HandSeg: An Automatically Labeled Dataset for Hand Segmentation from Depth Images

Abhishake Kumar Bojja*, Franziska Mueller†, Sri Raghu Malireddi*,
Markus Oberweger‡, Vincent Lepetit‡, Christian Theobalt†,
Kwang Moo Yi*, and Andrea Tagliasacchi*
{abojja, raghu, kyi, ataiya}@uvic.ca
{frmueller, theobalt}@mpi-inf.mpg.de
{oberweger, lepetit}@icg.tugraz.at

∗-University of Victoria, †-MPI Informatics, ‡-TU Graz

Abstract. We propose an automatic method for generating high-quality annotations for depth-based hand segmentation, and introduce a large-scale hand segmentation dataset. Existing datasets are typically limited to a single hand. By exploiting the visual cues given by an RGBD sensor and a pair of colored gloves, we automatically generate dense annotations for two hand segmentation. This lowers the cost/complexity of creating high quality datasets, and makes it easy to expand the dataset in the future. We further show that existing datasets, even with data augmentation, are not sufficient to train a hand segmentation algorithm that can distinguish two hands. Source and datasets will be made publicly available.

1 Introduction

Hand gestures are a natural way for humans to interact with the surrounding environment, and as such, many researchers have focused on obtaining accurate hand poses [1, 2]. Recently, as depth cameras have become more accurate and affordable [3, 4], substantial progress has been made towards this goal [5–7]. In many cases, the first step in obtaining accurate poses of hands is to find *where* the hand is in the image, preferably as accurately and robustly as possible. In *hand segmentation*, the detection happens at pixel-level accuracy.

A number of heuristic solutions have been proposed to simplify the task of hand segmentation [6, 5, 8]. While these approaches are well suited for small-scale lab experiments, they do not possess the robustness required for a consumer-level solution that needs to work under the full diversity of interactions in general real-world scenes – the violation of one of their underlying assumptions results in immediate tracking failure. One could learn a hand segmenter from a dataset of annotated depth images. However, as we will show, the limited size and quality of currently available datasets results in segmenters that typically overfit to the training data, and do not generalize well to unseen scenarios. Due to the limited

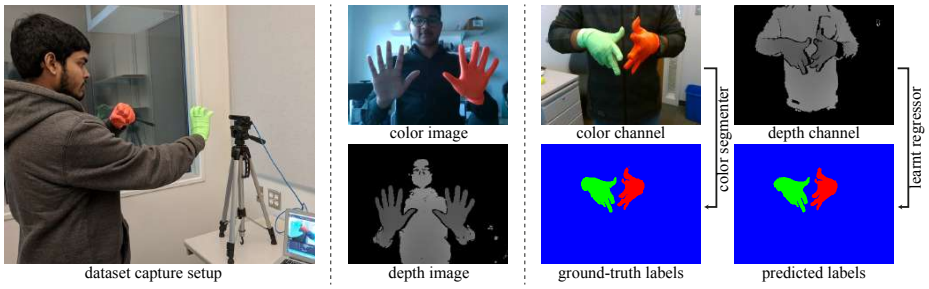

Fig. 1. Proposed data capture and automatic annotation framework. **(Left)** Our dataset is constructed by recording a user performing hand movements wearing a pair of brightly colored gloves in front of a depth camera. To the best of our knowledge, our dataset is the first *two-hand* dataset for hand segmentation. **(Middle)** The use of tight colored gloves provide a *quasi* non-invasive automatic annotation system, as the signal-to-noise ratio of a conventional depth sensor is not sufficiently high to distinguish between gloved and bare hands. **(Right)** Color images that are aligned with the depth images are exploited to automatically compute ground truth labels without user intervention. We then quickly filter out the few wrongly labeled images through human inspection. We can subsequently use these input-label pairs to train a depth-based semantic segmenter.

size of available datasets, the application of modern deep learning solutions to the problem of real-time hand segmentation has received limited attention.

Hence, a central goal is to capture a sufficiently *large* dataset equipped with *high-quality* ground truth annotations. To achieve this, we propose an automatic procedure to create high-quality per-pixel hand segmentation annotations from depth data, and introduce a large-scale dataset that we captured and annotated using the proposed method. As shown in Figure 1, we obtain this dataset by having a number of users perform hand gestures in front of an RGBD camera while wearing a pair of *colored gloves*. The color and depth channels are then used to generate high-quality ground truth annotations with minimal user intervention.

Note that the only additional equipment necessary for data acquisition is a pair of colored gloves, compared to the sophisticated setups used for hand capture (magnetic sensors [9] or optical IR markers [10]). Moreover, the quality of the dataset is much better than the ones that use motion capture sensors, as these methods require an additional heuristics to generate pixel-wise annotations for training a hand segmenter [11]. To the best of our knowledge, our dataset is the only one that provides both quality and quantity, with the quantity being orders of magnitude larger than what is currently available (see Table 1). We also provide an in-depth analysis of the effect of using our dataset on multiple neural network architectures for hand segmentation, as well as traditional Random Forests due to their computational efficiency. We empirically find that using *strided [transposed-]convolutions* in place of [un]pooling layers, and the use of skip-connections is essential for achieving high-accuracy. This further enables

Table 1. Existing and proposed datasets for *exocentric* hand segmentation from depth imagery. Our dataset is the only real dataset that distinguishes the two hands. Furthermore, our capture setup does not require expensive sensors as in the other two real datasets; see text for more details.

Dataset	Annotations	#Frames	#Subj	Hand	Sensor Type	Resolution
Freiburg [14]	synthetic	43,986	20	left/right	Unreal Engine	320 × 320
NYU [15]	automatic	6,736	2	left	Kinect v1	640 × 480
HandNet [11]	heuristic	212,928	10	left	RealSense SR300	320 × 240
Proposed	automatic	210,000	13	left/right	RealSense SR300	640 × 480

efficient forward-passes within $\approx 5ms$ on an NVIDIA Geforce GTX1080 Ti, making our approach suitable for real-time applications.

In the remainder of the paper we review related works (Sec. 2), and then detail our data acquisition setup and the annotation method (Sec. 3). We then discuss the methods we evaluate on our dataset, as well as suggest an empirically well performing deep network architecture (Sec. 4). We conclude by further detailing our experimental results (Sec. 5), and suggesting avenues for future works (Sec. 6).

2 Related works

We now introduce several heuristics that have been proposed for real-time hand segmentation, describe existing datasets, and overview techniques for semantic segmentation. For references on hand tracking, see [9].

2.1 Heuristics for hand segmentation

The pioneering approach of [12] leverages skin color segmentation and requires the user to wear long sleeves and to keep their face out of sight. Melax et al. [13] exploited short-range depth sensors by assuming that everything within the camera field of view is to be tracked, while Oberweger et al. [6] expect the hand to be the closest object to the camera. Some methods identify the ROI as the portion of the point cloud attached to the wrist, where this can be identified either with the help of a colored wristband [5], or by querying the wrist position in a full-body tracker [8]. As discussed, these heuristics fail as soon as their underlying assumptions are violated.

2.2 Datasets for hand segmentation

Datasets for hand segmentation from color images were previously proposed by [16] and [17] who provided pixel-level manually annotated ground truth for

respectively ≈ 500 and $\approx 15k$ color images. Manual annotation of segmentation masks from color images is extremely labor intensive. This not only makes it very difficult to collect large-scale datasets, but the quality of annotations also depends on the skills of the individual annotator. Gathering bounding-box annotations is easier, as demonstrated by the datasets of ≈ 500 annotated images in [18], or the $\approx 15k$ images in [19]. However, these annotations are too coarse for applications that require accurate hand/background or hand/object segmentation.

Automatic segmentation.

Hand segmentation can be cast as a skin color segmentation problem [14]. However, segmenting this not only detects hands but also other skin regions, such as faces or forearms when the user is not wearing sleeves. Further, datasets of this kind [20, 21] contain at most a few thousand manual annotations, which is magnitudes smaller than what is needed to train deep neural networks. Zimmermann et al. [14] recently proposed a dataset with $\approx 44k$ *synthetic* images. However, it is notoriously difficult to accurately model skin colors in unconstrained lighting settings considering complex effects like subsurface scattering, making it challenging to develop segmentation methods that could work in the wild. Conversely, hand segmentation from *depth* images does not suffer this problem. Tompson et al. [15] pioneered this approach and *painted* each user hand with bright colors which are segmented and post-processed with the help of depth information. However, while [15] contains $\approx 70k$ marker-annotated frames from three viewpoints, only $\approx 7k$ are provided with annotations suitable for hand segmentation. Furthermore, this dataset has been acquired with a Kinect v1 sensor, which is now deprecated for hand tracking – its *long-range* configuration and its use of *spatial structured light* results in a loss of small geometric features (e.g. fingertips) from the estimated depth map.

Segmentation via tracking. Recent datasets targeting hands have mostly focused on acquiring annotated 3D marker locations for joints [9]. Creating datasets via manual annotation is not only labor-intensive [22], but placing markers within a noisy depth map often results in inaccurate labels. Assuming marker locations are correct, simple heuristics can be employed to infer a dense labeling. Following this idea, Wetzler et al. [11] first employ a complex/invasive hardware setup comprising of *magnetic sensors* attached to fingertips to acquire their locations, and obtain the segmentation mask via a depth-based flood-fill. While the dataset by [11] contains $\approx 200k$ annotated exemplars, these heuristic annotations should not be considered to be ground truth for learning a high-performance segmenter; see our evaluations in Section 5.

2.3 Semantic segmentation

Recently, neural networks have been successfully applied to the problem of semantic segmentation of a broad range of real world objects and scenes. Popular methods include fully convolutional neural networks [23], which encode the input to a low-dimensional latent space, and decode via bilinear upsampling

to predict the semantic segmentation. Follow-up works perform learning at the decoder level as well, such as the well known *DeconvNet* [24] and *SegNet* [25] architectures; see Section 4. Learned encoder-decoder architectures have been shown to perform well on semantic segmentation [26–29], but when fast inference time is essential, random forests are an excellent alternative due to their easy parallelization [30, 31]. In human pose estimation applications, [32] inferred body part labels via random forests, which was later adopted for hand localization from depth images by [15], and color images by [14]. Recently, [33] employed a convolutional neural network to estimate two-hand segmentation masks for hand tracking. In multi-view setups, effective segmentation provides a strong cue for effective tracking [34], and the two tasks can even be coupled into a single optimization problem [35].Predicted segmentation masks can be noisy and/or coarse, and post-processing is typically employed to remove outliers by regularizing the segmentation [36].A recent approach by [37] accounts for the severity of mis-labeling by a loss encoding their spatial distribution, but this method has yet to be generalized to a multi-label classification scenario like ours. Relevant to our work is also the recent R-CNN series of works, of which the instance segmentation work by [38] represents the latest installment. While combining bounding box localization with dense segmentation could be effective, it is however unclear to which extent such networks could be adapted to demanding real-time applications such as hand tracking.

3 Data acquisition and automatic annotation

For scalable annotation with minimal human interaction, we rely on the synchronized color/depth input of an RGBD device, and a pair of skin-tight brightly colored gloves. As shown in Figure 1(middle), this allows a (quasi) non-invasive and cost effective setup, where we can automatically determine ground-truth labels at pixel level. As the gloves fit the user's hand tightly, minimal geometric aberration to the depth map occurs, while the consistent color of the glove can be used to extract the hand ROI via color segmentation. After an initial color calibration session, we ask the user to perform a few motions according to the *protocol* described below, and record sequences of (depth,color) image pairs at a constant 48Hz rate with an Intel RealSense SR300. We then execute a color segmentation to generate masks with a very small false positive rate; we finally quickly discard contiguous frames containing erroneous labels via manual inspection of video – this task is *significantly* simpler than manually editing individual images. In our process we roughly drop 10% of the automatically labeled images, selected conservatively to avoid any wrong label in the dataset.

3.1 Acquisition protocol

Similarly to [9], we attempt to maximize the coverage of the articulation space by asking each user to assume a number of example extremal poses, while capturing

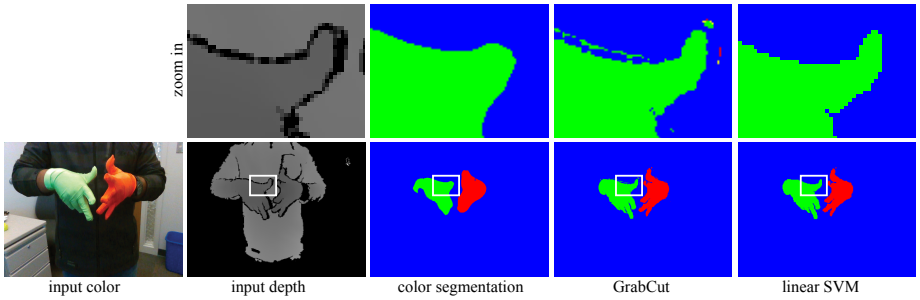

Fig. 2. Our automatic annotation pipeline. **(Bottom)** input images, the output of each segmentation step. **(Top)** Zoom in to highlight segmentation accuracy. We employ the color image to create ground truth annotations for depth. We first segment the color image via HSV thresholding, then perform GrabCut [39] to obtain better segmentation. We finally train a per-image linear support vector machine (SVM) with RGB, HSV, XYZ, Lab color spaces, as well as image coordinates as input cues to further refine the annotation. Note how the segmentation becomes more accurate as each step is performed. Through this three-stage process we are able to obtain highly accurate ground-truth annotations without manual annotations. See text for details.

the natural motion during each transition. We further move the camera during the capture process to enrich the dataset with various viewpoints. Our dataset includes complex poses, such as the ones shown in Figure 6, where fingers are overlapping with each other.

3.2 Automatic label generation

As illustrated in Figure 2, we perform color segmentation through a three-stage procedure. The quality of the labeling is enhanced at each stage of the pipeline.

Initial color segmentation. We first perform color space thresholding to obtain a rough segmentation S_r and S_l of the two hands, where r and l denote right and left hands respectively. We will denote both S_r and S_l together as S_*. Specifically, to obtain S_*, we threshold on the HSV color space after smoothing the input image with a Gaussian kernel (with standard deviation 30) to remove noise. The threshold values we use for our experiments are minimum and maximum values of $[3, 160, 100]$–$[15, 255, 255]$ for left hand, and $[28, 35, 100]$–$[70, 200, 255]$ for right hand, where [H, S, V] denotes the HSV values and $H \in [0, 180]$ while $S, V \in [0, 255]$.

Refinement through GrabCut [39]. As the initial segmentation is coarse due to the initial Gaussian filtering, we further apply GrabCut [39], followed by a linear support vector machine (SVM) classifier [40] to get a more fine-grained segmentation map; see Figure 2. We determine the seed points for GrabCut by first finding the enclosing rectangles R_* for each hand, and then using all points of S_* that are inside R_*; to be robust to noise, we enlarge R_* by 10 percent. At

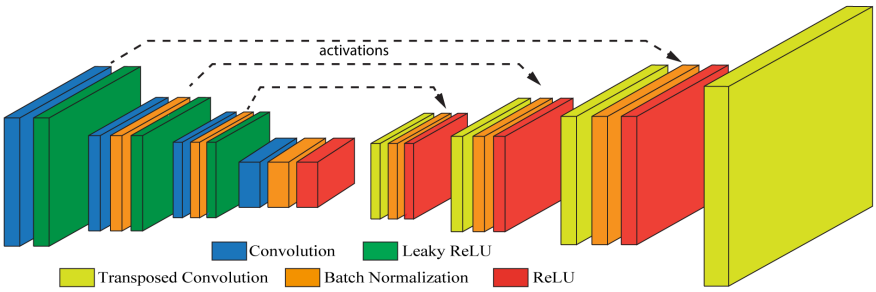

Fig. 3. The architecture of our hand segmentation CNN. Note that in our architecture we do not have any pooling/unpooling layers. Instead, we use strided convolutions in the encoder and strided transposed convolutions in the decoder.

this stage, some of the labels are still inaccurate, especially near the boundaries of the hands.

Refinement through linear SVM [40]. To further enhance the labels, we exploit the high distinctiveness of the glove's color, and train a linear SVM classifier *per image* with a large enough margin, and use the positives that are classified as positives during training as ground-truth labels. Note that this classifier is simply a per-image refinement process that automatically sets-up the per-image thresholds for a simple colour-based thresholding system, based on the GrabCut results. For robust performance we use RGB, HSV, XYZ and Lab color values as well as image coordinates as cues to linear SVM. We also empirically set the hyper-parameter $C = 900$ (margin strength).

4 Learning to segment hands

We now detail the segmentation techniques we evaluate on our dataset in Section 5. We investigate Random Forests as a representative *shallow* method (Sec. 4.1), and multiple deep architectures (Sec. 4.2).

4.1 Random Forests

Our first baseline is the *shallow* learning offered by Random Forests popularized for full-body tracking by [41]. Tompson et al. [15] pioneered its application to binary segmentation of one hand, while Sridhar et al. [7] extended the approach to also learn more detailed part labels (e.g. palm/phalanx labels to guide articulated registration). Analogously to [41, 7], our forest consists of 3 trees each of depth 22, and uses the typical depth differential features proposed by [41, Eq.1]. At inference time, Random Forests are highly efficient, making them suitable to applications like *real-time* tracking. However, while their optimal parameters (offset/threshold) are learned, the features themselves are fixed, and this can result in overall lower accuracy when compared to deep architectures.

4.2 Deep convolutional segmenters

We evaluate several recently proposed deep learning convolutional architectures, as well as propose a novel variant with enhanced forward-propagation efficiency and precision. As we have a multi-class labeling problem, we employ the soft-max cross entropy loss. In all our experiments we train our networks with ADAM optimizer with default parameters, and with appropriate learning rates between 10^{-4} and 10^{-6}, depending on the architecture. To prevent over-fitting, we apply early stopping according to the accuracy of the models on the validation set. To further improve the generalization capacity of the deep networks, we apply random data *augmentation* by: randomly flipping the images horizontally as well as the left/right labels; randomly translating the depth images horizontally and vertically by 20% proportionally to the input size; and randomly scaling the images in the range of 20% (log scale). We further normalize the input depth image so that the measured signals have a unit average.

Fully convolutional neural network (FCN). Long et al. [23] proposed an architecture where a coarse segmentation mask is produced via a series of convolutions and max-pooling stages (*encoder*), where the low-resolution image is then upsampled (*decoder*) via bilinear interpolation – the *FCN32s* variant in [23, Fig.3]. As this process produces a blurry segmentation mask, a sharper mask can be obtained by combining this image with the higher-resolution activations from earlier layers in the network; the *FCN16s* and *FCN8s* variants. Unfortunately, the initial layers in the network only encode very localized features. Hence while this process does produce sharper results, it also introduces high-frequency misclassifications in uncertain regions. Another problem of FCN is their difficulty in dealing with the problem of *class imbalance*: in our training images, the cardinality of background pixels is significantly larger than the one of hand pixels. We overcome this problem by incorporating the class frequency in the loss [42, 43], which effectively prevents the network from converging to one that trivializes the output to be always classified as background. Even with these changes, the limited accuracy achieved by this network can be understood by noting that the encoder layer is learned, while the decoder layer is not.

Learned encoder-decoder networks. The popular *SegNet* [25] and *DeconvNet* [24] semantic segmentation networks follow an encoder-decoder architecture. Similarly to FCNs, the encoder is realized via a sequence of convolutions and max-pooling operations. However, rather than relying on interpolation, the decoder used to generate high-resolution segmentations is also learned. Both architectures employ an *unpooling* operation that inverts the *max-pooling* in the encoder. Similarly to DeconvNet, SegNet upsamples the feature maps via memorized max-pooling indices in the corresponding encoder layer. Further, while unpooling in SegNet is followed by a simple series of convolutions, DeconvNet employs a series of *transposed convolution* layers. Transposed convolutions, coined "deconvolutions" in [24], invert the convolution process, and combined with strides are an effective way to create a feature map that is larger than the one in input. This allows learning a segmenter with a full image, and to create a bottleneck layer that

encodes the dataset manifold. However, these architectures lack skip connections, and thus require large numbers of intermediate channels to preserve information when downsampling and upsampling. Thus, they are computationally intensive to both train and test, as we show later in the experiments, while performing worse than our architecture.

Proposed architecture. We empirically found that the best performing architecture is a *hybrid* encoder-decoder, see Figure 3: we employ a hierarchy of transposed convolution layers (a la DeconvNet), and to improve sharpness and local detail of our predictions, without having the need for excessive amount of hidden neurons, we forward information from encoder to decoder through skip-connections (a la U-Net [44]). Differently from other architectures, note how our encoders/decoders do not contain any pooling/unpooling layer. Pooling layers are useful in classification tasks as they provide invariance to local deformations, which is exactly the opposite of what we would like in our case, that is, a segmentation output that is pixel-level accurate. Nonetheless it is critical to have downsampling and upsampling for efficiency and to incorporate context in estimating the label for each pixel. In our encoder network, this is achieved by *strides*. For the decoder network, we symmetrically employ *transposed convolution* layers with strides. This enables the network to learn an appropriate upsampling filter. The simplicity in our design results in *efficient* forward propagation, while simultaneously achieving state-of-the-art *accuracy*; see Figure 4 and Table 2

5 Evaluation

We quantitatively evaluate our dataset with various methods from three different angles. In Section 5.1, we evaluate how different methods perform on our data in terms of mean Intersection over Union (mIoU), as well as their runtime both during training and testing. In Section 5.2, we show the generalization capabilities of several datasets, including ours. In all our experiments, the dataset was split randomly in a 8:1:1 ratio to form train, validation, and test sets.

Evaluation metrics.

In our multi-label classification problem, each pixel can be classified as {left, right, background}. Within each class, we can have *true-positives* (TP), *false-positives* (FP) and *false-negatives* (FN). Given such a categorization, we use the *Intersection over Union*, defined as IoU $= |TP| \,/\, (|TP| + |FP| + |FN|)$, for quantitative evaluation. As in [45], to aggregate results for multiple classes, we use the class-wise average among classes, that is, mean IoU (mIoU). This is to account for the imbalance in the number of pixels for each class.

5.1 Segmenting with different architectures

In Figure 4, we compare the different learning approaches in terms of accuracy, and their runtime in Table 2. For the runtime experiments, all deep networks

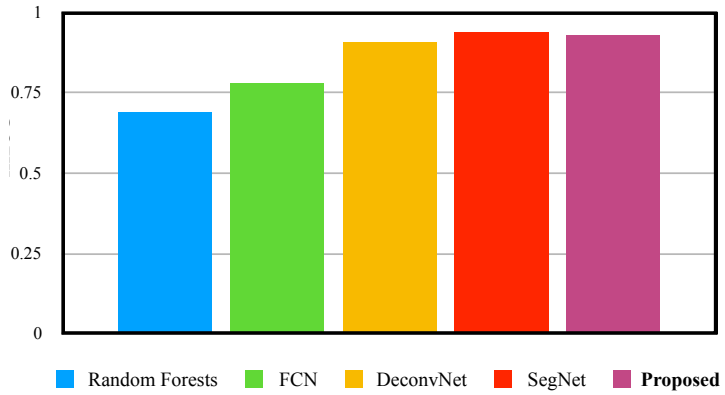

Fig. 4. Performance of different segmentation methods on our dataset in terms of mIoU (higher is better). Evaluation is performed on a two-class setup, where left and right hands are not distinguished. Otherwise, DeconvNet and SegNet fail to learn. However, the proposed network is able to achieve state of the art in any case.

Table 2. Runtime of each segmentation method. Ours is the fastest to train and test amongst compared deep architectures.

	Random Forests	FCN	DeconvNet	SegNet	**Proposed**
Train time	3h	149h	57h	83h	29h
Test time	1ms	41ms	16ms	30ms	5ms

were run on a single NVIDIA Geforce GTX 1080 Ti graphics card. In these experiments, we did not distinguish between left and right hands, as DeconvNet and SegNet completely failed due to the class imbalance between left hand, right hand, and background labels. Although Random Forests are clearly the fastest to train and to infer on, they perform poorly when compared to deep networks. Due to its simple upsampling scheme, FCN(32s) performs the worst among the evaluated networks. Thanks to its learned decoder network, DeconvNet and SegNet obtain much better results. However, their architectures are too computationally complex, resulting in a runtime that is not suitable for real-time tracking applications when considering that segmentation is typically a pre-processing step for a sophisticated vision pipeline. Our proposed architecture not only on par with the best performing method in terms of accuracy, but it is also fast to forward-propagate, running at $\approx$ 200fps. Furthermore, as noted previously, when we train networks to distinguish between left and right hands, our architecture, HandSeg, is the only network among the best three that gives any usable result (mIoU 0.877, as shown in Table 3).

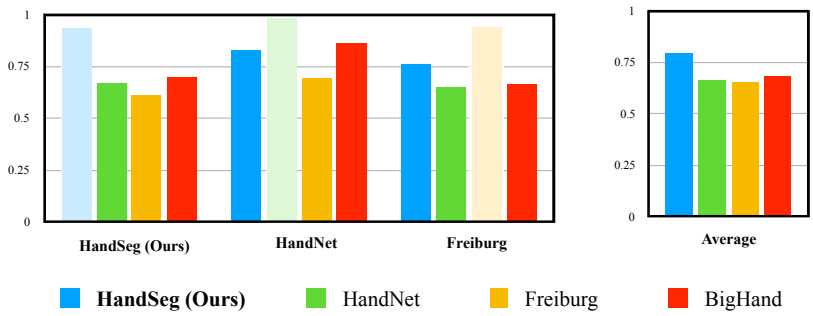

Fig. 5. Generalization performance across datasets for the **two-class setup**, in terms of mIoU. The dataset used for training is color-coded by the legend at the bottom, and the results are grouped by each test set. The washed-out colors denote the case when trained and tested on the same dataset. On the right, we show the average performance of segmenters trained on each dataset, when tested on other datasets excluding the one used for training. Note that the segmenter trained on our dataset, *HandSeg*, generalizes best on average and on Freiburg, and is on par with the best generalizing dataset on HandNet.

5.2 Cross-dataset evaluation

We test our baseline network by training/testing on all possible combinations of the datasets in Table 1 (with the exception of NYU which is captured using a deprecated sensor). We also include BigHands [9] as a dataset for training, which is a hand pose dataset composed of more than a million images. However, as this dataset is originally intended for hand pose estimation, it does not have per pixel labels. We therefore apply GrabCut [39] with the hands' joint locations as seed points to obtain a rough segmentation label. As these labels are not perfect, this dataset cannot be used for testing. As not all datasets distinguish left and right hands, we perform evaluations for the two-class (hands vs. background) as well as the three-class (left vs. right vs. background) scenario. We summarize the results in Figure 5 and in Table 3, respectively.

As shown in Figure 5, when left and right hands are not distinguished, the segmenter trained with our dataset generalizes better (in average) than when trained with other datasets. Furthermore, as shown in the results on each testing dataset, the segmenter trained on our dataset performs either the best or comparably to the best method, while simultaneously generalizing to unseen datasets.

In Table 3, we show the case when the two hands are distinguished. This three-class case is harder than the two-class case above, as the segmenter now has to distinguish between left and right hands. As shown, the segmenter trained with our dataset generalizes better to Freiburg, than the one trained with Freiburg on ours. Considering that the test performance on both datasets is similar, this shows the better generalization capability of our dataset. Furthermore, as the

Table 3. Generalization performance across datasets for the **three-class setup**, in terms of mIoU. For BigHands, we use data augmentation to generate both left and right hand labels. Segmenter trained on our dataset, *HandSeg*, performs best in terms of generalization.

Test \ Train	**HandSeg** (Ours)	Freiburg	BigHands
HandSeg (Ours)	0.877	0.437	0.492
Freiburg	0.574	0.870	0.408

BigHands dataset only features a single hand, data augmentation needs to be applied for the three-class setup, which is the result shown in Table 3. The poor numbers clearly demonstrate the need of a hand segmentation dataset that distinguishes left/right hands.

5.3 Qualitative evaluation

In Figure 6, we provide qualitative segmentation results on our novel dataset. Here, we show results of the proposed architecture, FCN, and the Random Forest. We excluded SegNet and DeconvNet, as for the three-class experiments, these two network architectures failed to deliver any meaningful results on our dataset and converged to a trivial solution, that is, all pixels considered as background. Note how the proposed architecture shows the best performance.

Figure 7 shows challenging frames where our network does not deliver perfect results. Sample #1 illustrates how the network can still segment the hands of multiple persons, although it was trained on frames containing a single individual. This reveals the generalization capabilities of our network, which did not only learn to segment *one/two* regions, but also learned a latent *shape-space* for human hands. Sample #2 shows a person holding a cup, while Sample #3 and Sample #4 have the hand lying flat on the body. These scenarios are difficult, as the network has never seen a hand interacting with objects. Although not perfect, the network successfully segments the hands in Samples #3 and #4, but fails on the cup for Sample #2. Accuracy could be improved by accounting for the additional information in the color channel, or by learning the appearance of the object via training examples.

6 Conclusions and future works

We have proposed an automatic annotation method for easily creating hand segmentation datasets with an RGBD camera, and have introduced a new high-quality dataset for hand segmentation that is significantly larger than what is currently available. Our annotation method requires minimal human interaction,

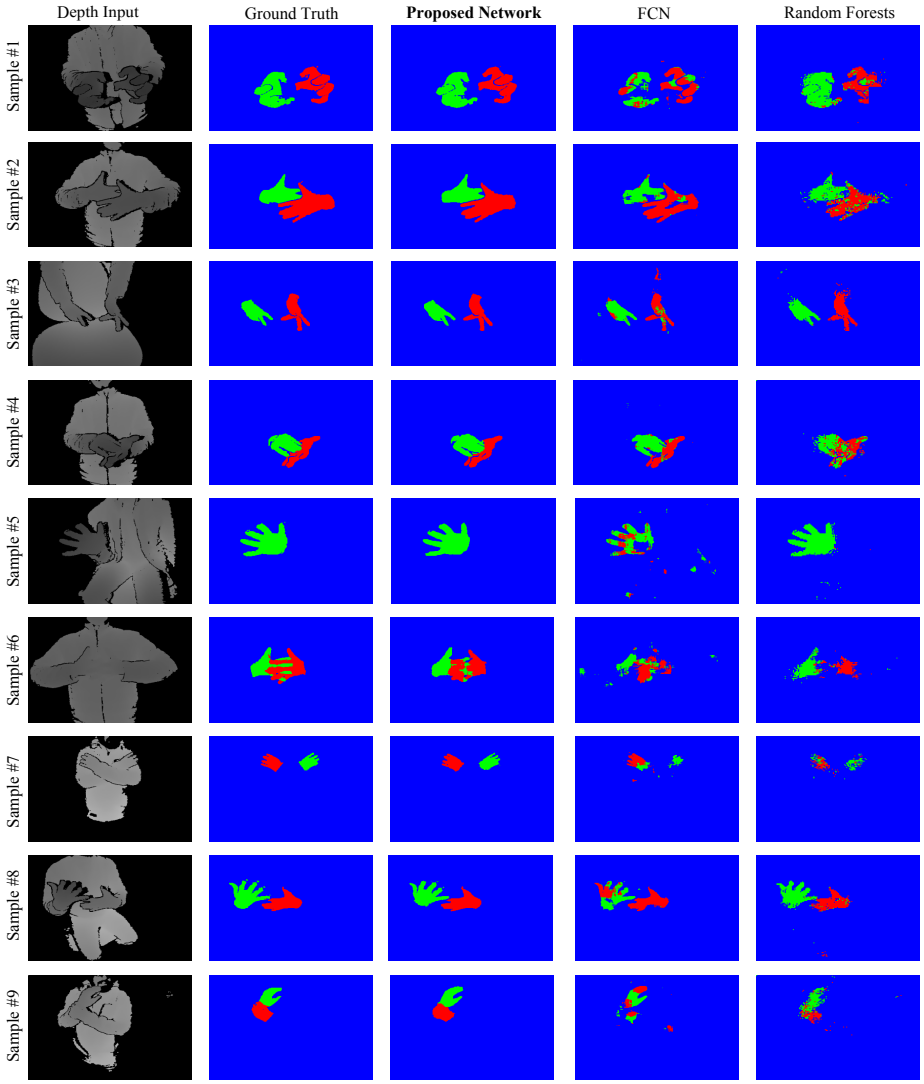

Fig. 6. Qualitative examples. We illustrate a few examples of hand segmentation performance on our dataset for the proposed network, FCN, and Random Forests. We exclude SegNet and DeconvNet here as they converge to estimating all pixel as the background for this setup. Note how the proposed network gives accurate segmentation for diverse poses, including when the hands are interacting as shown in Sample #4. Sample #6 shows a failure case of our network when there's extreme interaction between the two hands. Still, our architecture performs better than the compared ones, giving relatively accurate segmentation.

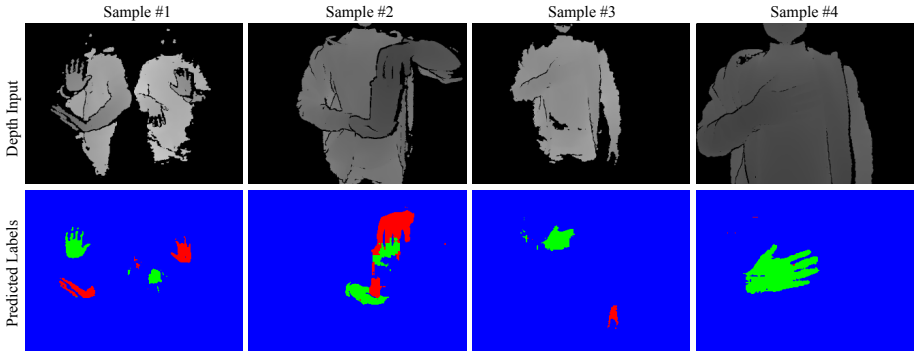

Fig. 7. A selection of segmentation failure cases. Due to the challenging nature of these examples, our segmenter does not return perfect results. Note that in Sample #1, our network is able to segment all four hands, although it was never trained with more than a single person in the field of view. In Sample #2, our network show error on the cup, as the network never saw hands interacting with objects.

and is highly cost effective. With the proposed method, we have created a dataset that contains high-accuracy dense pixel annotations, large pose variations, and many different subjects. Our results show that the new dataset, HandSeg, allows training of segmenters that are more general than the ones trained with existing datasets.

Our analysis has also revealed poor generalization characteristics for currently available methods. With the Microsoft Kinect v1 sensor being retired from production, this creates an immediate problem as the only high-quality (albeit small) dataset for the task at hand [15] becomes unusable. Conversely, our data is acquired on Intel RealSense SR300 sensors, one of the most commonly employed sensors available. Beyond these immediate needs, it would also be interesting to see whether simultaneously training on multiple datasets could generate architectures that are apt to transfer learning. While eventually the use of (very large) synthetic datasets like [14] could be very effective for training, the proposed HandSeg dataset will remain valuable for validation/testing.

We also propose a segmentation network that is faster than existing baselines, and provides superior mIoU accuracy. While these results are encouraging, our dataset opens new frontiers for investigation, such as the effectiveness of spatially-aware losses [37], the use of efficient quantized networks [46], or its use for weak-supervision of discriminative hand tracking [47].

References

1. Erol, A., Bebis, G., Nicolescu, M., Boyle, R.D., Twombly, X.: Vision-based hand pose estimation: A review. In: CVIU. (2007)
2. Supancic, J.S., Rogez, G., Yang, Y., Shotton, J., Ramanan, D.: Depth-based hand pose estimation: data, methods, and challenges. In: ICCV, IEEE (2015) 1868–1876
3. Keselman, L., Woodfill, J.I., Grunnet-Jepsen, A., Bhowmik, A.: Intel realsense stereoscopic depth cameras. arXiv (2017)
4. Fanello, S.R., Valentin, J., Rhemann, C., Kowdle, A., Tankovich, V., Izadi, S.: Ultrastereo: Efficient learning-based matching for active stereo systems. CVPR (2017)
5. Tagliasacchi, A., Schroeder, M., Tkach, A., Bouaziz, S., Botsch, M., Pauly, M.: Robust articulated-icp for real-time hand tracking. CGF (2015)
6. Oberweger, M., Wohlhart, P., Lepetit, V.: Hands deep in deep learning for hand pose estimation. In: Proc. Computer Vision Winter Workshop. (2015)
7. Sridhar, S., Mueller, F., Oulasvirta, A., Theobalt, C.: Fast and robust hand tracking using detection-guided optimization. In: CVPR. (2015)
8. Sharp, T., Keskin, C., Robertson, D., Taylor, J., Shotton, J., Kim, D., Rhemann, C., Leichter, I., Vinnikov, A., Wei, Y., et al.: Accurate, robust, and flexible real-time hand tracking. In: of ACM CHI. (2015)
9. Yuan, S., Ye, Q., Stenger, B., Jain, S., Kim, T.K.: Bighand2.2m benchmark: Hand pose dataset and state of the art analysis. In: CVPR. (2017)
10. Han, S., Liu, B., Wang, R., Ye, Y., Twigg, C.D., Kin, K.: Online optical marker-based hand tracking with deep labels. ACM TOG (2018)
11. Wetzler, A., Slossberg, R., Kimmel, R.: Rule of thumb: Deep derotation for improved fingertip detection. In: BMVC. (2015)
12. Oikonomidis, I., Kyriazis, N., Argyros, A.A.: Efficient model-based 3D tracking of hand articulation using kinect. In: BMVC. (2011)
13. Melax, S., Keselman, L., Orsten, S.: Dynamics based 3d skeletal hand tracking. In: Proc. of GI. (2013)
14. Zimmermann, C., Brox, T.: Learning to estimate 3d hand pose from single rgb images. In: ICCV. (2017)
15. Tompson, J., Stein, M., Lecun, Y., Perlin, K.: Real-time continuous pose recovery of human hands using convolutional networks. ACM TOG (2014)
16. Buehler, P., Everingham, M., Huttenlocher, D.P., Zisserman, A.: Long term arm and hand tracking for continuous sign language tv broadcasts. In: BMVC. (2008)
17. Bambach, S., Lee, S., Crandall, D.J., Yu, C.: Lending a hand: Detecting hands and recognizing activities in complex egocentric interactions. In: ICCV. (2015)
18. Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results (2012)
19. Mittal, A., Zisserman, A., Torr, P.H.S.: Hand detection using multiple proposals. In: BMVC. (2011)
20. del Solar, J.R., Verschae, R.: Skin detection using neighborhood information. In: Automatic Face and Gesture Recognition. (2004)
21. Kawulok, M., Kawulok, J., Nalepa, J.: Spatial-based skin detection using discriminative skin-presence features. Pattern Recognition Letters (2014)
22. Sridhar, S., Oulasvirta, A., Theobalt, C.: Interactive markerless articulated hand motion tracking using RGB and depth data. In: ICCV. (2013)
23. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: CVPR. (2015)

24. Noh, H., Hong, S., Han, B.: Learning deconvolution network for semantic segmentation. In: ICCV. (2015)
25. Badrinarayanan, V., Kendall, A., Cipolla, R.: Segnet: A deep convolutional encoder-decoder architecture for image segmentation. arXiv (2015)
26. Zhao, H., Shi, J., Qi, X., Wang, X., Jia, J.: Pyramid scene parsing network. In: CVPR. (2017)
27. Lin, G., Milan, A., Shen, C., Reid, I.: Refinenet: Multi-path refinement networks for high-resolution semantic segmentation. In: CVPR. (2017)
28. Pinheiro, P.O., Lin, T.Y., Collobert, R., Dollr, P.: Learning to refine object segments. In: ECCV. (2016)
29. Paszke, A., Chaurasia, A., Kim, S., Culurciello, E.: Enet: A deep neural network architecture for real-time semantic segmentation. In: arXiv. (2016)
30. Kontschieder, P., Bul, S.R., Bischof, H., Pelillo, M.: Structured class-labels in random forests for semantic image labelling. In: ICCV. (2011)
31. Shotton, J., Johnson, M., Cipolla, R.: Semantic texton forests for image categorization and segmentation. In: CVPR. (2008)
32. Shotton, J., Sharp, T., Kipman, A., Fitzgibbon, A., Finocchio, M., Blake, A., Cook, M., Moore, R.: Real-time human pose recognition in parts from single depth images. Comm. ACM (2013)
33. Taylor, J., Tankovich, V., Tang, D., Keskin, C., Kim, D., Davidson, P., Kowdle, A., Izadi, S.: Articulated distance fields for ultra-fast tracking of hands interacting. ACM Transactions on Graphics (TOG) **36** (2017) 244
34. Liu, Y., Gall, J., Stoll, C., Dai, Q., Seidel, H.P., Theobalt, C.: Markerless motion capture of multiple characters using multiview image segmentation. PAMI (2013)
35. Kohli, P., Rihan, J., Bray, M., Torr, P.H.S.: Simultaneous segmentation and pose estimation of humans using dynamic graph cuts. IJCV (2008)
36. Chen, L.C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L.: Semantic image segmentation with deep convolutional nets and fully connected crfs. In: ICLR. (2015)
37. Kolkin, N., Shakhnarovich, G., Shechtman, E.: Training deep networks to be spatially sensitive. In: ICCV. (2017)
38. He, K., Gkioxari, G., Dollár, P., Girshick, R.B.: Mask R-CNN. In: ICCV. (2017)
39. Rother, C., Kolmogorov, V., Blake, A.: Grabcut: Interactive foreground extraction using iterated graph cuts. In: ACM TOG. (2004)
40. Fan, R.E., Chang, K.W., Hsieh, C.J., Wang, X.R., Lin, C.J.: LIBLINEAR: A library for large linear classification. JMLR **9** (2008) 1871–1874
41. Shotton, J., Fitzgibbon, A., Cook, M., Sharp, T., Finocchio, M., Moore, R., Kipman, A., Blake, A.: Real-time human pose recognition in parts from single depth images. In: CVPR. (2011)
42. Mostajabi, M., Yadollahpour, P., Shakhnarovich, G.: Feedforward semantic segmentation with zoom-out features. In: CVPR. (2015)
43. Xu, J., Schwing, A.G., Urtasun, R.: Tell me what you see and i will show you where it is. In: CVPR. (2014)
44. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: arXiv. (2015)
45. Zhou, B., Zhao, H., Puig, X., Fidler, S., Barriuso, A., Torralba, A.: Scene parsing through ade20k dataset. In: CVPR. (2017)
46. Hubara, I., Courbariaux, M., Soudry, D., El-Yaniv, R., Bengio, Y.: Binarized neural networks. In: NIPS. (2016)
47. Neverova, N., Wolf, C., Nebout, F., Taylor, G.W.: Hand pose estimation through semi-supervised and weakly-supervised learning. CVIU (2017)