increasing running rapidity with the accuracy being maintained. However, the feature exploitation process of hand pose estimation should be a continuous process. Thus, the accuracy of RL-based feature exploitation module can be further improved by performing translations in continuous action space.

Overall, this paper extends our previous work [11], and proposes a lightweight strategy for rapid 3D hand pose estimation. The strategy cascades CNNs for feature extraction, and deep deterministic policy gradient (DDPG) is introduced to further enhance feature exploitation accuracy, which differs from [11]. Ablation studies and extensive experiments are carried out to show our strategy achieves leading real-time performance and accuracy in single-view depth-based hand pose estimation. The contributions of this paper are listed as follows:

- A lightweight strategy cascading CNNs and RL is proposed to perform rapid 3D hand pose estimation, which decomposes estimation into an extraction–exploitation format. Extraction is seen as a dimension reduction process, and exploitation is considered as path optimization.
- RL is introduced to solve the path optimization problem in feature exploitation, which is proved to be effective in [11]. More importantly, DDPG is introduced to extend the actions into continuous space to further enhance exploitation accuracy.
- Experimental results show that the accuracy of the continuous RL exceeds [11], and leads the backbone on accuracy and real-time performance. Leading rapidity and accuracy are achieved comparing to state-of-the-arts. A supplementary video is also uploaded to show its performance.

The remainder of the paper is organized as follows: Section 2 reviews related works on feature exploitation. Section 3 defines the problem and the cascaded strategy. Section 4 carries out the experiments, and Section 5 concludes the paper.

2. Related works

2.1. Feature exploitation strategies in 3D hand pose estimation

Biomedical constraint-based exploitation is used in first applied in 3D hand pose estimation. Fleishman et al. [10] generate hand pose hypotheses and use ICP to refine the hypotheses to accurate estimation based on inverse kinematics of the hand. Ye et al. [9] use partial PSO to refine the estimation within kinematic constraints. The hand pose features are divided into groups based on hand kinematics, and refined sequentially by partial PSO. However, these methods are sensitive to the initial states, and may be difficult to obtain global optimal solution. Constraints are also used in CNNs to achieve better feature exploitation [12–15]. Spurr et al. [13] introduce biomechanical constraints as loss functions for weakly-supervised hand pose estimation, while complex computation is required to achieve higher accuracy. Avola et al. [14] use mesh-based constraints for feature exploitation, where 2D hand mesh is introduced with 2D projections to improve accuracy. However, the real-time performance of [14] is under 30 fps, which may not satisfy real-time usage under some occasions. Huang et al. [15] propose a transformer-based decoder to exploit hand pose features, where a reference pose is used to exploit the correlations between adjacent hand joints. Recent works on depth-based 3D hand pose estimation usually use fully convolutional networks, where feature exploitation is taken as a refinement process [4,6,16–18]. The feature exploitation module is structurally similar, even identical to the extraction module, and uses the coarse estimations and the original image to perform feature refinement. These methods enhance accuracy, while the computation cost multiplies, and further sacrifices running rapidity. Thus, a parameter-parsimonious strategy is required to enhance rapidity without weakening accuracy.

2.2. RL-based pose estimation

RL is first introduced in object detection to avoid exhaustive search in sliding window methods [19], then transferred into pose estimation [20–24], etc. Shao et al. [20] proposed an RL-based framework to achieve 6D object pose estimation, where an agent is trained to move object in 3D space until the pose offset is below a certain threshold. Gäetner et al. [21] used RL to perform human pose estimation actively by selecting viewpoints and fusing multi-view features in video streams. Sock et al. [22] introduced policy gradient strategy to train an agent similar to that in [21] to achieve 6D object pose estimation. Krull et al. [23] used pose hypothesis to perform 6D pose estimation, and RL is used to select the correct hypothesis. Nie et al. [24] addressed the problem of estimating 3D object pose from 2D images. Q-learning is then proposed to rotate the 3D model of the query object to match

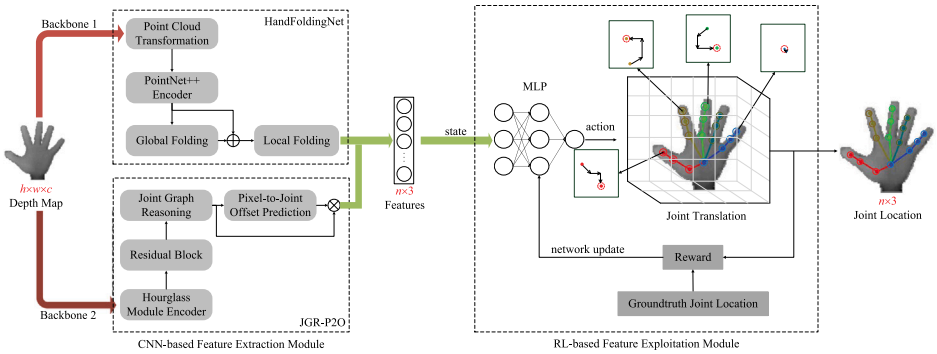


Fig. 2. Architecture of the proposed approach. Feature extraction is seen as a dimension reduction process using CNN, and feature exploitation is considered as a path optimization process in camera coordinates, which is tackled using RL.

its 3D pose. Chen et al. [11] considers feature exploitation as path optimization, which is then tackled using S-DQN for better real-time performance, while its accuracy has limitation due to discrete action space.

3. Methodology

3.1. Problem definition

As mentioned above, the proposed strategy decomposes 3D hand pose estimation into feature extraction and feature exploitation. The original depth image can be represented as $I_{h\times w} = \{P_1, P_2, \ldots, P_i, \ldots, P_{h\times w}\}$, where $P_i = (u_i, v_i, z_i)$ represents the pixel point i in pixel coordinates, h and w denotes the width and height of the image. In fact, pixel points in the depth image can be transformed into 3D points in camera coordinates using the intrinsic parameters of the camera [8] as

$$P'_i = (x_i, y_i, z_i) = \text{Proj}(u_i, v_i, z_i) = \text{Proj}(P_i) \tag{1}$$

where $P'_i = (x_i, y_i, z_i)$ represents point i in camera coordinates, Proj () represents the intrinsic transformation. During feature extraction, the dimension of I is reduced to extract feature vector $\mathbf{f} = \{f_1, f_2, \ldots, f_n\}$ using CNNs, namely $\mathbf{f} = \text{CNN}(I)$, where $n = j\times 3$ denotes the nth feature, and j represents the number of joints. $\mathbf{f}$ is then exploited to hand joint estimations.

Feature exploitation can be considered as a path optimization process. Sequential translation is performed on a randomly initial point, and a path is then given to translate the point towards target in minimal steps. The initial point of joint j can be defined as $P_0^j = (x_0^j, y_0^j, z_0^j)$, and the groundtruth point is $P_{gt}^j = (x_{gt}^j, y_{gt}^j, z_{gt}^j)$. The object of path optimization is to translate P_0^j to a position which is as close as possible to P_{gt}^j. We define the position of joint j at timestep t as $P_t^j = (x_t^j, y_t^j, z_t^j)$. Thus, the object of feature exploitation can be derived as

$$e_{variation} = \left\| P_0^j - P_{gt}^j \right\|_2 - \left\| P_t^j - P_{gt}^j \right\|_2 \le e_{threshold} \tag{2}$$

where $e_{threshold} \ge 0$ is a predefined threshold of error variation, and $\|\cdot\|_2$ is the L2 norm.

It is obvious that the object function shown in Eq. (2) is convex. However, considering sequential translations on a single point, it is difficult to define specific constraints. Thus, it is difficult to model the path optimization process. Tradition optimization algorithms may fall into local optimal, and is not capable of performing efficient feature exploitation.

Given the formulation of the problem, this paper aims at introducing a novel strategy to perform efficient 3D hand pose estimation, where the running speed is improved without sacrificing the overall accuracy. Thus, a novel framework needs to be introduced.

3.2. Architecture of the approach

Following the proposed strategy, the architecture of the proposed approach according to the strategy is shown in Fig. 2, where a CNN-based feature extraction module is cascaded with an RL-based feature exploitation module. The input of the framework can be the original $h \times w \times c$ depth map or 3D point cloud transformed from the depth map, where c represents the channels of the image. CNNs are used to reduce the dimension of the input and outputs $j \times 3$-dimensional feature vector. RL is used to perform model-free path optimization. In the feature exploitation module, the hand joint starts from P_0^j, and is translated sequentially in pixel or camera coordinates. When P_t^j satisfies Eq. (2), the translation is complete. RL is used to obtain an optimal path to achieve Eq. (2) in a model-free manner due to the difficulties of modeling the process. The modules are analyzed in detail in the next section.

3.3. Feature extraction using CNNs

In this paper, the real-time performance of the framework is focused, while improving accuracy can be our future work. Thus, we use two representative frameworks which show leading accuracy on NYU and ICVL datasets to extract features, namely HandFoldingNet [16] and JGR-P2O [17].

HandFoldingNet is a 3D CNN-based hand pose regression framework, where point clouds are transformed from depth maps and used as input. The network employs an encoder–decoder format to extract features using global and multiple local folding blocks. JGR-P2O is a 2D CNN-based hand pose detection framework. The framework extracts and decodes features using a graph convolutional network, and heatmaps are used to obtain features in both pixel and camera coordinates. Note that HandFoldingNet contains multiple local folding blocks, and JGR-P2O introduces multi-stage framework to refine features. The iteration of a single block or a stage limits real-time performance. Thus, in the proposed framework, HandFoldingNet is used with a single local folding block, and one-stage JGR-P2O is used to extract features, we further call these networks as backbones. Moreover, the original frameworks are used in ablation studies to evaluate the performance of the proposed feature exploitation module.

3.4. Feature exploitation using RL

In this paper, we extend the RL-based feature exploitation module from S-DQN [11] into continuous action space to enhance accuracy. DDPG is then used to perform continuous exploitation. In this section, S-DQN-based feature exploitation is first revisited, and the definition of states, actions and reward function of DDPG-based feature exploitation module are defined afterwards.

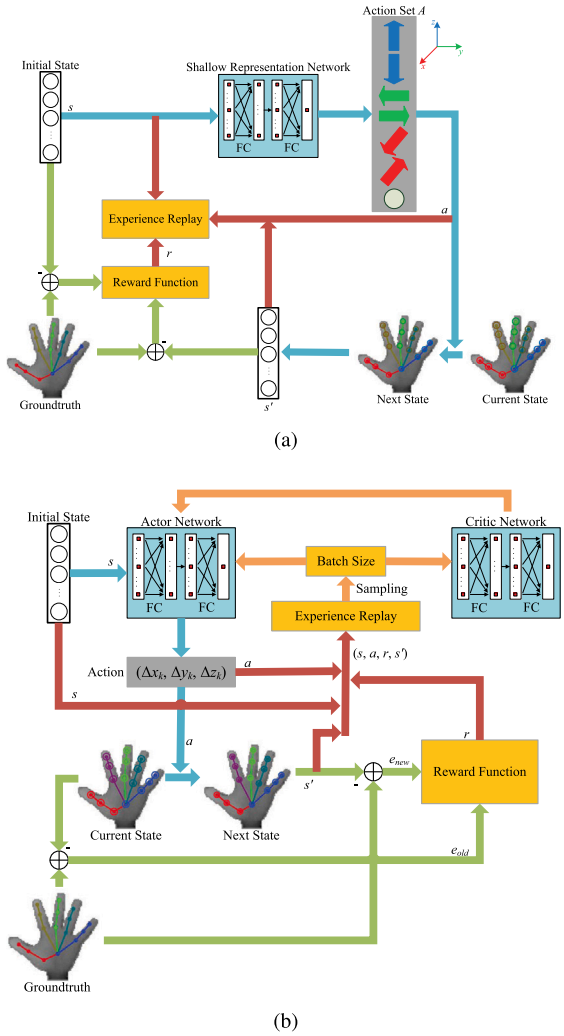


Fig. 3. Working procedure of RL-based feature exploitation module. In S-DQN-based feature exploitation (a), and fixed-step, single direction, discrete translations are made by the agent in each step. While in DDPG-based feature exploitation (b) where $(\Delta x_k, \Delta y_k, \Delta z_k)$ denotes the translation action on point (x_k, y_k, z_k), continuous action space is used, and adaptive 3D translation is achieved in each step, which shows better accuracy. The blue arrows show the feature exploitation process, which is used during testing. The green arrows show the computing of rewards, the red arrows show the storage of experience, and the yellow arrow denotes the training process.

3.4.1. Revisiting S-DQN-based feature exploitation

The working procedure of S-DQN-based feature exploitation module is shown in Fig. 3(a), which can be further referenced in [11] in detail. The output features of the backbones are taken as the initial states of the S-DQN agent. A shallow representation network containing a 2-layer multilayer perceptrons (MLP) is introduced to obtain the optimal direction of translation. Fixed-step translation is then performed by the agent. S-DQN is updated under the guidance of a reward function based on the variation of error. S-DQN significantly increases feature exploitation rapidity, while its accuracy has limitation due to a discrete action space. Thus, DDPG-based module is proposed to extend feature exploitation into continuous space.

3.4.2. DDPG-based feature exploitation module

The working procedure of DDPG-based feature exploitation module is shown in Fig. 3(b), where actor–critic strategy is introduced. In fact, the definition of states, actions and reward function are similar to that in S-DQN. This section analyzes the states, actions and reward function in detail, which also helps understand the working process of S-DQN.

States. The state of the feature exploitation module is defined according to the output of the feature extraction module. In the feature exploitation module, the 3D camera coordinates is used to describe the state space. The input states of the module are the flattened representation of the aforementioned $j \times 3$ matrix. In fact, to ensure stability and accelerate convergence, the initial states are the coarse joint locations output by the feature extraction module.

Actions. The main difference of DDPG and S-DQN is the continuity of the action space. In S-DQN, the estimated joints are fixed-step, and translated in a single direction in each step along X, Y, Z axes in 3D camera coordinates. While in DDPG, a 3D translation vector $(\Delta x_k, \Delta y_k, \Delta z_k)$ is directly given, and the estimated joints can be translated using a learned adaptive step size. Similar to [11], a *stop* action is defined as the terminal action, where the joint keeps its previous location. When the variation of estimation error reaches a certain threshold, a *done* notation is received, and the agent outputs a *stop* action to finish the exploitation process.

Reward function. To achieve accurate exploitation, the reward is set according to the variation of the 3D joint errors. When the error after translation is bigger than the previous error, a penalty of −1 is given, and the variation between the new error and the original error is computed. When the variation between the new error and the original error, the translation is considered to be finished, and a reward of 1 is given. The reward function of each joint can be rewritten as

$$R = \begin{cases} -1, & e_{old} - e_{new} < 0 \\ 1, & e_{ori} - e_{new} > e_{threshold} \\ 0, & \text{else} \end{cases} \tag{3}$$

where the reward is 0 under other circumstances. $e^j_{old} = \left\| P^j_{t-1} - P^j_{gt} \right\|_2$, $e^j_{new} = \left\| P^j_t - P^j_{gt} \right\|_2$, and $e^j_{ori} = \left\| P^j_0 - P^j_{gt} \right\|_2$ denote the estimation errors measured from the input, output, and the original, respectively. Once the variation reaches $e_{threshold}$, the exploitation is finished, a *done* notation is returned with a *stop* action.

4. Experiments

This section carries out experiments to validate the effectiveness of the proposed strategy. The datasets, training mechanism and evaluation metrics used are first introduced. Ablation studies are then performed to evaluate the effectiveness of the proposed framework. Comparisons with state-of-the-art are finally taken evaluate the performance of our work.

4.1. Implementation details

4.1.1. Datasets

NYU [25] and ICVL [26] are used to evaluate the performance of the framework, which are two influential publicly available datasets in depth-based 3D hand pose estimation. NYU contains 72757 training frames and 8252 testing frames from 2 subjects, where 36 joints are annotated. ICVL contains 22084 training frames and 1596 testing frames, with 16 joints being annotated. Following most previous work using the datasets, 16 joints are used in ICVL and 14 joints are used in NYU.

4.1.2. Network training

In this paper, the feature extraction module and the feature exploitation module are trained sequentially. Loss functions are thus defined separately for each module. For feature extraction modules, the loss functions follow the original backbones. The loss of HandFoldingNet is

$$\mathcal{L}_{\text{HandFold}} = \sum_{j=1}^{n} L1_{\text{smooth}}\left(\mathbf{j}^0_j - \mathbf{j}^*_j\right) + \sum_{j=1}^{n} L1_{\text{smooth}}\left(\mathbf{j}^1_j - \mathbf{j}^*_j\right) \tag{4}$$

Table 1
Test result using HandFoldingNet as backbone on ICVL dataset.

Method	Mean error (mm)	# Params	Running time (ms)
Backbone	6.34	0.78M	4.06
Backbone + Original	5.95	1.28M	4.82
Backbone + S-DQN [11]	5.95	0.78M	4.43
Backbone + DDPG	**5.90**	**1.02M**	**4.67**

Table 2
Test result using JGR-P2O as backbone on NYU dataset.

Method	Mean error (mm)	# Params	Running time (ms)
Backbone	8.63	0.72M	4.79
Backbone + Original	8.29	1.37M	6.04
Backbone + S-DQN [11]	8.34	0.72M	5.16
Backbone + DDPG	**8.32**	**0.96M**	**5.46**

where $\mathbf{j}_j^0$ and $\mathbf{j}_j^1$ represent the estimated locations of the jth joint of each folding block in the network, and $\mathbf{j}_j^*$ denotes its groundtruth location. Meanwhile, the loss of JGR-P2O is

$$\mathcal{L}_{\text{JGR-P2O}} = \sum_k \sum_c \mathcal{L}_\delta \left(c_{jk} - c_{jk}^* \right) + \beta \sum_k \sum_i \sum_c \mathcal{L}_\delta \left(\Delta c_{ki} - \Delta c_{ki}^* \right) \tag{5}$$

where c_{jk} is the predicted position of joint k, and c_{jk}^* is the corresponding groundtruth. L_δ denotes the Huber loss function. Δc_{ki} represents the offset value from pixel p_i to joint k along pixel coordinates, and Δc_{ki}^* is the corresponding groundtruth, $\beta = 0.0001$ is a balancing weight factor. The loss of the feature extraction module can be further referenced in [16,17].

DDPG-based feature exploitation module follows actor–critic strategy in training, where the networks of DDPG consist 3-layer MLPs. Target networks are also introduced in training. The parameters of the critic network is updated as

$$\mathcal{L}_{Critic} = \begin{cases} r & \text{if } done \\ \| r + \gamma \max_a Q(s', a; \theta^-) - Q(s, a; \theta) \|_2 & \text{else} \end{cases} \tag{6}$$

where θ and θ^- represent the parameters of the evaluation network and the target network, respectively. Meanwhile, the actor network of DDPG updates by backpropagating the gradient of the value function w.r.t the actions output by the actor network following

$$\nabla_{\theta^\pi} J \approx \mathbb{E}_\pi \left[\nabla_a Q \left(s, a \mid \theta^Q \right) \nabla_{\theta_\mu} \mu \left(s \mid \theta^\mu \right) \right] \tag{7}$$

where θ^Q and θ^μ represents the parameters of the critic network and actor network, respectively. $\mu(s \mid \theta)$ represents the actor network. Thus, an optimal policy following Eq. (2) can then be obtained, and the feature exploitation can be achieved efficiently.

4.1.3. Settings and evaluation metrics

The experiments are performed on a single NVIDIA TITAN V GPU using PyTorch. For feature extraction modules, images are preprocessed following the original methods. To achieve fair comparison with the backbones, the hyperparameters are also set following the original works in [16,17].

The size of experience replay in the feature exploitation module is 10^6. The batch size is 10^4, and the discount factor γ is 0.99, τ is 0.005 to achieve soft update of the target network. The learning rate α is 10^{-4} in the actor network, and is 10^{-3} in the critic network. The standard step size of feature exploitation is set according to the output of the backbones, which is 0.005 for HandFoldingNet and 1.5 for JGR-P2O. To achieve better accuracy, the states and actions are equally scaled by 100. In addition, an exploration noise of is used to enhance data richness, which is 0.2 for HandFoldingNet and 20 for JGR-P2O after scaling.

3D mean error and success rate are used to evaluate the accuracy. 3D mean error is calculated by averaging the Euclidean distance between the estimated joints and groundtruth. Success rate denotes the percentage of success frames in which the worst joint 3D distance error is below a threshold. Meanwhile, running speed is introduced to evaluate the rapidity of the framework, which is obtained by computing the per sample running time during testing, and fps is further computed to achieve fair comparison with state-of-the-arts.

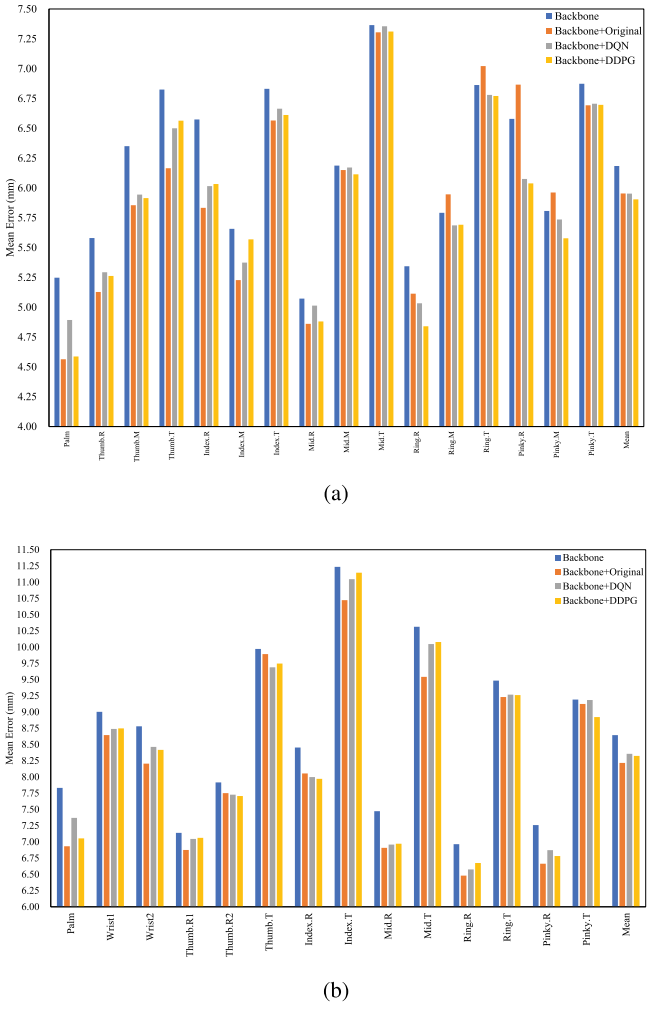
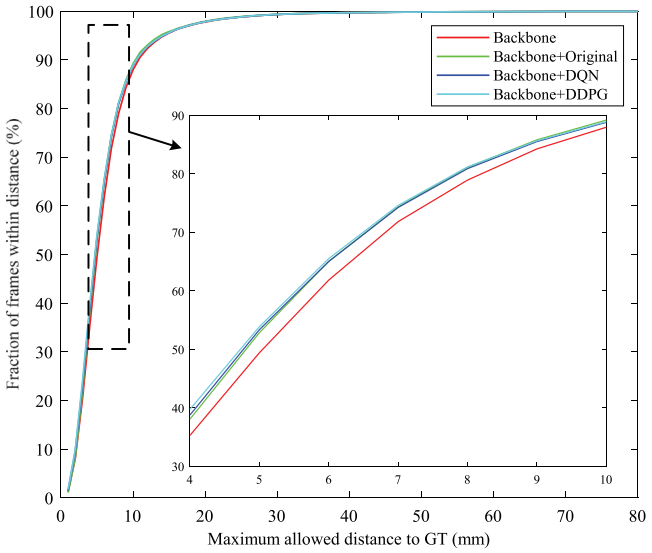


Fig. 4. Joint-wise mean 3D error based on (a) HandFoldingNet on ICVL and (b) JGR-P2O on NYU.

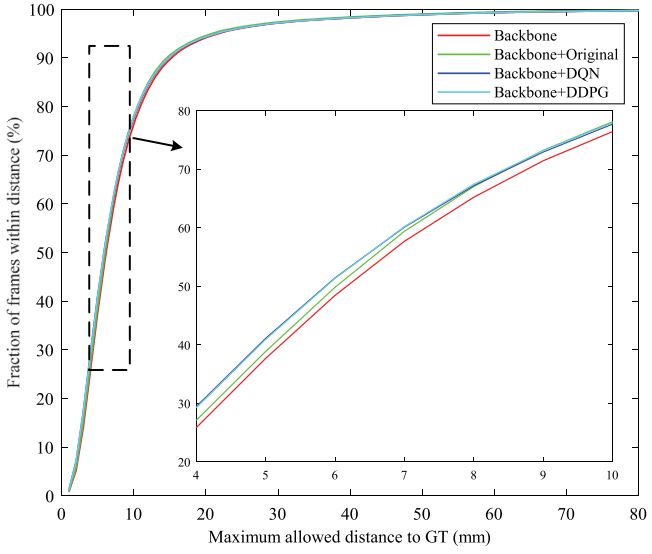
4.2. Ablation studies

Ablation studies are first performed to validate the effectiveness of the proposed strategy. The proposed framework is compared with the coarse output of the backbone, the refined output by iterating the backbone, and the refined output by S-DQN. To achieve fair comparison with the backbones, ICVL is used in HandFoldingNet, and NYU is used in JGR-P2O. The results are listed in Tables 1 and 2.

From the tables above, the 3D mean error is decreased by 0.47 mm and 0.31 mm comparing to the backbones, respectively, which means the DDPG-based module achieves feature exploitation. The accuracy exceeds S-DQN, even surpass the original HandFoldingNet. To evaluate the accuracy in detail, joint-wise mean 3D error and success rate are shown in Figs. 4 and 5. It is obvious in Fig. 4 that errors on all joints are compensated via DDPG comparing to the backbones. The average decrease reaches 0.66 mm and 0.78 mm using the proposed strategy, respectively, which is better than S-DQN. Meanwhile, the success rate



(a)



(b)

Fig. 5. Success rate based on (a) HandFoldingNet on ICVL and (b) JGR-P2O on NYU.

under 5 mm is increased by 4.48% and 3.24% via DDPG, and the success rate under 10 mm is increased by 0.93% and 1.51% on both datasets, which also exceeds S-DQN, as shown in Fig. 5. In summary, the proposed feature exploitation module using continuous RL shows enhancement on accuracy comparing to discrete RL, and can generalize on different backbones and datasets.

More importantly, the proposed strategy still shows improvement on running rapidity comparing to the backbones. The average per sample running time 0.64 ms for DDPG-based feature exploitation module, and the per sample running time is reduced by 0.15 ms and 0.58 ms comparing to iterating backbones, which further means the proposed strategy can save 3.11% and 9.60% on each backbone, respectively. In fact, only the actor network is used in DDPG during testing. The actor contains 0.24M parameters, whose number is significantly decreased comparing to the original refinement blocks. The running time sacrifices slightly using DDPG comparing to S-DQN, while the accuracy increases. As a result, DDPG-based feature exploitation module receives higher real-time performance comparing to iterating backbones, and shows better tradeoff between accuracy and rapidity comparing to S-DQN.

Table 3
Comparisons with state-of-the-arts on NYU and ICVL datasets, where the bold items are the proposed strategy. The red items shows the running speed of the backbones in fps on our own platform, while others follows their original works.

Method	Mean error (mm)		# Params	Speed (fps)	Input type
	NYU	ICVL			
DeepModel [12]	17.04	11.56	–	–	2D
REN-4×6×6 [27]	13.39	7.63	–	–	2D
REN-9×6×6[27]	12.69	7.31	–	–	2D
DeepPrior++ [5]	12.24	8.10	–	–	2D
Pose-REN [8]	11.81	6.79	0.72M	5.16	2D
DenseReg [18]	10.20	7.30	5.8M	27.8	2D
CrossInfoNet [4]	10.08	6.73	23.8M	124.5	2D
A2J [28]	8.61	6.46	44.7M	105.1	2D
JGR-P2O [17]	8.29	6.02	1.4M	111.2	2D
Zhang et al.[29]	9.21	6.67	–	5	2D
SRNet [6]	9.17	6.15	3.3M	–	2D
AWR [30]	7.18	5.98	460M		2D
JGR-P2O + S-DQN [11]	8.34	–	0.72M	139.7	2D
JGR-P2O + DDPG	**8.32**	–	**0.96M**	**134.0**	**2D**
3D CNN [31]	14.1	–	–	215	3D
V2V-PoseNet [32]	8.42	6.28	457.5M	3.5	3D
SHPR-Net [33]	10.78	7.22	–	–	3D
HandPointNet [34]	10.54	6.94	2.58M	48.0	3D
HandFoldingNet [16]	8.58	5.95	1.28M	76.8	3D
HandFoldingNet + S-DQN [11]	–	5.95	0.78M	79.2	3D
HandFoldingNet + DDPG	–	**5.90**	**1.02M**	**77.7**	**3D**

Fig. 6. Qualitative results on ICVL dataset, where the groundtruth, estimation based the backbone and our strategy are visualized. The left three columns show accurate cases, while the right three columns show "failure cases".

4.3. Comparisons with state-of-the-art

Comparisons are made with state-of-the-arts on single-view depth-based 3D hand pose estimation to evaluate the performance on NYU and ICVL. The results are listed in Table 3, where state-of-the-arts are divided into two categories according to the types of input, namely 2D-based methods and 3D-based methods.

From Table 3, the proposed strategy shows the best accuracy on ICVL, and ranks third on NYU, which means the proposed strategy achieves leading accuracy in single-view depth-based 3D hand pose estimation. Meanwhile, the number of parameters in the proposed framework is relatively smaller in Table 3, which leads to a better real-time performance comparing to state-of-the-arts. The running speed of the proposed frameworks are computed after adding a preprocessing time of the backbones, which is 8.2 ms on HandFoldingNet and 2.0 ms on JGR-P2O. DDPG achieves second best running speed in 2D-based methods cascading JGR-P2O, and ranks third in 3D-based methods cascading HandFoldingNet. Note that the listed running speed of the backbones are computed on our own platform, which differs slightly from the original works. Thus, the proposed strategy achieves leading performance on both accuracy and real-time performance, which validates its superiority. A supplementary video is also uploaded to further examine the performance of the proposed strategy.

Qualitative results are shown in Fig. 6 to further evaluate the performance of the proposed strategy, where the test set of ICVL is used as an example. From the left three columns, the proposed strategy shows closer joint locations to the groundtruth comparing to the backbone, which validates its leading accuracy. Some hand poses regressed by the proposed strategy are even better than the groundtruth, as shown in the third column. However, "failure cases" still exist, as shown in the right three columns. self-occlusion leads to most of the "failure cases", which affects the initial states of the DDPG-based exploitation module severely, as coarse estimations from backbones are taken as initial states. Meanwhile, all joints are refined using an identical standard step size in DDPG, which may not be suitable for joints with larger errors.

In summary, the proposed framework shows good performance on running rapidity, and the accuracy leading most existing works on single-view frameworks, while "failure cases" should be further analyzed and discussed in the future.

5. Conclusions

In this paper, a cascaded lightweight strategy for 3D hand pose estimation is proposed following an extraction–exploitation strategy. Feature extraction follows CNN to ensure accuracy, and feature exploitation is formulated as path optimization, where DDPG is introduced to exploit features in continuous action space. The proposed strategy is validated using multiple backbones on different publicly available datasets in the experiments. Ablation studies show that the mean 3D errors are decreased by 0.66 mm and 0.78 mm on ICVL and NYU, respectively. Success rates under 5 mm are also increased by 4.48% and 3.24%. The feature exploitation module based on continuous RL not only shows better accuracy comparing to S-DQN, the running time is but also saved by 3.11% and 9.60% on average. Comparisons are also made with state-of-the-arts to further evaluate the performance of the proposed strategy. The results show the proposed strategy achieves leading performance on both accuracy and real-time performance, which further validates the superiority of the proposed strategy.

The proposed strategy regresses better hand poses comparing to backbones, even better than groundtruth, while self-occlusions are the main cause of some larger errors. Future work will introduce multi-modal features and novel exploitation methods such as multi-task learning to obtain better performance.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Shaodong Li reports financial support was provided by Guangxi Natural Science Foundation. Shaodong Li reports financial support was provided by Middle-aged and Young Teachers' Basic Ability Promotion Project of Guangxi. Feng Shuang reports financial support was provided by Bagui Scholars Program of Guangxi Zhuang Autonomous Region.

Data availability

The data used in this paper are already publicly available.

Acknowledgments

This work was supported in part by the Guangxi Natural Science Foundation, China (Grant No. 2022JJB170009), and in part by Middle-aged and Young Teachers' Basic Ability Promotion Project of Guangxi, China (Grant No. 2022KY0008). Feng acknowledges support by the Bagui Scholars Program of Guangxi Zhuang Autonomous Region, China.

Appendix A. Supplementary data

Supplementary material related to this article can be found online at https://doi.org/10.1016/j.patrec.2023.09.004.

References

[1] Q. Fan, X. Shen, Y. Hu, C. Yu, Simple very deep convolutional network for robust hand pose regression from a single depth image, Pattern Recognit. Lett. 119 (2019) 205–213.
[2] F.A. Kondori, S. Yousefi, J. Kouma, L. Liu, H. Li, Direct hand pose estimation for immersive gestural interaction, Pattern Recognit. Lett. 66 (2015) 91–99.
[3] J. Cheng, Y. Wan, D. Zuo, C. Ma, J. Gu, P. Tan, H. Wang, X. Deng, Y. Zhang, Efficient virtual view selection for 3D hand pose estimation, in: Proceedings of the AAAI Conference on Artificial Intelligence, 2022, pp. 419–426.
[4] X. Zhang, F. Zhang, Differentiable spatial regression: A novel method for 3D hand pose estimation, IEEE Trans. Multimed. 24 (2022) 166–176.
[5] K. Du, X. Lin, Y. Sun, X. Ma, Crossinfonet: Multi-task information sharing based hand pose estimation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2019, pp. 9896–9905.
[6] M. Oberweger, V. Lepetit, DeepPrior++: Improving fast and accurate 3D hand pose estimation, in: Proceedings of the IEEE International Conference on Computer Vision Workshops, 2017, pp. 585–594.
[7] Y. Wang, L. Zhang, L. Wang, Z. Wang, Multitask learning for object localization with deep reinforcement learning, IEEE Trans. Cogn. Dev. Syst. 11 (4) (2018) 573–580.
[8] X. Chen, G. Wang, H. Guo, C. Zhang, Pose guided structured region ensemble network for cascaded hand pose estimation, Neurocomputing 395 (2020) 138–149.
[9] Q. Ye, S. Yuan, T.-K. Kim, Spatial attention deep net with partial PSO for hierarchical hybrid hand pose estimation, in: Proceedings of the European Conference on Computer Vision, 2016, pp. 346–361.
[10] S. Fleishman, M. Kliger, A. Lerner, G. Kutliroff, ICPIK: Inverse kinematics based articulated-ICP, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops, 2015, pp. 28–35.
[11] M. Chen, S. Li, F. Shuang, K. Luo, Cascading CNNs with S-DQN: A parameter-parsimonious strategy for 3D hand pose estimation, in: MultiMedia Modeling: 29th International Conference, MMM 2023, Bergen, Norway, January 9–12, 2023, Proceedings, Part I, 2023, pp. 358–369.
[12] X. Zhou, Q. Wan, W. Zhang, X. Xue, Y. Wei, Model-based deep hand pose estimation, in: Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, 2016, pp. 2421–2427.
[13] A. Spurr, U. Iqbal, P. Molchanov, O. Hilliges, J. Kautz, Weakly supervised 3D hand pose estimation via biomechanical constraints, in: Proceedings of the European Conference on Computer Vision, 2020, pp. 211–228.
[14] D. Avola, L. Cinque, A. Fagioli, G.L. Foresti, A. Fragomeni, D. Pannone, 3D hand pose and shape estimation from RGB images for keypoint-based hand gesture recognition, Pattern Recognit. 129 (2022) 108762.
[15] L. Huang, J. Tan, J. Liu, J. Yuan, Hand-Transformer: Non-autoregressive structured modeling for 3D hand pose estimation, in: Proceedings of the European Conference on Computer Vision, 2020, pp. 17–33.
[16] W. Cheng, J.H. Park, J.H. Ko, HandFoldingNet: A 3D hand pose estimation network using multiscale-feature guided folding of a 2D hand skeleton, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2021, pp. 11260–11269.
[17] L. Fang, X. Liu, L. Liu, H. Xu, W. Kang, JGR-P2O: Joint graph reasoning based pixel-to-offset prediction network for 3D hand pose estimation from a single depth image, in: Proceedings of the European Conference on Computer Vision, 2020, pp. 120–137.
[18] C. Wan, T. Probst, L. Van Gool, A. Yao, Dense 3D regression for hand pose estimation, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 5147–5156.
[19] J.C. Caicedo, S. Lazebnik, Active object localization with deep reinforcement learning, in: Proceedings of the IEEE International Conference on Computer Vision, 2015, pp. 2488–2496.
[20] J. Shao, Y. Jiang, G. Wang, Z. Li, X. Ji, PFRL: Pose-free reinforcement learning for 6D pose estimation, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 11454–11463.
[21] E. Gärtner, A. Pirinen, C. Sminchisescu, Deep reinforcement learning for active human pose estimation, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 34, (07) 2020, pp. 10835–10844.
[22] J. Sock, G. Garcia-Hernando, T.-K. Kim, Active 6D multi-object pose estimation in cluttered scenarios with deep reinforcement learning, in: 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems, (IROS), 2020, pp. 10564–10571.
[23] A. Krull, E. Brachmann, S. Nowozin, F. Michel, J. Shotton, C. Rother, PoseAgent: Budget-constrained 6D object pose estimation via reinforcement learning, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 6702–6710.

[24] W.-Z. Nie, W.-W. Jia, W.-H. Li, A.-A. Liu, S.-C. Zhao, 3D pose estimation based on reinforce learning for 2D image-based 3D model retrieval, IEEE Trans. Multimed. 23 (2020) 1021–1034.
[25] J. Tompson, M. Stein, Y. Lecun, K. Perlin, Real-time continuous pose recovery of human hands using convolutional networks, ACM Trans. Graph. 33 (5) (2014) 1–10.
[26] D. Tang, H. Jin Chang, A. Tejani, T.-K. Kim, Latent regression forest: Structured estimation of 3D articulated hand posture, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2014, pp. 3786–3793.
[27] G. Wang, X. Chen, H. Guo, C. Zhang, Region ensemble network: Towards good practices for deep 3D hand pose estimation, J. Vis. Commun. Image Represent. 55 (2018) 404–414.
[28] F. Xiong, B. Zhang, Y. Xiao, Z. Cao, T. Yu, J.T. Zhou, J. Yuan, A2J: Anchor-to-joint regression network for 3D articulated pose estimation from a single depth image, in: Proceedings of the IEEE/CVF International Conference on Computer Vision, 2019, pp. 793–802.
[29] Y. Zhang, S. Mi, J. Wu, X. Geng, Simultaneous 3D hand detection and pose estimation using single depth images, Pattern Recognit. Lett. 140 (2020) 43–48.
[30] W. Huang, P. Ren, J. Wang, Q. Qi, H. Sun, AWR: Adaptive weighting regression for 3D hand pose estimation, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 34, (07) 2020, pp. 11061–11068.
[31] L. Ge, H. Liang, J. Yuan, D. Thalmann, 3D convolutional neural networks for efficient and robust hand pose estimation from single depth images, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2017, pp. 1991–2000.
[32] G. Moon, J.Y. Chang, K.M. Lee, V2V-PoseNet: Voxel-to-voxel prediction network for accurate 3D hand and human pose estimation from a single depth map, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 5079–5088.
[33] X. Chen, G. Wang, C. Zhang, T.-K. Kim, X. Ji, SHPR-Net: Deep semantic hand pose regression from point clouds, IEEE Access 6 (2018) 43425–43439.
[34] L. Ge, Y. Cai, J. Weng, J. Yuan, Hand PointNet: 3D hand pose estimation using point sets, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 8417–8426.