Embedding Gesture Prior to Joint Shape Optimization Based Real-Time 3D Hand Tracking

YUNLONG CHE\(^{1}\) AND YUE QI\(^{1,2,3}\), (Member, IEEE)

\(^{1}\)State Key Laboratory of Virtual Reality Technology and Systems, Beihang University, Beijing 100191, China
\(^{2}\)Peng Cheng Laboratory, Shenzhen 518000, China
\(^{3}\)Qingdao Research Institute, Beihang University, Qingdao 266100, China

Corresponding author: Yue Qi (qy@buaa.edu.cn)

This work was supported in part by the National Key Research and Development Program of China under Grant 2017YFB1002602, in part by the National Natural Science Foundation of China under Grant 61572054, and in part by the Leading Talents in Innovation and Entrepreneurship of Qingdao.

ABSTRACT In this paper, we present a novel approach for 3D hand tracking in real-time from a set of depth images. In each frame, our approach initializes hand pose with learning and then jointly optimizes the hand pose and shape. For pose initialization, we propose a gesture classification and root location network (GCRL), which can capture the meaningful topological structure of the hand to estimate the gesture and root location of the hand. With the per-frame initialization, our approach can rapidly recover from tracking failures. For optimization, unlike most existing methods that have been using a fixed-size hand model or manual calibration, we propose a hand gesture-guided optimization strategy to estimate pose and shape iteratively, which makes the tracking results more accuracy. Experiments on three challenging datasets show that our proposed approach achieves similar accuracy as state-of-the-art approaches, while runs on a low computational resource (without GPU).

INDEX TERMS Hand pose estimation, deep learning, articulated tracking, hand tracking.

I. INTRODUCTION

Hand motion tracking has become increasingly important in fields of computer vision, human-computer interaction, and virtual/augmented reality (V/AR) [9] in recent years. It is mostly used to interact with virtual objects and environments. In the early years of hand tracking, there were many color-based methods [9], but they are far more challenging, because they estimate the 3D hand pose from an RGB image, which is an ill-posed problem due to depth ambiguities. Recently, many deep-based approaches [1], [4] have been proposed for human body key-point detection from color image, and they have been successfully applied to hand pose estimation [2], [13], [30], [47]. They localize the key-points in the 2D image via per-pixel predictions [26], [45]. However, creating heat maps for key-points detection requires higher computations. In addition, the ‘3D’ pose obtained from the color image is not real 3D, it [16] usually consists of 2D coordinates of the key-points in the input image, and scale normalized depth for each key-point relative to the hand center. Most user interfaces of V/AR applications require the hand location in the real world for input. It also limits the usage scenarios of color-based methods.

With the launch of depth sensors such as Intel RealSense and Kinect V2, alternatively, purpose-designed hardware, such as the leap motion, many depth-based methods have been proposed to solve such a challenging problem. Due to the highly articulation of hand, these methods still suffer from the following problems: (i) Fingers are easily hidden by each other(as shown in Figure.1). (ii) Various hands may have various shapes. (iii) Fast hand motion may cause severe depth noise(flying pixels or contour noise).

Recently, many modern hand tracking systems have been proposed to accomplish such a task. Typically, they use deep learning for pose initialization and followed by model-based pose optimization. For the pose initialization, although many approaches achieve high accuracy for joint regression or per-pixel semantic segmentation, they require complex network structures. For pose optimization, they usually use a fixed-size hand model and may fail to reach a minimum.
Moreover, most of these approaches rely on high computational resources or are unable to achieve real-time performance.

Based on the above considerations, we propose a novel approach that embeds hand gesture prior to joint shape optimization to accomplish such a task.

Compared to previous works, our proposed approach has several contributions. We propose a GCRL network for pose initialization, which achieves a robust and efficient result compare with directly regressing the key-points or per-pixel hand part classification. The results of the GCRL network are treated as an initial solution for iterative optimization and prevent tracking loss caused by self-occlusion or fast hand motion. Additionally, based on the GCRL network, we design a progressive method that constructs the initial solution from inferred gesture and previous hand shape. This strategy increases the stability of the tracking result. We propose a gesture-guided optimization to estimate the pose and shape of the input hand. During the model fitting, we only optimize the visible bone of hand (which depends on current hand gesture). This may reduce the computations of jointly tracking the pose and shape. In effect, we seamlessly combine the GCRL network and gesture-guided optimization to construct a pipeline (as shown in Figure 2). We conduct a set of experiments on three public datasets to evaluate our approach.

The remainder of this paper is structured as follows: In section 2, we discuss the related works on hand tracking. In section 3, we describe the GCRL network, hand pose/shape model used in our work and real-time tracking approach. In section 4, we analyze the performance of the hand tracking approach and provide comparisons to the state-of-the-art approaches. Finally, we conclude the paper with a brief discussion and a conclusion in section 5.

II. RELATED WORK

Depth-based hand tracking is the subject of a large number of published works [11], [15], [21], [22], [33], [37], [46], [49], in computer vision or human-computer interaction. Current work can be grouped into discriminative, generative, and hybrid methods.

Discriminative methods [48] mostly tend to learn a function from large quantity of training data, and directly map features to corresponding hand poses. Ge et al. [12] fed a set of hand parts rendered from different views into a 2D CNN for pose regression. Zhou et al. [50] embedded the joint forward kinematic process into a regression network to improve the accuracy. In [28], Oberweger et al. compared several different CNN architectures and chose the best one to estimate hand joint locations. These methods attempt to
estimate the hand joint configuration directly and usually do not use temporal information and other priors.

**Generative methods** synthesize observations that are compared to inputs. Then, through optimization, the hand pose that most accurately matches the observation is identified. Oikonomidis et al. [29] fit a hand (spheres/cylinders model) to the depth image with particle swarm optimization (PSO). Qian et al. [31] used an ICP-PSO method to find the possible pose that matches the observed hand point cloud. Sharp et al. [34] adopted a mesh model for tracking in a render-and-compute framework and achieved efficient performance. Melax et al. [24] used rigid body dynamics (RBD) to fit the input depth image and achieves robust results. Generative methods usually focus on more efficient methods for adaptive hand models, model representations, and optimization strategies. They are more computationally complex than discriminative approaches and prone to local minima due to the fast hand motion and complex articulation structure.

These two classes of methods have had a slight overlap over the last years. **Hybrid methods** combine discriminative and generative methods to improve the robustness and preciseness of model fitting with per-frame pose reinitialization. Tompson et al. [43] first estimated the 2D joint’s location with a heatmap and then refined the alignment with PSO-based model fitting. Another type of hybrid algorithm replaces the correspondences found by the iterative closest point (ICP) with the result of a per-pixel classifier (Random Forest (RF) or Convolutional Neural Network (CNN)). Sridhar et al. [35] optimized the hand pose using a detection guide strategy. They achieved correspondences through a per-pixel random forest classifier. These point-to-model correspondences can guide the fitting process.

In a modern motion tracking system, model calibration plays a core role. Unlike face or body model, the large number self-occlusions of hand, the heavy noise of depth sensors, and globally unconstrained pose, and, the self-occlusion of fingers makes the hand shape calibration harder. Current work of hand calibration mainly includes offline and online calibration.

**Offline calibration.** Taylor et al. [40] generated hand models from input depth frames while the user needs to rotate his fingers manually. Tan et al. [39] personalized a hand model from a set of depth measurements using a trained shape basis and achieved a robust result. Remelli et al. [32] calibrated a hand sphere-mesh in a low dimensional shape-space with a multi-stage optimization.

**Online calibration.** de La Gorce et al. [7] tracked hand from a set of RGB images with a model with preset hand shape (in the first frame). Makris and Argyros [23] proposed a model-based approach to jointly solve the pose and shape estimation problem in a tracking system. They fit a set of frames with a fixed-size cylindrical model of a hand, it runs in 30 fps, but the render-and-compute framework limits future improvement. Tkach et al. [42] used an online optimization algorithm that jointly estimates pose and shape in each frame. Moreover, it is more robust and precise than other algorithms. It leverages the estimated shape parameter confidence and builds a tracking sphere model. However, their approach needs to run on high computing resources (GPUs).

Compared to offline methods, Online algorithms can offer immediate feedback to the user and have the potential to adapt to hands with different shapes dynamically. More importantly, they will improve the accuracy of the hand tracking systems.

In recent years, most works in this direction [12], [48], [50] have been based on end-to-end networks. However, existing deep-based approaches mainly have the following problems. Most of them require expensive computing resources (GPUs) to achieve real-time performance, which limits the scope of use of these methods. Existing networks improve the generalization of hands with different shapes only by normalizing the input data or adding training data. Moreover, in some scenarios of virtual/augmented reality, like interacting with deformed objects, the application requires more details of the user’s hand, such as the hand-to-object contact position on hand, and only estimating the hand joint locations is not enough.

**III. OUR APPROACH**

**A. OVERVIEW**

In this paper, we denote pose \( \theta \) and shape \( \beta \) as \([\theta, \beta]\). The goal of our hand pose and shape tracking is to find a hand model with parameter \([\theta, \beta]\) that best matches the input depth image, and the \([\theta, \beta]\) must be plausible and realistic. We propose a regularized hand gesture-guided optimization that carefully balances data fitting with suitable priors limits to solve the problem. Our data fitting term contains both 3D point-model and 2D silhouette constraints. The 3D point to model constraint ensures that each depth point is close to the tracked model, it will pull the hand model to depth points. The 2D silhouette constraint pushes the tracked model into the depth’s contour. Additionally, we adopt a set of hand priors to ensure that the recovered \( \theta \) corresponds with the constraints of reality. Above all, we aim to solve the following optimization problem

\[
[\hat{\theta}, \hat{\beta}] = \arg \min_{\theta, \beta} (E_{data}(\theta, \beta) + E_{prior}(\theta))
\]  

(1)

Recently, learning-based initialization has been widely used in hand tracking. These methods attempt to guess a coarse \( \theta_{\text{init}} \), which is around the global minimum of \( \theta \). It avoids trapping in multiple local minima and improves convergence speed when using the \( \theta_{\text{init}} \) as the start ‘seed’. We adopt a similar strategy, our approach comprises per-frame initialization and model fitting. We first pre-process the input depth image (including hand interesting area image (ROI) cropping, noise filtering, and 2D silhouette extraction). The hand ROI is then fed into a trained GCRL network to estimate the hand gesture and hand root location, thus providing an ‘initialization’. Finally, starting with the initial pose, we optimize \( x(\theta_{\text{init}}, \beta) \) until it is cover with both data fitting and prior limits.
The following sections describe these components. We begin with our GCRL network and pose/shape model, and then describe the definition of our energy function and optimization strategy.

**B. INITIALIZATION**

Due to the highly non-convex of the object function, the existing approach uses reinitializer like multi-layer random forests [20], convolution neural networks [43] or fingertip detection [31] for search a single good pose solution. In [34], Sharp et al. realized that it is hard to predict a single good pose solution. They designed a two-layer tree-based hand pose reinitialization to predict a distribution over poses. The first layer exclusively focuses on predicting global hand rotation. The second layer is trained to infer finger rotations and other elements. In [40], Taylor et al. used retrieval forest to search four postures that are closest to the real value, and this strategy achieved robust results. Inspired by this approach, we believe that the result of the reinitializer should remain stable in sequence. Furthermore, hand gestures can be treated as a label that represents a set of discrete predetermined posture vectors [8], [19].

Based on the above considerations, we split the pose reinitialization problem into two parts: gesture/pose classification and root position location. The proposed GCRL network solves two sub-tasks: gesture classification and root position location.

The root location stream regresses the per-pixel likelihood heatmap for the hand root, so we can effectively estimate the global position of the hand. It has an encoder and a corresponding decoder, followed by a pixel-wise likelihood heatmap layer. As shown in Figure 4. First, we use three convolution(CONV) layers(each layer follows a max-pooling layer) to down-sample the input, and four convolution layers to capture the low-level image features. Then we perform two unpooling operations between convolutions to up-sample the given depth features to a heatmap of the hand root. The unpooling process balances computational time and accuracy. The per-pixel likelihood of the heatmap is computed as follows:

$$L = \sum_{n=1}^{N} \sum_{i,j,k} ||R_{\text{gt}}(i, j, k) - R_{c}(i, j, k)||$$  \hspace{1cm} (2)

where $R_{\text{gt}}$ and $R_{c}$ are the ground-truth and estimated heatmap for the hand root.

The gesture classification stream identifies the category of the gesture. In our experiment, we classify the gesture by employing intermediate features of the hand. The classification branch has five additional convolutions and two fully connected layers, the output probabilities of 17 classes. As shown in Figure 3, each gesture corresponds to a hand pose vector $\theta_{\text{gesture}}$, from the $\theta_{\text{gesture}}$ and global pose(orientation is the direction of the point cloud, position is estimated from root location sub-network), and we can restore the current hand pose $\theta_{\text{initial}}$.

The architecture of the GCRL network is shown in Figure 4. We train the GCRL network by minimizing the total training loss:

$$L_{\text{total}} = \lambda_{\text{cls}}L_{\text{cls}} + \lambda_{\text{loc}}L_{\text{loc}}$$  \hspace{1cm} (3)

The GCRL network can provide an initial $\theta_{\text{initial}}$ for the subsequent optimization phase. In addition, it is used to guide the optimization, to be specific. We optimize only the visible bones(which usually means high confidence of $\beta$) based on
the predicted gesture. This reduces the number of optimized parameters.

There are two types of initialization, the first directly estimates joints’ locations and uses inverse kinematics (IK) to calculate the rotation angle value of joints, and the second performs per-pixel hand part segmentation, then guides the model fitting with the semantic information. The former may not generalize well for different hand shapes, the latter is slow and violates the inherent topology structure. When tracking is lost, our GCRL network first infers the hand gesture, then, we construct a new initial seed from the last tracked frame’s hand shape and the inferred initial posture. This strategy makes our GCRL network robust to various hand shapes, and costs less infer time.

C. POSE AND SHAPE MODEL

We use a 3D mesh of triangles $M$ and vertexes $N$ to represent the human hand. It parameterizes both pose $\theta$ and shape $\beta$ to deform a $N$-vertex triangular mesh with a hierarchical skeleton $J$. It is illustrated in Figure 5.

![Figure 5. The mesh model of the hand and corresponding hierarchical skeleton. The skeleton is set in a 'rest' pose.](image)

To be able to articulate the complex hand motion, we use a standard kinematic skeleton to denote a hierarchy of joints and the transformations between them. We parameterize the hand pose as $\theta \in \mathbb{R}^{26}$ (six parameters for global translation and orientation, four for each finger), and hand shape is encoded via scalar length parameters $\beta \in \mathbb{R}^{20}$. Given a vector $[\theta, \beta]$ consists of pose $\theta$ and shape $\beta$, the surface (including vertexes and corresponding normal) of the mesh model can be computed by a standard technique which in computer graphics is called linear blend skinning (LBS). We refer the reader to [18] for details.

Figure 6 shows several tracking templates used in recent model-based real-time hand tracking methods. The model in [31] and [35] cannot represent the details of the hand. [24] in terms of some of the problems represented thumb finger. [32] used a spheres-mesh model to efficient estimate hand pose. However, the number of parameters is too large. To tradeoff the accuracy and computational time, we use the LBS model in our experiment.

D. ENERGY FUNCTION

The objective energy function plays an important role in modern hand tracking systems, and a key choice is the energy terms. In this section, we describe each term in our optimization. We first introduce the 2/3D data fitting terms and the computation of correspondences between input depth and model. Then, we discuss the prior terms and their benefits in terms of tracking accuracy and robustness.

The human hand consists of a set of links (bones) and is connected by joints. The joints have two types, rotational or translation. A rotational joint is denoted by a rotation axis and rotate angles, and a translation joint is parameterized by a direction vector and lengths of the link (bone). In this paper, we denote the set of rotation angle (hand pose) as $[\theta_1, \theta_2, \ldots, \theta_n]$, and lengths of link (hand shape) as $[\beta_1, \beta_2, \ldots, \beta_m]$.

To compute $f_1$. We first introduce the skeleton Jacobian, which is first proposed in [3]. The skeleton Jacobian $J_{skel}(t)$ is a $[3 \times n]$ matrix, $n$ is the number of $\theta$, which represents the affected DOFs in the kinematic chain that each 3D point $t$ determined.

As shown in Figure 7, $t_i$ is a depth point, and $s_i$ is the corresponding point on the surface of the hand mesh. We compute $J_{skel}(t)_{i,j}$ by manual differentiation. For the $j$-th joint, let $\theta_j$ be its angle of rotation, $p_j$ is its position, $w_j$ is the LBS weight of $p_j$, and let $v_j$ be the vector pointing along its current axis of rotation (see [38]). The corresponding entry $\frac{\partial s_i}{\partial \theta_j}$ in the skeleton Jacobian matrix for joint $j$ affecting the i-th surface point $s_i$ is

$$
\frac{\partial s_i}{\partial \theta_j} = \begin{cases} w_j v_j \times (s_i - p_j) & j \text{ is rotational joint} \\
\frac{w_j v_j}{||w_j v_j||} & j \text{ is translation joint} 
\end{cases}
$$

If the i-th surface point is not affected by the j-th joint, then $\frac{\partial s_i}{\partial \theta_j} = 0$.

To jointly optimize the hand shape $\beta$, we propose a bone Jacobian matrix. The i-th column of $J_{bone}(t)$ contains the linearization of the i-th bone about $t$. Each entry of is $J_{bone}(t)_{i,j}$

$$
\frac{\partial s_i}{\partial \beta_j} = w_j v_j
$$

1) DATA ENERGIES

a: 3D POINTS ALIGNMENT

Our $E_{3D}$ term computes the corresponding point $y \in H(\theta, \beta)$ on the surface model for each sensor point $t$. We only compute the correspondence between depth points and the model point on the front-facing part of $H(\theta, \beta)$, which is different from the traditional ICP. In our experiments, for computational efficiency, we set the view ray of the camera model to $n = [0, 0, 1]^T$. We linearize the point cloud alignment energy (pose and shape fitting) as

$$
E_{3D} = \frac{\sum_{y \in H} ||n^T (J_{skel}(y) \delta \theta + d)||^2}{\sum_{y \in H} ||n^T (J_{bone}(y) \delta \beta + d)||^2} \text{ pose fitting}
$$

$$
E_{3D} = \frac{\sum_{y \in H} ||n^T (J_{skel}(y) \delta \theta + d)||^2}{\sum_{y \in H} ||n^T (J_{bone}(y) \delta \beta + d)||^2} \text{ shape fitting}
$$

34208
where $||*||$ denotes the L2 norm, $n$ is the surface normal of $y$, and $d$ is the distance between $y$ and $t$.

**b: 2D SILHOUETTE ALIGNMENT**

The human hand is highly articulated, and fast motion may cause self-occlusions during tracking. Only the 3D alignment energy will not constrain the occluded parts. Based on this consideration, the term $E_{2D}$ align the 2D silhouette of our hand model (we project the front-face point as the 2D silhouette) and the 2D silhouette of the input hand part. $E_{2D}$ energy term is given by

$$E_{2D} = \sum_{x \in H} ||n^T (J_{persp}(x) \ast J_{skel}(x) \delta \theta + (p - q))||^2$$  \hspace{1cm} (7)

where $x$ is the 3D location of a rendered silhouette point $p$, $n$ is the 2D normal at the sensor silhouette location $q$. Here, we compute the 2D correspondences with a 2D distance transform, and we refer the user to appendix B in [38] for more detail of $J_{persp}(y)$.

**2) PRIOR ENERGIES**

Only considering the data fitting will easily lead to unrealistic hand poses. In reality, the motion of the hand is more constrained. We regularize our optimization with finger collision, joint rotation, and temporal priors to ensure that the result is plausible. Each of these terms plays an important role in the stability of our objective function.

---

**a: JOINT ROTATION LIMIT**

To discourage the incorrect tracked posture, we adopt the joint rotation constraint and encode this prior to the energy term

$$E_{\text{rot}} = \sum (\omega_1 \theta_i - \theta_i^{\text{min}})^2 + (\omega_2 \theta_i - \theta_i^{\text{max}})^2$$  \hspace{1cm} (8)

where $\theta^{\text{max}}$ is maximal vector of joint angles and $\theta^{\text{min}}$ is the minimal. $\omega_1$ is set to one if $\theta_i < \theta_i^{\text{min}}$ and to zero otherwise. $\omega_2$ is equal to one if $\theta_i > \theta_i^{\text{max}}$ and to zero otherwise. For the $\theta^{\text{min}}$ and $\theta^{\text{max}}$, we also use the values experimentally determined by [5]. Figure 8 shows the example of joint rotation constraint.

---

**b: COLLISION LIMIT**

To prevent our model from taking on anatomically incorrect result, e.g. the collision between fingers and palm, we approximate fingers and palm in our hand using a set of $S$ spheres. Using spheres instead of triangles may reduce the computations of collision detection. Figure 9 shows a sample
of collision limit. The linearization of the collision energy becomes
\[ E_{col} = \sum_{i,j} \chi(i,j)(n_i^T(J_{skel}(x_i) - J_{skel}(x_j))\delta\theta + d)^2 \]  
(9)
where \( x_i \) and \( x_j \) are the end-points of the shortest distance between the collision sphere \( S_i \) and \( S_j \), \( n_i \) is the surface normal at \( x_i \), and \( d \) is the distance between \( x_i \) and \( x_j \). \( \chi(i,j) \) is an indicator function that evaluates to one if the sphere \( S_i \) and \( S_j \) are colliding, and to zero otherwise.

c: TEMPORAL LIMIT
We use the temporal limit provided by [38] to smooth the tracking result between the current and previous frames. The purpose of this term ensures that the pose of the current frame should be near the previous pose. We encode the temporal prior as
\[ E_{temp} = \sum_{k_i \in K} ((J_{skel}(k_i)\delta\theta + (k_i - k_i^{pre}))^2 \]  
(10)
where \( k_i^{pre} \) is the position of the joints’ locations from the previously optimized frame. \( K \) is the set of current joints’ locations.

E. OPTIMIZATION
We treat the optimization of energy over \([\theta, \beta] \) as a nonlinear least squares problem, and solve it with the Levenberg-Marquardt approach. We adopt Taylor expansion to iteratively approximate the energy terms in Equation-1 and solve the linear system to obtain the update \( \delta\theta \) and \( \delta\beta \) at each iteration.

To speed the convergence, we propose a gesture-guided optimization strategy. We first perform pose estimation to align hand to points(pose fitting stage). Then, we optimize the hand shape alone(shape fitting stage). Finally, we jointly optimize the shape and pose of the hand(fully fitting stage). During shape fitting, we only need to optimize the visible joint associated \( \beta \). For example, a finger that extends outside means that the corresponding \( \beta \) is more confident in optimization. Figure.10 shows our multi-stage optimization.

IV. EXPERIMENTS
A. PRELIMINARIES
We extract the region ROI of the single hand using a similar method as that presented in [27]. We also employ a data augmentation to increase the generalization of the network. Specifically, we randomly rotate the image along the \( z \) axis with a range of \(-15^\circ \) and \( 15^\circ \) and scaling displacement \([0.8, 1.2]\). We train and evaluate our networks on a PC with Intel Core i7 6700K, 32GB of RAM, and an Nvidia 1080-Ti GPU. Net models are implemented with the Caffe framework [17]. When training the networks, we set Adam optimizer with learning rate 0.005, batch-size 32, and weight decay 0.0005. Our approach runtime is 2 ms pre-processing, GCRL network runs in 5ms per-frame, optimization stage costs 30 ms(20 iterations for pose fitting, 5 iterations for shape fitting and 5 iterations for joint optimization). During each iteration, pose update costs nearly 1.0 ms, and shape update costs 0.8 ms. This translates to 30~35 frames per second (only use initial network when larger tracking error in previous frame).

B. EVALUATION METRIC
We compare our results with state-of-the-art approaches on the MSRA-2014 [31], MSRA-2015 [36] and NYU [43].

On MSRA-2014, following the most commonly used metrics in the literature, we choose the average Euclidean distance between the 3D joint location and the ground truth to compare our approach with other state-of-the-art approaches [29], [31].

We choose MSRA-2015 to evaluate the performance of our gesture classifier and root location network. For the classification task, we select the mean classification accuracy to evaluate our gesture classifier. For the root location task, we evaluate the hand location performance using a 3D distance error between the root and ground truth.

We also use NYU [43] to compare our approach with several state-of-the-art methods. We utilize two metrics in this work to evaluate the performance. One is mean Euclidean distance error for each joint across all the test frames. The second is the worst-case accuracy, which represents the fraction of

FIGURE 10. We first initialize the pose with the result provided by the GCRL network, then optimize for a coarse pose(pose fitting stage); after pose fitting, we optimize the hand shape, and finally, perform a refinement in the full-dimensional space(both pose and shape). The red area represents the model in front of the depth data, the blue means model behind the depth data, and the white area represents model near the depth data. (less than 5 mm).
test frames that all have estimated joint Euclidean error below an error tolerance.

C. COMPARISON TO STATE-OF-THE-ART

1) RESULTS ON MSRA-2014

MSRA-2014 consists of six subjects, each subject has a different hand shape and is annotated with 3D position for 21 joints. We compare our approach to two state-of-the-art model-based approaches, including Forth [29], Qian [31]. Forth [29] use PSO to perform fitting between a fixed-size hand model and points. Qian [31] proposed a hand tracking system using ICP-PSO based model fitting. For a fairly comparison, we use the version without the GCRL network as baseline. Additionally, we conduct a version with a fixed size hand model as a baseline w/o to validate the effectiveness of our joint optimization.

As Figure.11 illustrates, our approach outperforms the Forth [29] that uses a fixed-size hand model. We also improve 1 mm than our shape w/o fitting. The results show that our joint optimization strategy can improve the accuracy of hand tracking. The mean error distance for all joints of our strategy is 8.6 mm, which is 0.5 mm smaller than the results of [31] and 9.3 mm smaller than the results of [29]. Some qualitative results of our approach on the MSRA-2014 dataset are illustrated in Figure.12. We see that our approach still obtains reasonable results in complex hand poses.

In summary, we can draw the following conclusions: (i) local gradient descent is more precise and faster than the global search. (ii) jointly optimizing pose and shape will improve the robustness and accuracy of tracking.

Moreover, we list the runtime of several model-based approaches(results from the papers) in Table.1. Compared with the previous approaches like [29] and [31], our approach improves both speed(see Table. 1)and accuracy (see Figure. 11) while being able to restore the hand shape. We consume fewer computing resources (only CPU) than [38] and [42]. Compared with [10] and [38], our joint tracking strategy performs the shape estimation online.

| name          | fps | estimate shape | computation resource |
|---------------|-----|----------------|----------------------|
| Forth [29]    | 14  | no             | GTX 580              |
| Qian [31]     | 25  | no             | i7 3.4GHz            |
| ICP IK [10]   | 60  | no             | i7 4.0Ghz, GTX980     |
| Andrea [38]   | 60  | no             | i7 3.4GHz            |
| Tkach [42]    | 120 | yes            | i7 4.0Ghz, GTX1080    |
| Our           | 35  | yes            | i7 4.0Ghz            |

2) RESULTS ON MSRA-2015

MSRA-2015 has larger viewpoint variations, and yaw nearly spans the full $[-90, 90]$ range and pitch within $[-10, 90]$ degrees, which can make pose estimation challenging. This dataset consists of 76,500 images(only depth) with nine subjects, and each subject has 17 different gestures. For the reason that MSRA-2015 has labels of hand gestures, we first use it to validate the effectiveness of our proposed networks.

Baselines: we create several baselines to validate the effectiveness of our network.

1) RF-C: classify the input depth using the Random Forest, this baseline estimate the gesture of input.

2) RF-R: directly regress the hand global root position using the random forest.

3) CNN-R: directly regress the hand global root position using a network similar to the baseline in [28].

For RF-C and RF-R, we choose the pixel difference features, and the maximum depth of trees is set to 20.

We compare the accuracy of our gesture classifier with several baselines on the MSRA-2015. For all approaches, we use the subjects 1-8 for training and the 9th subject for testing. For the gesture classification task, our classifier achieves a mean accuracy of 93.8% of its highest accuracies on 17 different gestures. The RF-C only achieves 85.2%. For the root location task, our sub-network of root location has much higher accuracy than RF-R and CNN-R. As shown in Table. 2, the mean error distance for the root position of our approach is 8.5 mm, which is 2.3 mm smaller than the results of RF-R and 0.9 mm smaller than the CNN-R. The performance gain is more obvious, showing that our network can capture more complex hand structure.

| Approaches | Our Mean error(mm) | RF-R Mean error(mm) | CNN-R Mean error(mm) |
|------------|---------------------|----------------------|----------------------|
| Our        | 8.5                 | 10.7                 | 9.4                  |

3) RESULTS ON NYU

On NYU, we compare our approach with several the state-of-the-art work [6], [12], [27], [44], [50] and [28]. Several baselines are also used to validate the effectiveness of our GCRL network and joint optimization strategy. We refer to a version that directly joints optimize the hand pose and
shape as w/o GCRL, and use the GCRL network but only optimize the hand pose using a fixed-size hand mesh as w/o shape.

Figure.13 shows the mean error results of our approach compared with these approaches. The results show that our approach is slightly superior to [12], [27], [28], [50], and is comparable to [6], [44]. Our approach achieves a mean joints error of 11.87 mm, which is approximately 4.1 mm smaller than [27], 5.1 mm smaller than [50], and near 9 mm smaller than [28]. The accuracy of our approach is similar to [6] and [44], while both of them rely heavily on GPU to achieve real-time performance.

Compared with the w/o GCRL(mean error 15.37mm), the results show the effectiveness of our GCRL network. In some cases, the self-occlusion and noise cause the w/o GCRL(mean error 13.69mm) tracking loss. The comparison result with the w/o shape also indicates the success of our joint optimization strategy.
V. DISCUSSION AND CONCLUSION

The current implementation of our approach works well for the majority of poses, and the reconstruction is hard when the hand is in serious occlusion cases. In addition, we find that the viewpoint variations of the camera will seriously influence the tracking result. When the hand is in the ‘fist’ state, although our GCRL network provides an initialization, if the palm is not facing the camera, the occluded part of the hand will be lost. Additionally, in this angle of camera view, although the GCRL network set a corresponding initial pose of estimated hand gestures in the predicted hand center, the serious edge noise and heavy self-occlusion often cause matching failure. In principle, after using the GCRL network and jointly optimizing the shape and pose of the hand, our approach fails badly only in extreme views and severe self-occlusion.

Another contribution of this paper is the demonstration that both the initialization network and joint optimization strategy not only contribute to the state-of-the-art accuracy shown above but also allow us to maintain this approach on a low computation device. Three variables that determine the amount of computing fitting procedure: (i) The number of data points used in the data term; (ii) The number of iterations we perform for each starting point, (iii) The initial result provided by the GCRL network.

In recent years, some works [14], [25], [41] have focused on the tracking two interacting hands. They all adopted the strategy of ‘left/right-hand segment + pose/shape optimization’. This strategy makes this problem feasible. The end-to-end network has not yet emerged due to the lack of a large quantity of effective training data. Therefore, the model-based optimization method will play an important role in tracking two interacting hands in the future.

In this paper, we propose a novel approach for hand tracking that consists of deep-based pose initialization and gesture-guided pose/shape optimization. The GCRL network captures a meaningful hand structure to estimate gesture and hand root location, thus providing a robust initial pose. Starting from the estimated pose, we jointly estimate pose and shape. By integrating the deep-based initialization and optimizing the parameters of shape selectively, our approach results in faster convergence and increased robustness. Extensive experiments on three datasets demonstrate the effectiveness of our proposed approach.

REFERENCES

[1] R. A. Guler, N. Neverova, and I. Kokkinos, “Densepose: Dense human pose estimation in the wild,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., Jan. 2018, pp. 7297–7306.

[2] A. Boukhayma, R. de Bem, and P. H. S. Torr, “3D hand shape and pose from images in the wild,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2019, pp. 10843–10852.

[3] S. R. Buss, “Introduction to inverse kinematics with jacobian transpose, pseudoinverse and damped least squares methods,” IEEE J. Robot. Automat., vol. 17, no. 1, p. 1, Apr. 2004.

[4] Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei, and Y. Sheikh, “OpenPose: Realtime multi-person 2D pose estimation using part affinity fields,” 2018, arXiv:1812.08008. [Online]. Available: http://arxiv.org/abs/1812.08008

[5] T. F. Chan and R. V. Dube, “A weighted least-norm solution based scheme for avoiding joint limits for redundant manipulators,” IEEE Trans. Robot. Automat., vol. 11, no. 2, pp. 286–292, Apr. 1995.

[6] X. Chen, G. Wang, C. Zhang, T.-K. Kim, and X. Ji, “SHPNet: Deep semantic hand pose regression from point clouds,” IEEE Access, vol. 6, pp. 43425–43439, 2018, doi: 10.1109/ACCESS.2018.2863540.

[7] M. de La Gorce, D. J. Fleet, and N. Paragios, “Model-based 3D hand pose estimation from monocular video,” IEEE Trans. Pattern Anal. Mach. Intell., vol. 33, no. 9, pp. 1793–1805, Sep. 2011.

[8] P. Dolioti, V. Athitsos, D. Kosmopoulos, and S. Perantonis, “Hand shape and 3D pose estimation using depth data from a single cluttered frame,” in Advances in Visual Computing. ISVC (Lecture Notes in Computer Science), vol. 7431, G. Bebis, Eds., et al. Berlin, Germany: Springer, 2012.

[9] A. Erdol, G. Bebis, M. Nicosia, R. D. Boyle, and X. Twombly, “Vision-based hand pose estimation: A review,” Comput. Vis. Image Understand., vol. 108, nos. 1–2, pp. 52–73, 2007.

[10] S. Fleishman, M. Klinger, A. Lerner, and G. Kutliroff, “ICPIK: Inverse kinematics based articulated-ICP,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW), Jun. 2015, pp. 28–35.

[11] G. Garcia-Hernando, S. Yuan, S. Baek, and T.-K. Kim, “First-person hand action benchmark with RGB-D videos and 3D hand pose annotations,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., Jun. 2018, pp. 409–419.

[12] L. Ge, H. Liang, J. Yuan, and D. Thalmann, “3D convolutional neural networks for efficient and robust hand pose estimation from single depth images,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jul. 2017, pp. 5679–5688.

[13] L. Ge, Z. Ren, Y. Li, Y. Wang, J. Cai, and J. Yuan, “3D hand shape and pose estimation from a single RGB image,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2019, pp. 10833–10842.

[14] S. Han, B. Liu, R. Wang, Y. Ye, C. D. Twigg, and K. Kin, “Online optical-marker-based hand tracking with deep labels,” ACM Trans. Graph., vol. 37, no. 4, pp. 1–10, Jul. 2018.

[15] Z. Hu, Y. Hu, J. Liu, B. Wu, D. Han, and T. Kurfess, “A CRNN module for hand pose estimation,” Neurocomputing, vol. 333, pp. 157–168, Mar. 2019.

[16] U. Iqbal, P. Molchanov, T. Breuel, J. Gall, and J. Kautz, “Hand pose estimation via latent 2.5D heatmap regression,” in Proc. Eur. Conf. Comput. Vis., 2018, pp. 118–134.

[17] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, “Caffe: Convolutional architecture for fast feature embedding,” 2014, arXiv:1408.5093. [Online]. Available: http://arxiv.org/abs/1408.5093

[18] S. Khanski, J. Taylor, J. Shotton, C. Keskin, S. Izadi, and A. Fitzgibbon, “Learning an efficient model of hand shape variation from depth images,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), 2015, pp. 2540–2548.

[19] P. Krejov, “Real time hand pose estimation for human computer interaction,” Ph.D. dissertation, Dept. Eng. Phys. Sci., Univ. Surrey, Guildford, U.K., 2016.

[20] P. Krejov, A. Gilbert, and R. Bowden, “Combining discriminative and model based approaches for hand pose estimation,” in Proc. 11th IEEE Int. Conf. Workshops Autom. Face Gesture Recognit. (FG), May 2015, pp. 1–7.

[21] Y. Lin, X. Chai, and X. Chen, “Kinematic constrained cascaded autoencoder for real-time hand pose estimation,” in Proc. 13th IEEE Int. Conf. Automat. Face Gesture Recognit. (FG), May 2018, pp. 45–51.

[22] M. Madadi, S. Escalera, A. Carruesco, C. Andujar, X. Baro, and I. Gonzalez, “Top-down model fitting for hand pose reconstruction in sequences of depth images,” Image Vis. Comput., vol. 79, pp. 63–75, Nov. 2018.

[23] A. Makris and A. Argyros, “Model-based 3D hand tracking with on-line shape adaptation,” in Proc. Proceedings Brit. Mach. Vis. Conf., 2015, p. 77.

[24] S. Melax, L. Keselman, S. Orsten, “Dyn based 3D skeletal hand tracking,” in Proc. ACM SIGGRAPH Symp. Interactive 3D Graphics, Games, 2013, p. 184.

[25] F. Mueller, M. Davis, F. Bernard, O. Sotnychenko, M. Verschour, M. A. Studeny, D. Casas, and C. Theobalt, “Real-time pose and shape reconstruction of two interacting hands with a single depth camera,” ACM Trans. Graph., vol. 38, no. 4, pp. 1–13, Jul. 2019.

[26] A. Newell, K. Yang, and J. Deng, “Stacked hourglass networks for hand pose estimation,” 2016, arXiv:1603.06937. [Online]. Available: http://arxiv.org/abs/arXiv:1603.06937

[27] M. Oberweger, P. Wohlhart, and V. Lepetit, “Training a feedback loop for hand pose estimation,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Dec. 2015, pp. 3316–3324.
[28] M. Oberweger, P. Wohlhart, and V. Lepetit, “Hands deep in deep learning for hand pose estimation,” 2016, arXiv:1502.06807. [Online]. Available: https://arxiv.org/abs/1502.06807

[29] I. Oikonomidis, N. Kyriazis, and A. Argyros, “Efficient model-based 3D tracking of hand articulations using kinect,” in Proc. Procsdings Brit. Mach. Vis. Conf., 2011, p. 3.

[30] G. Pavlakos, V. Choutas, N. Ghorbani, T. Bolkart, A. A. Osman, D. Tzionas, and M. J. Black, “Expressive body capture: 3D hands, face, body from a single image,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Jun. 2019, pp. 10975–10985.

[31] C. Qian, X. Sun, Y. Wei, X. Tang, and J. Sun, “Realtime and robust hand tracking from depth,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Jun. 2014, pp. 1106–1113.

[32] E. Remelli, A. Tkach, A. Tagliasacchi, and M. Pauly, “Low-dimensionality calibration through local anisotropic scaling for robust hand model personalization,” in Proc. IEEE Int. Conf. Comput. Vis. (ICCV), Oct. 2017, pp. 2535–2543.

[33] J. Romero, D. Tzionas, and M. J. Black, “Embodied hands: Modeling and capturing hands and bodies together,” ACM Trans. Graph., vol. 36, no. 6, pp. 1–17, Nov. 2017.

[34] T. Sharp, C. Keskin, D. Robertson, J. Taylor, J. Shotton, D. Kim, C. Rhemann, J. V. A. Leichter, and Y. Wei, “Accurate, robust, flexible real-time hand tracking,” in Proc. ACM Conf. Hum. Factors Comput. Syst., 2015, pp. 3633–3642.

[35] S. Sridhar, F. Mueller, A. Oulasvirta, and C. Theobalt, “Fast and robust hand tracking using detection-guided optimization,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2015, pp. 3213–3221.

[36] X. Sun, Y. Wei, S. Liang, X. Tang, and J. Sun, “Cascaded hand pose regression,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2015, pp. 824–832.

[37] J. S. Supancic, G. Rogez, Y. Yang, J. Shotton, and D. Ramanan, “Depth-based hand pose estimation: Data, Methods, and challenges,” in Proc. IEEE Int. Conf. Comput. Vis. (ICCV), Dec. 2015, pp. 1868–1876.

[38] A. Tagliasacchi, M. Schröder, A. Tkach, S. Bouaziz, M. Botsch, and M. Pauly, “Robust articulated-ICP for real-time hand tracking,” Comput. Graph. Forum, vol. 34, no. 5, pp. 101–114, Aug. 2015.

[39] D. J. Tan, T. Cashman, J. Taylor, A. Fitzgibbon, D. Tarlow, S. Khamis, S. Izadi, and J. Shotton, “Fits like a glove: Rapid and reliable hand shape personalization,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2016, pp. 5610–5619.

[40] J. Taylor, L. Bordeaux, T. Cashman, B. Corish, C. Keskin, T. Sharp, E. Soto, D. Sweeney, A. Topalian, and A. Topalian, “Efficient and precise interactive hand tracking through joint, continuous optimization of pose and correspondences,” ACM Trans. Graph., vol. 35, no. 4, p. 143, 2016.

[41] J. Taylor, V. Tankovich, D. Tang, C. Keskin, D. Kim, P. Davidson, A. Kowdle, and S. Izadi, “Articulated distance fields for ultra-fast tracking of hands interacting,” ACM Trans. Graph., vol. 36, no. 6, pp. 1–12, Nov. 2017.

[42] A. Tkach, A. Tagliasacchi, E. Remelli, M. Pauly, and A. Fitzgibbon, “Online generative model personalization for hand tracking,” ACM Trans. Graph., vol. 36, no. 6, pp. 1–11, Nov. 2017.

[43] J. Tompson, M. Stein, Y. Lecun, and K. Perlin, “Real-time continuous pose recovery of human hands using convolutional networks,” ACM Trans. Graph., vol. 33, no. 5, pp. 1–10, Sep. 2014.