HandMap: Robust Hand Pose Estimation via Intermediate Dense Guidance Map Supervision

Xiaokun Wu, Daniel Finnegan, Eamonn O'Neill and Yong-Liang Yang

Department of Computer Science,
University of Bath, UK
{xw943,d.j.finnegan,E.ONeill,y.yang2}@bath.ac.uk

Abstract. This work presents a novel hand pose estimation framework via intermediate dense guidance map supervision. By leveraging the advantage of predicting heat maps of hand joints in detection-based methods, we propose to use dense feature maps through intermediate supervision in a regression-based framework that is not limited to the resolution of the heat map. Our dense feature maps are delicately designed to encode the hand geometry and the spatial relation between local joint and global hand. The proposed framework significantly improves the state-of-the-art in both 2D and 3D on the recent benchmark datasets.

Keywords: hand pose estimation, dense guidance map, intermediate supervision

1 Introduction

Robust hand pose estimation is essential for emerging applications in human-computer interaction, such as virtual and mixed reality, computer games, and freehand user interfaces. In this work, we focus robust hand pose estimation from a single depth image, a challenging task due to the wide possibility of poses, missing geometric information caused by self-occlusions, and extreme viewpoints.

The recent development of 3D sensing and machine learning techniques have resulted in large datasets with labeled hand pose frames [1–3], along with sophisticated network structures that can cope with challenging learning tasks [4]. The state-of-the-art learning-based hand pose estimation methods [1, 5–14] take the above advantages and have demonstrated promising performances over traditional methods using random forests and their variants [15–17, 2, 18].

With respect to the learning outcome, recent work can be classified into regression-based and detection-based methods [19]. Regression-based methods directly regress 3D coordinates of hand joints. While the method is straightforward, the mapping from input data to joint locations is highly non-linear and poses challenges for the learning procedure. Detection-based methods learn the probability distribution of individual hand joints. The output from the learned model is a heat map that consists of discrete probability values of joint locations. As a result, the accuracy of hand joint estimation is restricted to the resolution of the heat map.
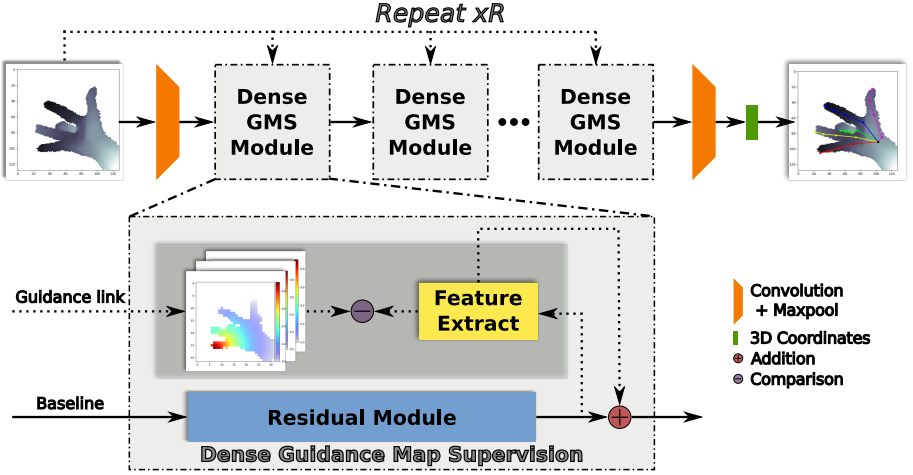
Fig. 1: The pipeline of our algorithm starts from a single depth image. Our baseline method (shown in solid line) stacks R repetitions of a residual module on lower dimensional feature space, then directly regresses 3D coordinates of each joint. In comparison, our proposed method (shown in dashed line) densely samples geometrically meaningful constraints from the input image, which provides coherent guidance to the feature representation of residual module.

In this paper, we present a novel hand pose estimation framework that leverages the advantages of regression-based and detection-based methods. Our framework incorporates feature space constraints on the joint predictions, which act as an 'intermediate' supervision module to a regression-based learning pipeline. This helps to regularize the learning problem, resulting in more robust estimations. Our method is inspired by [20], where intermediate supervision was tested for 2D human pose estimation. We demonstrate our approach accurately estimates hand poses regardless of the dimension of hand data representation (e.g., 2D or 3D). To resolve the ambiguity and occlusion between hand joints, we use dense guidance maps for the supervision instead of sparse heat maps, giving better estimation results.

We summarize our contributions as follows:

1. We apply feature space supervision via dense guidance maps, which are consistent within the entire feature domain, and robust to occlusions.
2. The design of our network structure combines detection based method and regression based method, and benefits from the added accuracy of intermediate predictions.
3. We systematically evaluate different types of guidance maps to prove their effectiveness, achieving improved results by combining with state-of-the-art approaches.

2   Related work

Methods for estimating hand poses from a single view using depth information are generally classified into three categories: discriminative methods, generative methods, and hybrid methods. We review related methods from each category, in particular discriminative methods, the category into which our method falls.

Generative methods fit a pre-defined hand model to each frame of the depth data to temporally track hand poses. At the beginning, they require a model-data calibration step by aligning the hand model to a standard pose to start the tracking, where user input is often required to guarantee a good start. During tracking, the estimated hand pose in the current frame is used to initialize the fitting of the next frame, which means the error can be easily accumulated due to self-occlusion, quick movement of the hand, etc. In the worst case, a re-calibration step is needed to restart the tracking. Hand models with different representations are used to balance the efficiency and accuracy of the temporal tracking: examples include the Linear Blend Skinning model which uses a skeleton to drive the deformation of the hand skin [21, 22], the primitive-based hand model which uses cylinder and cones to represent hand segments [23], the Gaussian mixture model that represents hand as a mixture of Gaussian kernels [24, 25], the mesh model that indicates the envelop of the hand [26], and the sphere mesh that defines the hand using blending surfaces between spheres at key locations with different radii [27]. Various optimization techniques are employed to fit the pre-defined hand model to the depth data according to a carefully designed matching function, such as particle swarm optimization [28], Iterative Closest Point registration [27], and a combination of the two [29].

Discriminative methods directly learn a hand pose estimator from pre-labelled data. The mapping between depth data and hand pose is established using different discriminative models. Early work such as [15–17, 2, 18] apply random forests and their variants which rely on hand-crafted features for learning, restricting their performance compared to methods which use deep neural networks, benefiting from learned features. Recent works utilize CNNs and the maturity of massively labelled data to further improve the performance [1, 5–14]. The state-of-the-art methods have been classified and discussed in a recent survey based on the 'HANDS 2017' challenge [19] with respect to different aspects, such as learning outcome (regression vs. detection), dimension of CNN (2D vs. 3D), learning model structure (hierarchical vs. non-hierarchical), etc. A detailed review of previous work from all aspects is beyond the scope of this paper. In the rest of this paper we limit our discussion to regression or detection-based approaches to highlight our motivation and contribution. Interested readers are directed to [19] for a comprehensive study.

Regression-based methods take a depth image and directly regresses the 3D coordinates of hand joints. Guo et al. [9, 10] present a region ensemble network for hand joint regression. Based on this network, Chen et al. [12] propose to apply iterative refinement of the estimated pose for better results. Oberweger

et al. [13] estimate hand poses based on enhanced network architecture, data augmentation, and better initial hand localization. Madadi et al. [11] exploit a hierarchical tree-like structured CNN to estimate hand joints from local poses. Unlike the above methods which treat depth data as 2D images, Ge et al. [14] train a 3D CNN based on projective distance fields from three canonical views to regress hand joint locations. In contrast to regression, detection-based methods predict a probability density map for each joint. Tompson et al. [1] predict the probability distribution of joint locations as a heat map using CNNs. Ge et al. [5] extend the method by using depth information from multiple views. Moon et al. [30] use a 3D CNN to estimate per-voxel likelihood for each joint, resulting in the best overall performance in the 'HANDS 2017' challenge compared with previous methods based on 2D CNNs and hand joint regression.

According to the estimation error statistics in [19], detection-based methods seem to be superior to regression-based methods given the top three estimation results are all detection-based. This reflects the difficulty of the highly non-linear mapping between the depth map and 3D hand joint coordinates. Simply relying on neural networks to do a joint regression is not good enough. On the other hand, in our hand pose estimation practice, we realize that detection-based methods also have their own restrictions due to the limited resolution of the predicted heat map for joint distribution, making it hard to identify accurate hand location within a single map element (i.e., pixel in 2D or voxel in 3D). Recent work in the human body pose estimation proposes a combined framework in a multi-task setup [31, 32], but the performance of adopting a similar idea in hand pose estimation is still unknown. In this work, we leverage the advantages of a multi-stage multi-target framework to attack domain-specific challenges, which leads to increased accuracy and robustness.

Zeiler et al. [33] posited that intermediate outputs of neural networks can be used to represent extracted features from the networks' overall input. From this, we posit that adequately well designed features could serve as good constraints to the intermediate layers of a neural network. Newell et al. [20] adopt the idea of *intermediate supervision* and test 2D human pose estimation in their work, but their results are limited to 2D which cannot easily resolve depth ambiguity. We propose to combine both detection and regression based approaches by adding the regression stage after the dense guidance map supervision module to robustly output 3D joint locations. Our 3D pose estimation system takes the benefits of added accuracy from detection-based methods as reported in [19], but we impose intermediate constraints on the feature space instead of the output space.

Hybrid methods perform temporal hand tracking using generative approach while re-initializing tracking via discriminative approach if error accumulates. Various re-initialization strategies have been used, including particle swarm optimization [29], Deep Neural Networks [1], random ferns and forests [34], and retrieval forests [26].

3   Hand pose estimation via intermediate dense guidance map supervision

Due to constrained viewing direction, and intra-occlusion among different hand parts (e.g., finger-finger occlusion, finger-palm occlusion), the depth camera can only see partial hand surface S in one frame. Also due to limited scanning resolution, we can only rely on a discretely sampled 2D depth map D as the raw input to our problem. Hand pose is conventionally represented by a constellation of 3D key points, which correspond to a fixed number of anatomic hand joints. Our goal is to estimate the 3D coordinates of all hand joints, which altogether form a vector $Y \in \mathbb{R}^{J \times 3}$, where J is the number of joints. These joints provide crucial spatial information for hand tracking based applications.

Next we describe the details of our hand pose estimation framework presenting the mathematical foundations based on direct regression, and elaborate on our novel learning framework introducing its key module: intermediate guidance map supervision.

Hand pose estimation via direct regression Suppose a partial hand surface S could be described from D through *inverse depth sampling* function $\iota : D \mapsto \bar{S}$, where $\bar{S}$ is sufficiently smooth and can approximate S up to any required order, we aim to estimate the hand pose prediction function $\Psi : D \mapsto \bar{Y}$.

Since directly estimating the highly non-linear function Ψ in high dimensional space is very hard, we adopt a learning-based approach to first establish the feature extraction mapping $\Phi : D \mapsto \chi$ through Convolutional Neural Networks (CNNs). The hidden feature mapping Φ is subsequently linked to $\bar{Y}$ as a regression function $\Pi : \chi \mapsto \bar{Y}$, which is optimized together with Φ in the training procedure:

$$\mathcal{L}_{regre}(\Psi) = \mathcal{L}_{regre}(\Pi(\Phi)) = \frac{1}{2}\|\bar{Y} - Y\|_2^2. \tag{1}$$

We call this approach *direct regression*, which reflects the raw approximation capability of its underlying network. However, naïve application of simple learning systems can hardly achieve necessary representing power to produce high accuracy results, as our target function Ψ is highly non-linear. Also, resorting to complicated deep network design is overwhelming, and can easily cause overfitting problem. Therefore, we treat this approach as the baseline to demonstrate the effectiveness of our proposed approach as follows.

3.1   Overview of our approach

Figure 1 illustrates the overall structure of our pipeline, which has a repeated residual module integrated into a conventional CNN-based framework as the stem (solid line). Note that we do not limit the dimension of input data: our pipeline can cope with both 2D and 3D hand representation, and here we denote m as the input resolution (i.e., number of pixels in 2D, or voxels in 3D). The input data is passed through multiple convolution and max-pooling layers (in

orange), until it reaches the desired resolution k. This stage can be seen as preliminary feature extraction and down-sampling, which reduces computational cost by working on lower resolution features. Then the preliminary features are fed into our core Guidance Map Supervision (GMS) module (shown in grey), where features are refined R times. At last the refined features are served as the input of the final pose regression stage. We first employ convolution and max-pooling layers (in orange) to adapt the feature dimension to the final output [35], then utilize a fully-connected layer (in green) for the final regression.

We use residual module [36] as the foundation of our GMS module, which has the ability to learn feature differences. We calculate geometrically meaningful constraints from the input, then incorporate them as guidance maps through a similar design as the residual module. As shown in the dark grey region, a side branch is spread out (in dashed line) from the output of the residual module (in blue), which can be seen as higher level feature abstraction (in yellow). The extracted higher-level features are compared to guidance maps (which provides an error function), then these supervised features are added back to the main branch. Guidance map supervision and residual link together leverage the feature extraction effectiveness of the residual module through error feedbacks, which further enhance the entire system's learning strength.

3.2   Guidance map supervision

The guidance map $\Gamma : D \mapsto \zeta$ calculates the spatial response ζ across the entire input domain D, which reflects the probability of a specific hand joint. It is used to enhance the direct regression method such that the resultant hand joint locations can be robustly estimated. This is inspired by the multi-stage supervision approach in [20]. The core problem here is how to design a geometric meaningful guidance map that is effective for hand joint regression. We first present the simplest choice of using heat map based on Gaussian distribution, which we call *sparse guidance map*. We will then discuss our contribution on *dense guidance map* that better represents the geometric and spatial property of hand joints.

Sparse Guidance Map Supervision The most straightforward guidance map in 2D could be easily implemented as J heat-map images [37], each of which contains the sampled pixel-wise probability values for one hand joint (see Figure 2a). In our implementation, we first assign 1 to the pixel projected from the labelled joint, then filter the image using Gaussian kernel $\mathcal{G} : D \mapsto \zeta_{heat}$ to ensure that the resultant heat-map ζ_{heat} is still a probability distribution. We also choose very small variance σ for $\mathcal{G}$ to reduce inter-joint ambiguity.

In 3D we use more restricted one-shot probability maps ζ_{one} [30] where only a single voxel is positive for each joint in the volume, compromising resolution but saving on computational cost.

Note that these kind of guidance maps are inherently narrow-band probability distributions, necessary for producing high accuracy likelihood output. Hence we call them sparse guidance maps. Here we indiscriminately denote both 2D
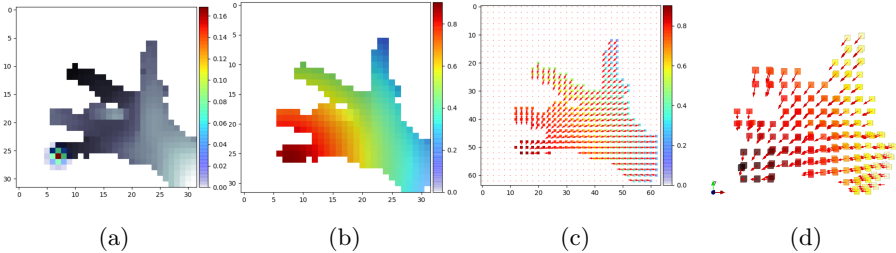
Fig. 2: Different guidance maps (here we only show illustrations for the pinky finger tip). (a) 2D probability map ζ_{heat}. (b) Normalized Euclidean distance Ω_{dist}. (c-d) 2D/3D Euclidean distance Ω_{dist} plus unit offset Ω_{unit}.

and 3D guidance map as K, then the loss of intermediate supervision could be formulated as the cross-entropy between the predicted probability map $\bar{\zeta}$ and the ground-truth probability map ζ:

$$\mathcal{L}_{sparse}(\Gamma) = \sum_{j \in J} \sum_{v \in K} -\zeta_j^v * log(\bar{\zeta}_j^v). \qquad (2)$$

The problem with these approaches is that localized joint detection without a large supporting neighbourhood of pixels/voxels in consensus can easily result in false positive predictions [38], especially in the case of hand pose estimation where different joints are ambiguous to each other. Also, as most of the guidance map entries are zero, this suppresses the activation of the feature maps' energy function (Equation 2), consequently leading to unsatisfactory results in the final regression stage.

Dense guidance map supervision Instead of using sparse guidance maps, we propose to use densely sampled vector or scalar fields (called dense guidance maps) as the intermediate feature maps. The dense guidance map is carefully designed to represent each individual joint while maintaining consistency across the entire feature domain K. Our design is not limited to a specific form of dense field, but in practice we have tested several geometric meaningful choices as listed below.

Offset maps: One simple choice of dense guidance map is to use a vector field Ω_{offset} composed of vectors pointing to the joint location from individual pixels/voxels. The magnitude of each vector is the Euclidean distance between the pixel/voxel and the joint. It is easy to see that in 2D we need to calculate $J \times 2$ guidance maps, while in 3D we need to calculate $J \times 3$ guidance maps. In practice such un-normalized offsets in the feature space may cause numerical instability due to drastic change of scales of learned convolution weights, leading to unsat-

isfactory regression results. This motivates us to design better dense guidance maps as follows.

Normalized distance plus unit offset maps: An alternative approach is to divide Ω_{offset} into two parts in our experiment: 1) a scalar field Ω_{dist}, with inverse Euclidean distance to the correct joint location, normalized by the feature domain extend k; and 2) a vector field Ω_{unit} which is calculated as $\Omega_{offset}/\|\Omega_{offset}\|_2$.

Notice that we calculate Ω_{dist} using inverse distance, resulting in maximum value 1 at the joint location. This can be treated as a natural generalization of sparse heat-map ζ_{heat}, with support extended to the entire feature domain. Here we also do not truncate Ω_{dist} within any localized support, because joints are frequently outside of the truncating radius due to occlusion (see Figure 4b).

Normalized distance: The problem with combined Ω_{dist} and Ω_{unit} is that the computational cost is too high ($J \times 3$ guidance maps for 2D and $J \times 4$ for 3D). Notice that Ω_{unit} is proportional to the gradients (first order derivatives) of Ω_{dist}: $\Omega_{unit} = \frac{\nabla\Omega_{dist}}{\|\nabla\Omega_{dist}\|}$, so we can actually only use distance map Ω_{dist} as the guidance feature.

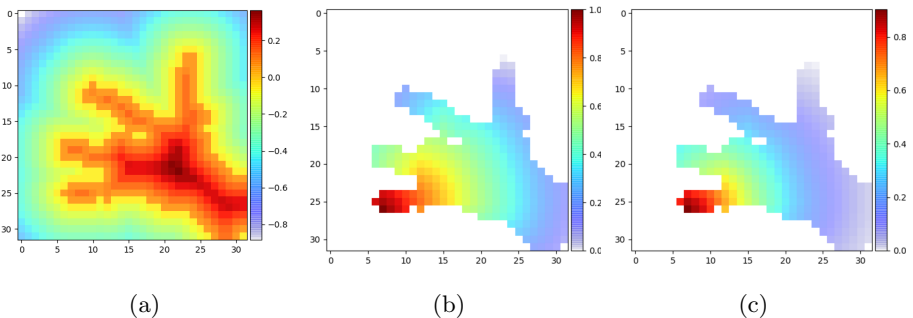(a)   (b)   (c)

Fig. 3: Geometrically more meaningful guidance maps. (a) EDT map used for propagating distance from a single point. (b-c) Our implementation of approximate geodesic distance map Ω_{edt} and Ω_{edt2} for the pinky finger tip.

Approximated geodesic distance: From the last discussion we know that the design of distance map is the key to the performance of intermediate supervision. Also, given that $\iota(D)$ is a smooth surface embedded in $\mathbb{R}^3$, we propose to use a geodesic distance map Ω_{geo} as our ultimate choice of guidance map. Note that geodesics are computationally expensive and only well defined on complete surfaces. However, the assumption of $\mathbb{R}^3$ is problematic, considering our input depth image D is only "2.5D" by definition, as we can only capture partial surface that is visible from one view point.

To solve this problem, we propose a computationally efficient approxima-
tion to Ω_{geo} by first calculating (signed) Euclidean Distance Transform (EDT)
map [39] on the projected depth image D in a preprocessing step (see Figure 3a).
Here we need to be careful to compute distances outside of the hand region as
well, since it is very common that some joint annotations are isolated due to
missing hand data in the surrounding space caused by occlusions (see Fig. 4b).
Then we can propagate distances from each joint location using Fast Marching
Method (FMM) [40], resulting in our approximate surface distance map Ω_{edt}
(see Figure 3b). Notice that the distance between pinky finger tip and ring fin-
ger tip is not short any more, in contrast to using simple Euclidean distance as
shown in Figure 2b.

Weighted geodesics approximation: Our design of a surface distance map Ω_{edt} is
not from geometrically accurate measurement, as all calculations are performed
on the 2D image space due to missing data and computational cost in 3D.
However, since we use normalized distances, Ω_{edt} can be interpreted as weights
providing local support. Therefore, we also present another weighted distance
map $\Omega_{edt2} = \Omega_{edt} \odot \Omega_{dist}$, which is more meaningful in terms of measurement,
and also proportional to geometric distances.

For each network prediction output $\bar{\Omega}$, we use the same loss function for all
dense guidance maps:

$$\mathcal{L}_{dense}(\Omega) = \|\bar{\Omega} - \Omega\|_{sl_1}, \tag{3}$$

where $\|x\|_{sl_1}$ is the smoothed l_1-norm that evaluated as $x - 0.5$ for $|x| > 1$; or
$0.5 * x^2$ otherwise [41].

3.3   Final regression stage

The final regression stage estimates all the hand joint locations by training a
CNN based network while minimizing the following loss function $\mathcal{L}$:

$$\mathcal{L} = \mathcal{L}_{regre} + \lambda_1 * \mathcal{L}_{super} + \lambda_2 * \mathcal{L}_{regu}, \tag{4}$$

where $\mathcal{L}_{super}$ can be the loss of either sparse (Equation 2) or dense (Equa-
tion 3) guidance map supervision, and $\mathcal{L}_{regu}$ is the l_2-regularization term ap-
plied to the weights of convolutional operators. $\lambda_{1,2}$ are balancing weights: we
set $\lambda_1 = \frac{1}{k}$, and $\lambda_2 = 0.01$.

Network structure: The core part of our network is the guidance map supervision,
repeated R times, for maximizing supervision effectiveness as in [20]. We didn't
use the "Hourglass" module which recursively applies a residual module [36] in
different scales: while this leads to a computationally inefficient pipeline, the
accuracy in our baseline tests was low.

As a result, we use the Inception-ResNet-v2 module [42] as our main building
block, which is able to extract features at different scales. We empirically found
$R = 2$ performs well for most cases, but the core idea of our algorithm is general
enough without being confined to network specifications.

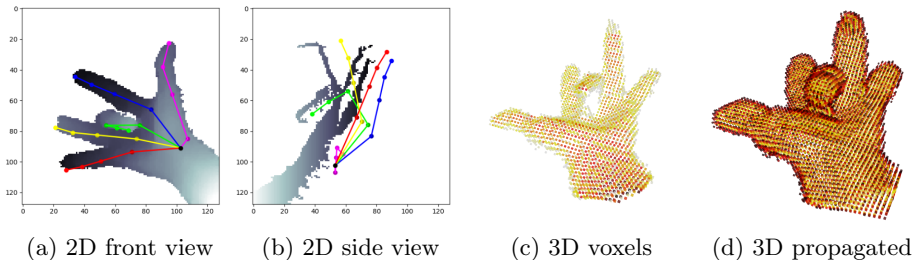| (a) 2D front view | (b) 2D side view | (c) 3D voxels | (d) 3D propagated |

Fig. 4: Input hand data representations with ground-truth markers shown in color. Most algorithms in our discussion use single frontal view (a), while side views (b) are used in multi-view approaches. We use voxelized representation (c) and its variance (d) in 3D cases.

4   Experimental evaluations

'HANDS 2017' challenge dataset [4]: This dataset contains 957K depth images sampled from BigHand2.2M [3] and First-Person Hand Action [43] dataset, together with high accuracy annotations from 6D magnetic sensors. We follow the $8 : 1 : 1$ rule to split the data into training/validation/test set without further augmentation, as our main focus is comparing each method's learning performance. Evaluations are performed on all 21 joints of this dataset.

Data preprocessing: A 3D volume V containing only the hand part segmented out of raw input depth image is a prerequisite, especially for 3D volumetric approaches. We empirically find that an axis-aligned cube of 240mm each side is a good range for cropping out hand part, and we align the cube centre with the centre of all joints. It is also possible to train a separate localizer for detecting hand region [13]. We also attempted a localizer based on "Faster RCNN" [44]. But the extra complexity does not help for fair evaluation of our primary task, so we leave the hand region detection for future work. As input, our method requires the hand region segmented from the raw image frame. We provide this to our system as we are not concerned with automatic segmentation and detection of the hand region.

We re-project points within V onto the image space, and rescale it to size of 128×128 which we take as input D to our pipeline (see Figure 4a). Unless specified, the size of the guidance feature maps is fixed to 32×32 throughout the rest of the paper.

In the case of 3D, we voxelize V into $64 \times 64 \times 64$ grid, where the value C at each voxel is the number of data points within that voxel (see Figure 4c, where darker colour means bigger value). We do not adopt the popular approach [30] that just use binary occupancy as the input, as the intentional extra overhead introduced in our approach works better in capturing local geometry.
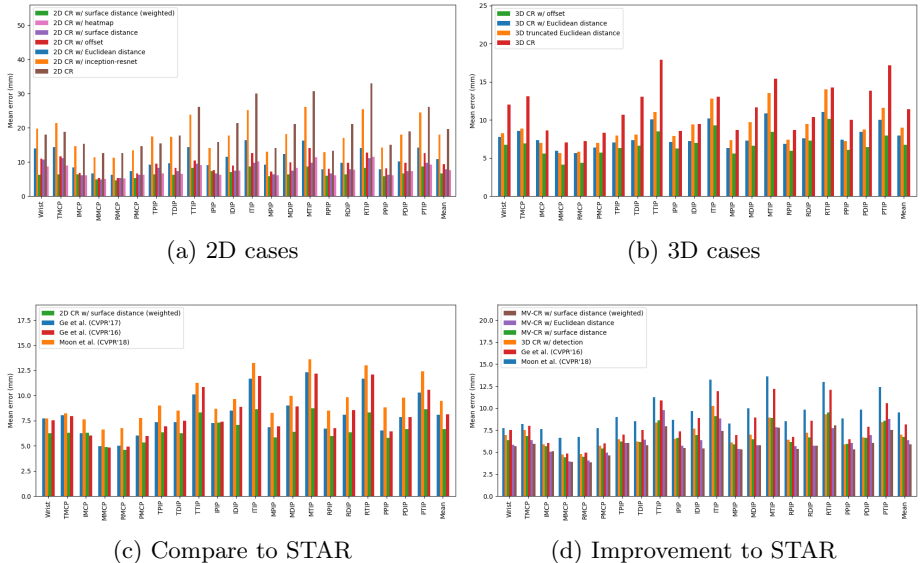
(a) 2D cases

(b) 3D cases

(c) Compare to STAR

(d) Improvement to STAR

Fig. 5: Comparisons of per-joint mean errors.

Other than using raw voxel-wise point number C as input, we also found that propagating those statistics along surface normals within certain distance can act as a better input form P (see Figure 4d). The propagated version produces extended details into the occluded volume, and results in better predictions.

In our experiments, we set the size of 3D guidance feature maps to be 32 for most of our tests, and reduce it to 16 in case of exceeding hardware limitations.

Training: We use Tensorflow [45] to develop learning framework, and our framework is trained using the Adam optimizer [46] with default parameters. The initial learning rate is 0.001, and we use exponential decay with decay rate of 0.94. We use a single GeForce GTX 1080 Ti as the main computing hardware, and choose 50 as the default batch size. We train all models with maximum 10 epochs, stopping the training early when validation loss has grown 10% higher than the last epoch on the validation set. We typically choose the repetition number $R = 2$ for the guidance map supervision module for balancing performance and computational cost, and each epoch takes about 2 hours for approximate geodesics based method. Evaluations are performed on a separate test set.

4.1 Evaluation metrics

The primary metric we use is the mean errors across all test frames for each joint, and we also take their average as a summary of each method's performance. Figure 5 compares methods across all hand joints, using colour to distinguish between methods. The detailed descriptions on tested methods and corresponding

(a) 2D cases

(b) 3D cases
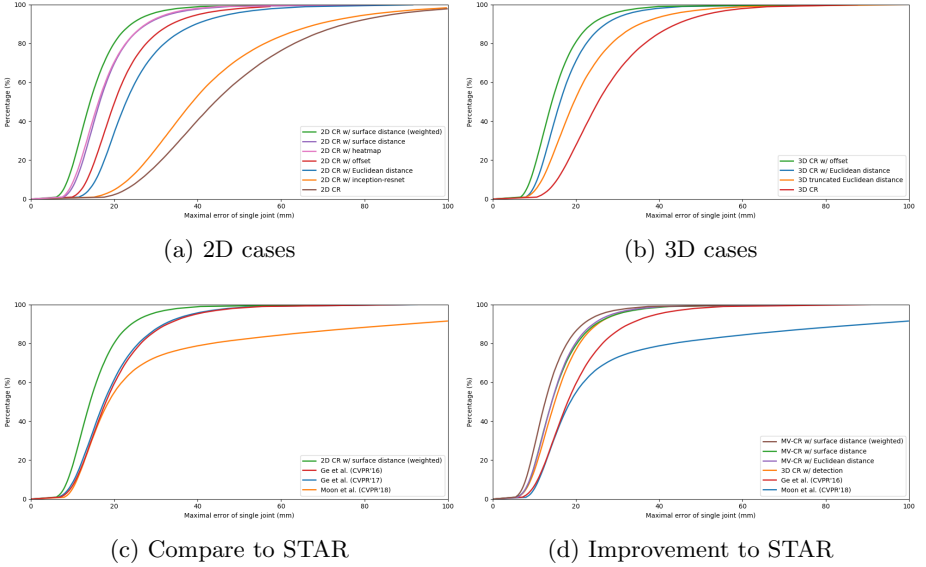
(c) Compare to STAR

(d) Improvement to STAR

Fig. 6: Comparisons of the percentage curves of maximal per-joint errors.

error statistics can be found in Table 1. We also evaluate the maximal per-joint error within each single frame. Figure 6 shows the percentage curve of each method by visualizing the ratio of correct prediction versus maximal allowed error to ground-truth annotations. The detailed error statistics can also be found in Table 1. We also provide more results in the supplementary material.

4.2 Effectiveness of dense guidance map supervision

The direct Coordinate Regression (CR) method is our baseline, and we also show the improved performance after using Inception-ResNet [42].

2D cases: All of the CR methods with guidance map supervision achieve clear improvements over the baseline methods. Intermediate supervision using weighted approximate geodesic distance (Figure 3c) gives best performance in terms of mean error (6.68mm) and maximal per-joint error (8.73mm). The difference between mean and maximal error is only 2.05mm, indicating the robustness of our framework. Intermediate supervision based on sparse guidance map (Figure 2a) and dense guidance map with approximate geodesic distance (Figure 3b) are less effective, and their performances are roughly at the same level. Dense supervision with normalized offset (Figure 2c) performs better than dense supervision with plain Euclidean distance map (Figure 2b) in our tests; however, we dismiss this approach given it requires 3 times computing power (as explained in Section 3.2) for a relatively small improvement in performance.

Table 1: A summary of hand joint estimation methods and the corresponding estimation errors on 'HANDS 2017' challenge dataset. CR: direct Coordinate Regression. MV: Multi-View method [5]. Inception-resnet: Inception-ResNet-v2 module [42]. Euclidean distance: calculated in 3D world coordinates. Truncated Euclidean distance: propagate voxel-wise point count along surface normal. Surface distance: approximate geodesic distance. Offset: normalized Euclidean distance plus unit offset. Offset regression: directly compute 3D joint locations using weighted average of the offset map, without final coordinate regression.

Methods	Mean error (mm)	Max-per-joint error (mm)
MV-CR w/ surface distance (weighted)	5.90	8.04
MV-CR w/ Euclidean distance	6.35	9.77
2D CR w/ surface distance (weighted)	6.68	8.73
MV-CR w/ surface distance	6.74	9.51
3D CR w/ offset	6.76	10.16
3D CR w/ detection	7.00	10.25
2D CR w/ heatmap	7.62	11.53
2D CR w/ surface distance	7.89	11.18
3D CR w/ Euclidean distance	7.92	11.04
Ge et al. (CVPR'17) [14]	8.07	12.30
Ge et al. (CVPR'16) [5]	8.15	12.17
3D truncated Euclidean distance	8.98	14.01
2D CR w/ offset	9.34	14.09
Moon et al. (CVPR'18) [30]	9.49	13.60
2D CR w/ Euclidean distance	10.90	16.44
3D CR	11.39	17.92
2D CR w/ inception-resnet	17.94	26.08
2D CR	19.63	32.98
2D offset regression	49.83	70.88
3D offset regression	51.17	73.58

The benefit of combined detection and regression: Given offset predictions as shown in Figures 2c and 2d, we compute pose estimations: each joint is calculated as weighted average of top 5 Ω_{dist} activations. However, as shown in Table 1, this "Offset Regression" approach under-performs in both 2D and 3D, justifying the benefit of combining detection with regression. We suspect this poor performance is due to: 1) The output space (e.g., $32 \times 32 \times 32 \times 21 \times 4$ for 3D) exceeding the representation capabilities of CNNs implemented with current hardware standards. 2) Over-fitting in extremely high dimensional space can be exaggerated and lead to unreliable predictions.

3D cases: We used direct CR with both voxel-wise point number C and its propagated version P as the baseline, and the proposed framework obtained significantly improved results as 2D tests. We adopted a lowered feature map size $k = 16$ in dense supervision with normalized offset (Figure 2d), due to exhausting computational resources. We skipped surface distance tests due to

high computational costs required while still sacrificing local geometric details
due to limited voxel resolution.

4.3   Comparisons with state-of-the-art (STAR) methods

We have compared our best method (2D CR with Ω_{edt2}) to both 2D [5] and
3D [30, 14] STAR methods, and achieved convincing better performance as shown
in Figures 5c and 6c.

Improvement to STAR methods: Another advantage of our dense guidance map
supervision method is that we can easily insert one or multiple of our mod-
ules into existing learning pipelines, using intermediate feature space constraints
to achieve better results. We applied Multi-View Coordinate Regression (MV-
CR) variants on [5] in 2D, using detection output of [30] as supervision feature
maps, as shown in Figures 5d and 6d. Notice that our MV-CR method with
approximate geodesic distance achieved the minimal mean error across all of our
experiments (5.90mm, see Table 1). We see the improvement as an positive ev-
idence that certain hidden inter-joint constraints could be learned through our
multi-stage dense guidance map supervision approach.

5   Conclusion

We present a general hand pose estimation framework via intermediate super-
vision on dense guidance maps. Our method overcomes issues with high non-
linearity of hand joint regression and the resolution restriction of detection-based
methods. The dense guidance maps are designed to better incorporate the geo-
metric and spatial information of hand joints. We demonstrate the effectiveness
of our framework and the choice of guidance maps by extensive comparisons
with baseline methods in both 2D and 3D. Results show that our framework can
robustly produce hand pose estimates with improved accuracy. Future work will
explore temporal hand tracking using our framework, integrating hand detection
to handle data in the wild, and optimizing computational performance.

Acknowledgements

References

1. Tompson, J., Stein, M., Lecun, Y., Perlin, K.: Real-time continuous pose recovery
   of human hands using convolutional networks. ACM TOG **33**(5) (2014) 169:1–
   169:10

2. Sun, X., Wei, Y., Liang, S., Tang, X., Sun, J.: Cascaded hand pose regression. In: IEEE CVPR. (2015) 824–832

3. Yuan, S., Ye, Q., Stenger, B., Jain, S., Kim, T.K.: Bighand2.2m benchmark: Hand pose dataset and state of the art analysis. IEEE CVPR (2017) 2605–2613

4. Yuan, S., Ye, Q., Garcia-Hernando, G., Kim, T.K.: The 2017 hands in the million challenge on 3d hand pose estimation. arXiv preprint arXiv:1707.02237 (2017)

5. Ge, L., Liang, H., Yuan, J., Thalmann, D.: Robust 3d hand pose estimation in single depth images: from single-view cnn to multi-view cnns. In: IEEE CVPR. (2016) 3593–3601

6. Sinha, A., Choi, C., Ramani, K.: Deephand: Robust hand pose estimation by completing a matrix imputed with deep features. In: CVPR. (2016) 4150–4158

7. Ye, Q., Yuan, S., Kim, T.K.: Spatial attention deep net with partial pso for hierarchical hybrid hand pose estimation. In: ECCV. (2016) 346–361

8. Zhou, X., Wan, Q., Zhang, W., Xue, X., Wei, Y.: Model-based deep hand pose estimation. In: IJCAI. (2016) 2421–2427

9. Guo, H., Wang, G., Chen, X., Zhang, C.: Towards good practices for deep 3d hand pose estimation. CoRR **abs/1707.07248** (2017)

10. Guo, H., Wang, G., Chen, X., Zhang, C., Qiao, F., Yang, H.: Region ensemble network: Improving convolutional network for hand pose estimation. In: 2017 IEEE International Conference on Image Processing (ICIP). (Sept 2017) 4512–4516

11. Madadi, M., Escalera, S., Baró, X., Gonzàlez, J.: End-to-end global to local CNN learning for hand pose recovery in depth data. CoRR **abs/1705.09606** (2017)

12. Chen, X., Wang, G., Guo, H., Zhang, C.: Pose guided structured region ensemble network for cascaded hand pose estimation. CoRR **abs/1708.03416** (2017)

13. Oberweger, M., Lepetit, V.: Deepprior++: Improving fast and accurate 3d hand pose estimation. In: ICCV workshop. Volume 840. (2017) 2

14. Ge, L., Liang, H., Yuan, J., Thalmann, D.: 3d convolutional neural networks for efficient and robust hand pose estimation from single depth images. In: IEEE CVPR. (2017) 5679–5688

15. Keskin, C., Kıraç, F., Kara, Y.E., Akarun, L.: Hand pose estimation and hand shape classification using multi-layered randomized decision forests. In: ECCV. (2012) 852–863

16. Xu, C., Cheng, L.: Efficient hand pose estimation from a single depth image. In: IEEE ICCV. (2013) 3456–3462

17. Tang, D., Chang, H.J., Tejani, A., Kim, T.K.: Latent regression forest: Structured estimation of 3d articulated hand posture. In: IEEE CVPR. (2014) 3786–3793

18. Tang, D., Taylor, J., Kohli, P., Keskin, C., Kim, T.K., Shotton, J.: Opening the black box: Hierarchical sampling optimization for estimating human hand pose. In: IEEE ICCV. (2015) 3325–3333

19. Yuan, S., Garcia-Hernando, G., Stenger, B., Moon, G., Yong Chang, J., Mu Lee, K., Molchanov, P., Kautz, J., Honari, S., Ge, L., Yuan, J., Chen, X., Wang, G., Yang, F., Akiyama, K., Wu, Y., Wan, Q., Madadi, M., Escalera, S., Li, S., Lee, D., Oikonomidis, I., Argyros, A., Kim, T.K.: Depth-based 3d hand pose estimation: From current achievements to future goals. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (June 2018)

20. Newell, A., Yang, K., Deng, J.: Stacked hourglass networks for human pose estimation. In: ECCV. (2016) 483–499

21. Ballan, L., Taneja, A., Gall, J., Van Gool, L., Pollefeys, M.: Motion capture of hands in action using discriminative salient points. In: ECCV. (2012) 640–653

22. Tzionas, D., Ballan, L., Srikantha, A., Aponte, P., Pollefeys, M., Gall, J.: Capturing hands in action using discriminative salient points and physics simulation. Int. Journal of Computer Vision **118**(2) (2016) 172–193

23. Tagliasacchi, A., Schröder, M., Tkach, A., Bouaziz, S., Botsch, M., Pauly, M.: Robust articulated-icp for real-time hand tracking. Computer Graphics Forum (Proc. SGP) **34**(5) (2015) 101–114

24. Sridhar, S., Mueller, F., Oulasvirta, A., Theobalt, C.: Fast and robust hand tracking using detection-guided optimization. In: IEEE CVPR. (2015) 3213–3221

25. Sridhar, S., Mueller, F., Zollhöfer, M., Casas, D., Oulasvirta, A., Theobalt, C.: Real-time joint tracking of a hand manipulating an object from rgb-d input. In: ECCV. (2016) 294–310

26. Taylor, J., Bordeaux, L., Cashman, T., Corish, B., Keskin, C., Sharp, T., Soto, E., Sweeney, D., Valentin, J., Luff, B., Topalian, A., Wood, E., Khamis, S., Kohli, P., Izadi, S., Banks, R., Fitzgibbon, A., Shotton, J.: Efficient and precise interactive hand tracking through joint, continuous optimization of pose and correspondences. ACM TOG (Siggraph) **35**(4) (2016) 143:1–143:12

27. Tkach, A., Pauly, M., Tagliasacchi, A.: Sphere-meshes for real-time hand modeling and tracking. ACM TOG (Siggraph) **35**(6) (2016) 222:1–222:11

28. Iason Oikonomidis, N.K., Argyros, A.: Efficient model-based 3d tracking of hand articulations using kinect. In: BMVC. (2011) 101.1–101.11

29. Qian, C., Sun, X., Wei, Y., Tang, X., Sun, J.: Realtime and robust hand tracking from depth. In: IEEE CVPR. (2014) 1106–1113

30. Moon, G., Yong Chang, J., Mu Lee, K.: V2v-posenet: Voxel-to-voxel prediction network for accurate 3d hand and human pose estimation from a single depth map. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (June 2018)

31. Lin, M., Lin, L., Liang, X., Wang, K., Chen, H.: Recurrent 3d pose sequence machines. In: CVPR. (2017)

32. Popa, A.I., Zanfir, M., Sminchisescu, C.: Deep multitask architecture for integrated 2d and 3d human sensing. In: IEEE International Conference on Computer Vision and Pattern Recognition. (2017)

33. Zeiler, M.D., Fergus, R.: Visualizing and understanding convolutional networks. In: European conference on computer vision, Springer (2014) 818–833

34. Sharp, T., Keskin, C., Robertson, D., Taylor, J., Shotton, J., Kim, D., Rhemann, C., Leichter, I., Vinnikov, A., Wei, Y., Freedman, D., Kohli, P., Krupka, E., Fitzgibbon, A., Izadi, S.: Accurate, robust, and flexible real-time hand tracking. In: SIGCHI Conf. on Human factors in Computing Systems. (2015) 3633–3642

35. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2016) 2818–2826

36. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: IEEE CVPR. (2016) 770–778

37. Zhou, X., Huang, Q., Sun, X., Xue, X., Wei, Y.: Towards 3d human pose estimation in the wild: a weakly-supervised approach. In: IEEE ICCV. (2017)

38. Tompson, J.J., Jain, A., LeCun, Y., Bregler, C.: Joint training of a convolutional network and a graphical model for human pose estimation. In: NIPS. (2014) 1799–1807

39. Maurer, C.R., Qi, R., Raghavan, V.: A linear time algorithm for computing exact euclidean distance transforms of binary images in arbitrary dimensions. IEEE Trans. Pattern Anal. Mach. Intell. **25**(2) (2003) 265–270

40. Sethian, J.A.: A fast marching level set method for monotonically advancing fronts. Proceedings of the National Academy of Sciences **93**(4) (1996) 1591–1595
41. Girshick, R.: Fast r-cnn. In: Proceedings of the IEEE international conference on computer vision. (2015) 1440–1448
42. Szegedy, C., Ioffe, S., Vanhoucke, V., Alemi, A.A.: Inception-v4, inception-resnet and the impact of residual connections on learning. In: AAAI. Volume 4. (2017) 12
43. Garcia-Hernando, G., Yuan, S., Baek, S., Kim, T.K.: First-person hand action benchmark with rgb-d videos and 3d hand pose annotations. In: The IEEE Conference on Computer Vision and Pattern Recognition (CVPR). (June 2018)
44. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: Towards real-time object detection with region proposal networks. In: NIPS. (2015)
45. Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., Zheng, X.: TensorFlow: Large-scale machine learning on heterogeneous systems (2015) Software available from tensorflow.org.
46. Kingma, D.P., Ba, J.: Adam: A Method for Stochastic Optimization. ArXiv e-prints (December 2014)