Hand Pose Estimation through Semi-Supervised and Weakly-Supervised Learning

Natalia Neverova[a,*], Christian Wolf[a], Florian Nebout[b], Graham W. Taylor[c]

[a] *Université de Lyon, INSA-Lyon, CNRS, LIRIS, F-69621, France*
[b] *Awabot SAS, France*
[c] *School of Engineering, University of Guelph, Canada*

Abstract

We propose a method for hand pose estimation based on a deep regressor trained on two different kinds of input. Raw depth data is fused with an intermediate representation in the form of a segmentation of the hand into parts. This intermediate representation contains important topological information and provides useful cues for reasoning about joint locations. The mapping from raw depth to segmentation maps is learned in a semi- and weakly-supervised way from two different datasets: (i) a synthetic dataset created through a rendering pipeline including densely labeled ground truth (pixelwise segmentations); and (ii) a dataset with real images for which ground truth joint positions are available, but not dense segmentations. Loss for training on real images is generated from a patch-wise restoration process, which aligns tentative segmentation maps with a large dictionary of synthetic poses. The underlying premise is that the domain shift between synthetic and real data is smaller in the intermediate representation, where labels carry geometric and topological meaning, than in the raw input domain. Experiments on the NYU dataset [1] show that the proposed training method decreases error on joints over direct regression of joints from depth data by 15.7%.

Keywords: Hand pose estimation, Deep learning, Semantic segmentation

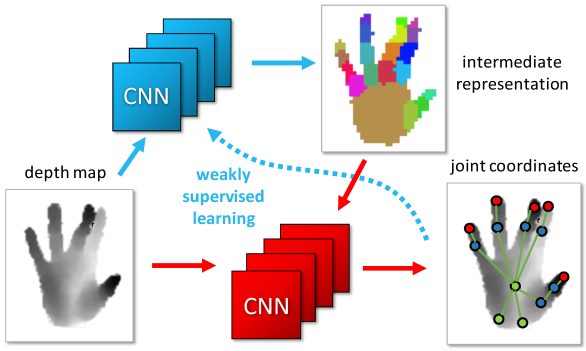


Figure 1: A rich intermediate representation is fused with raw input for hand joint regression. The intermediate representation is learned in a semi/weakly-supervised setting from real and synthetic data (see Figure 3 for an illustration of the training procedure). The input image is from the NYU dataset [1].

1. Introduction

Hand pose estimation and tracking from depth images, i.e. the estimation of joint positions of a human hand, is a first step for various applications: hand gesture recognition, human-computer interfaces (moving cursors and scrolling documents), human-object interaction in virtual reality settings and many more. While real-time estimation of full-body pose is now available in commercial products, at least in cooperative environments [2], the estimation of hand pose is more complex. In situations where the user is not directly placed in front of the computer, and therefore not close to the camera, the problem is inherently difficult. In this case, typically the hand occupies only a small portion of the image, and fingers and finger parts are only vaguely discernible.

Existing discriminative solutions to articulated pose from depth maps (see Section 2 for a full description of related work) have concentrated on two different strategies: (i) early work first contructed an intermediate representation of the body or hand into parts, from which joints were then estimated in a second step [2, 3]; (ii) subsequent work proceeded by direct regression from depth either through heat maps or to joint coordinates [1, 4, 5, 6, 7, 8, 9, 10].

There is clear interest in the second strategy, i.e. direct regression of joint positions, with a large body of recent work pointing in this direction. One of the reasons is that intermediate representations are only really efficient when training is restricted to synthetic data, where intermediate labels can be easily obtained. On real data, precise and dense annotations like part segmentations are difficult to come by, with the exception of finger painting datasets [11]. On the other hand, the amount of information passed to the training algorithm is significantly higher in the case of the intermediate segmentation: several bits *per pixel* of the input image vs. 2 or 3 real values *per joint and per image*. In this paper we argue that this advantage is important, and we propose a new model for regression as well as a semi-supervised and weakly-supervised training algorithm which allows us to extract this information automatically from real data.

The novelty of our approach compared to existing work lies in the integration of the intermediate representation as a *latent*

*Corresponding author (currently at Facebook AI Research, Paris)
Email address: neverova@fb.com (Natalia Neverova)

variable during training: no dense annotation on real data is required, only joint positions are needed.

In our target configuration, pose estimation is performed frame-by-frame without any dynamic information. A model is learned from two training sets: (i) a (possibly small) set of real images acquired with a consumer depth sensor. Ground truth joint positions are assumed to be available, for instance obtained by multi-camera motion capture systems. (ii) a second (and possibly very large) set of synthetic training images produced from 3D models by a rendering pipeline, accompanied by dense ground truth in the form of a segmentation into hand parts. This ground truth is easy to come by, as it is usually automatically created by the same rendering pipeline. The main arguments we develop are the following:

- an intermediate representation defined as a segmentation into parts contains rich structural and topological information, since the label space itself is structured. Labels have adjacency, topological and geometric relationships, which can be leveraged and translated into loss for weakly-supervised training;
- a regression of joint positions is easier and more robust if the depth input is combined with a rich semantic representation like a segmentation into parts, provided that this semantic segmentation is of high fidelity (see Figure 1).

We show that the additional information passed to the training algorithm is able to improve pose regression performance. A key component to obtaining this improvement is obtaining reliable segmentations for real data. While purely supervised training on synthetic data has proven to work well for full-body pose estimation [2, 3], hand pose estimation is known to require real data captured from depth sensors for training [12, 13, 1] due to low input resolution and data quality. Manually annotating dense segmentations of large datasets is not an option, and estimating segmentation maps from ground truth joint positions is unreliable in the case of low quality images. We propose a semi/weakly supervised setting in order to tackle this problem, where this intermediate representation is learned from densely labeled synthetic depth images as well as from real depth images associated with ground truth joint positions.

In particular, the proposed training method exploits the rich geometrical and topological information of the intermediate representation. During the training process, predicted segmented patches from real images are aligned with a very large dictionary of labelled patches extracted from rendered synthetic data. The novelty here lies in the fact that we do not match input patches but patches taken from the intermediate representation, which include the to-be-inferred label and its local context.

We call the proposed training method *weakly supervised*; since part of the ground truth data only contains joint locations and not the dense labels of part segmentations used during inference. The method is also *semi-supervised*, as part of the dataset is fully labeled, while another part is not.

The paper is organized as follows. Section 2 discusses related work. Section 3 introduces the model and the semi/weakly supervised learning setting. Section 4 gives details about the deep architectures employed in the experiments. Section 5 explains the experimental setup and provides results. Section 6 concludes.

2. Related work

Compared to the study of hand pose, a much larger body of work has focused on full-body pose estimation. We draw influence from this literature and therefore include it in our brief review.

Learning — The majority of recent work on pose estimation is based on machine learning. Body part segmentation as an intermediate representation for joint estimation from depth images was successfully used by Shotton et al. [2], where random forests were trained to perform pixel-wise classification. This was adapted to hand pose estimation by Keskin et al. [3], where an additional pose clustering step was introduced. Later, Sun et al. [14] proposed a cascaded regression framework where positions of hand joints are predicted in a hierarchical manner adapted to the kinematic structure of a hand. In work by Tang et al. [12], a random forest performs different tasks at different levels: viewpoint clustering in higher levels, part segmentation in intermediate levels, and joint regression in lower levels. An explicit transfer function is learned between synthetic and real data. In follow-up work by Tang et al. [13], a latent regression forest is learned, which automatically extracts a hierarchical and topological model of a human hand from data. Instead of pixel-wise voting, the forest is descended a single time starting from the center of mass of the hand. The traditional split nodes in the RF are accompanied by division nodes, which trigger parallel descents of sub-trees for multiple entities (joints or groups of joints). Li et al. [15] went further in this direction and proposed a method where the topological model of a hand is learned jointly with hand pose estimation. Later work uses deep networks for body part segmentation from RGB images [16] or depth images [17].

Recent work estimates joint positions by regression with deep convolutional neural nets [1, 5, 4, 18, 19, 10**? ?**]. Typically such models have been trained to produce heatmaps encoding the joint positions as spatial Gaussians, though direct regression to an encoding of joints has also been attempted. Recently, training with a combination of classification and regression losses for producing heatmaps has been proven particularly effective [20].

Post-processing to enforce structural constraints is based on graphical models [19, 4, 5, 7] or inverse kinematics [1]. In a work by Oberweger et al. [21] a deep learning framework is regularized by a bottleneck layer, forcing the network to model underlying structure of joint positions. Ge at al. [22] proposed a multi-view deep learning framework based on feature extraction from several projections of a point cloud representing a depth image of a hand. Stacked hourglass networks [23] introduce a series of modules, each of which perform convolutions followed by deconvolutions with skip connections. The different modules allow the method to model context, in the spirit of auto-context models for semantic segmentation [24].

In a recent overview [25], deep learning models were shown to perform the best among existing approaches, but still far from

human performance. In this context of data-driven methods, there were a number of recent works dedicated to automating data labeling [26] and weakly-supervised learning for sparsely annotated videos [27].

Graphical models — In recent work by Chen and Yuille [5], a graphical model is implemented with deep convolutional nets, which jointly estimate unary terms, given evidence of joint types and positions, and binary terms, modelling relationships between joints. This work was later extended in [7] for modelling occluding parts by introducing an additional connectivity prior. Tompson et al. [4] jointly learn a deep full-body part detector with a Markov Random Field which models spatial priors. Joint learning is achieved by designing the priors as convolutions and implementing inference as forward propagation approximating a single step in a message passing algorithm. Alternatively, Chu et al. [8] proposed a method for learning structured representations by capturing dependencies between body joints during training with introduced geometrical transform kernels.

Top down methods — A different group of methods is based on top-down processes which fit 3D models to image data. A method by Oikonomidis et al. [28] is based on pixelwise comparison of rendered and observed depth maps. Inverse rendering of a generative model including shading and texture is used for model fitting in [29]. Several works [30, 31] employ ICP for hand pose reconstruction and 3D fingertip localization under spatial and temporal constraints. More recent work by Tang et al. [9] is based on hierarchical sampling optimization, where a sequence of predictors is aligned with kinematic structure of a hand.

A hybrid method by Qian et al. [32] for real-time hand tracking uses a simple hand model consisting of a number of spheres and combines a gradient-based discriminative step with stochastic optimization. More recent work by Sharp et al. [11] directly exploits a hand mesh consisting of triangles and vertices to formulate a multi-level discriminative strategy followed by generative model-based refinement. In [33], a person-specific and hybrid generative/discriminative model is built for a system using five RGB cameras plus a ToF camera. In the spirit of hybrid methods, Oberweger et al. [10] proposed a neural model including a feedback loop based on iterative image generation from obtained predictions followed by self-correction.

Top down down methods can be very accurate, if the geometric models are flexible enough and can closely fit the real geometric data. In recent work, geometric models are adapted to each instance (body or hand). In [34], for instance, the two fitting steps (pose estimation and shape estimation) are integrated into a single joint optimization procedure. The downside of these models is their requirement for initialization. Hybrid models combine generative and discriminative steps, for instance by performing discriminative initialization followed by fitting of a generative model.

[0] https:/www.youtube.com/watch?v=7GMiExWKM8c

Our work can be seen as an improvement of the discriminative stage which could be augmented with a generative counterpart of choice.

Correspondence — Pose estimation has been attempted by solving correspondence problems in other work. In a method by Taylor et al. [35], correspondence between depth pixels and vertices of an articulated 3D model is learned. In [36], approximate directed Chamfer distances are used to align observed edge images with a synthetic dataset.

Recently, model fitting and training a network on dense correspondances between the input and the template has proven to be extremely efficient in the context of facial landmark localization [37]. Along these lines, creation of large-scale synthetic datasets for human pose estimation [16] will facilitate adaptation of dense prediction methods to the human pose. Our work is essentially motivated by the same idea of exploiting dense signals as a source of universal and rich supervision.

Other work — Work on applications other than pose estimation share similarities with our method. Patchwise alignment of segmentations in a transductive learning setting has been performed on 3D meshes [38], matching real unlabelled shape segments to shapes from a large labelled database. A large number of candidate segments is created and the optimal segmentation is calculated solving an integer problem. Our method can be viewed as a kind of multi-task learning, a topic actively pursued by the deep vision community [39, 40]. While these techniques rely on subnetworks that share parameters, our approach does not use weight sharing, instead, it co-ordinates subnetworks via a patchwise restoration process and a joint error function.

Our work bears a certain resemblance to the recently proposed N^4-Fields [41]. This method, which was proposed for different applications, also combines deep networks and a patchwise nearest neighbor (NN) search. However, whereas in [41], NN-search is performed in a feature space learned by deep networks, our method performs NN-search in a patch space corresponding to semantic segmentation learned by deep networks. This part of our work also bears some similarity to the way label information is integrated in structured prediction forests [42], although no patch alignment with a dictionary is carried out there.

3. Semi-supervised and weakly-supervised learning of joint regression

When prediction models are learned automatically, representation learning and full end-to-end training are often desirable and described as the holy grail in machine learning. This approach indeed benefits from properties, such as freedom of the training procedure to explore and find the best representation and a lower burden on the scientist or practitioner, who is not required to handcraft representations from knowledge of the application and/or the input data domain.

In practice, however, depending on the difficulty of the problem, the depth of the model and the amount of available training data, it might be suboptimal to give the model complete freedom over the intermediate representations it explores. Prob-

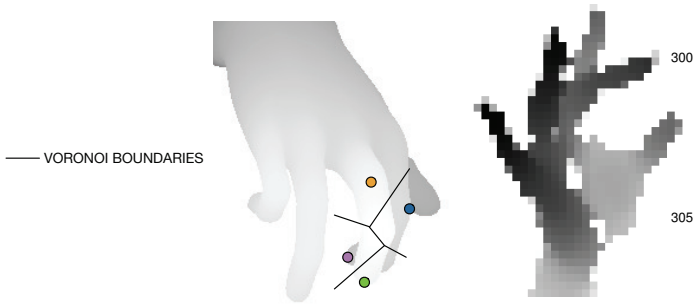


Figure 2: Complementarity of segmentation maps and key points: the space occupied by the index finger and the ring finger is difficult to segment, as seen by the Voronoi borders (on the left); depth values are often very similar and close for different fingers at realistic sensor resolutions (taken from the NYU Hand Pose dataset, on the right).

lems can either arise from overfitting or from suboptimal solutions found by the minimization procedure. Instead of falling back to handcrafting feature representations, solutions can be intermediate supervision, or a decomposition approach, where an intermediate representation is temporarily imposed during a pre-training step before full end-to-end training.

In this work we propose an intermediate representation in the form of a segmentation map. In contrast to traditional decomposition approaches, the intermediate representation is available as additional information to the final regressor, which also receives raw input. Our intermediate representation is a segmentation into 20 parts, illustrated in Figure 1. 19 parts correspond to finger parts, one part to the palm (see Figure 6 for exact definitions). The background is considered to be subtracted in a preprocessing step and is not part of the segmentation process.

This dense segmentation is complementary to groundtruth key points and the former is very difficult to obtain from the latter in the case of strong auto occlusions. This is illustrated in Figure 2a: the space occupied by the index finger and the ring finger is difficult to segment. Resorting to spatial distances alone, traditionally done in easier settings, fails due to complex curved shapes of fingers and the low amount of keypoints which can be obtained with automatic methods (2 points per finger in the NYU dataset [1]). In this context, a Voronoi diagram gives regions which are very different from the complex finger regions. Depth values might theoretically help, but in practice this fails since the values are often very similar and close for different fingers at realistic sensor resolutions, as shown in Figure 2b taken from the NYU dataset.

Compared to the initial input depth image, this representation has several important advantages:

- The part label space is characterized by strong topological properties. In contrast to other semantic segmentation problems (for instance, semantic full scene labeling), strong neighborhood relationships can be defined on the label space. They can be leveraged to restore noisy part label images and to generate a loss function for training. This property serves as a key component of creating reliable estimators and for performing transfer from synthetic to real images.
- For a given view and pose, the part label itself carries strong geometrical information: the label alone of a given pixel is a very strong prior on the position of the pixel in 3d. This provides important cues for regression to the desired joint positions. This property will also be exploited in our system to motivate patchwise searches in label space.

In our setting, the intermediate representation is available during training time for the synthetic data only. For real data and during test time, it is automatically inferred. More precisely, our training dataset is organized into two partitions: a set of real depth images with associated ground truth joint positions, and a second set of synthetic depth images with associated ground truth label images (segmentation maps in the intermediate representation). We will denote by $D^{(j)}$ the j^{th} pixel in image D.

The synthetic images have been rendered from different 3D hand models using a rendering pipeline (details will be given in Section 5). As in [33], we also sample different pose parameters and hand shape parameters. Variations in viewpoints and hand poses are obtained taking into account physical and physiological constraints. In contrast to other weakly-supervised or semi-supervised methods, for instance [12], we do not suppose that any ground truth data for the intermediate representation is available for the real training images. Manually labelling segmentations is extremely difficult and time consuming. Labelling a sufficiently large number of images is hardly practical.

We do, however, rely on ground truth for joint positions, which can be obtained in several ways: In [2], external motion capture using markers is employed. In [1], training data is acquired from multiple views from three different depth sensors and an articulated model is fitted offline.

The functional decomposition of the method as well as the dataflow during training and testing are outlined in Figure 3. The goal is to regress joint positions from input depth maps, and to this end, the proposed method leverages two different mappings learned on two training sets. A *segmentation network* learns a mapping $f_s(\cdot, \theta_s)$ from raw depth data to intermediate segmentation maps, parametrized by a parameter vector θ_s. A *regression network* learns a mapping $f_r(\cdot, \theta_r)$ from raw depth data combined with segmentation maps to joint positions, parametrized by a vector θ_r.

The training procedure uses both synthetic and real data, and proceeds in three steps:

1. First, the segmentation network f_s is pre-trained on synthetic training data in a supervised way using dense ground truth segmentations, resulting in a prediction model $f_s(\cdot, \theta_s)$. The parameters are learned minimizing classical negative log-likelihood (NLL). This training step is shown as blue data flow in Figure 3.
2. Then the prediction model for f_s is fine-tuned in a subsequent step by weakly supervised training on real data, resulting in a refined prediction model $f_s(\cdot, \theta_s')$. This step, shown as green data flow in Figure 3, is described in more detail in Subsection 3.1.
3. Finally, the regression network f_r is trained on real data. It is implemented as a mapping $f_r(\cdot, \theta_r)\colon (D, N) \rightarrow \boldsymbol{z}$

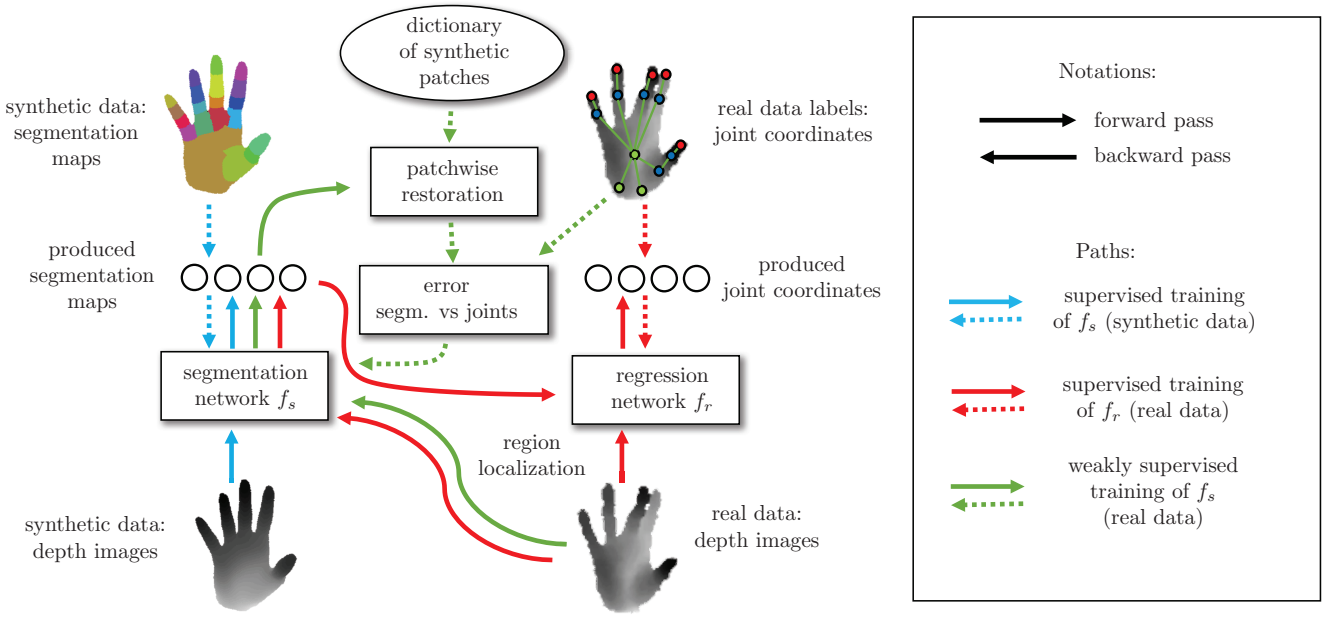


Figure 3: A functional overview of the method. Blue data flow corresponds to supervised training of the segmentation network f_s. Green flow corresponds to weakly supervised training of f_s. Red arrows show supervised training of the regressor f_r (for more detail see a supplementary video available online [2]).

Layer	Filter size / units	Pooling
Segmentation network f_s		
Depth input	48×48	-
Conv. layer 1	32×5×5	2×2
Conv. layer 2	64×5×5	2×2
Conv. layer 3	128×5×5	1×1
Hidden layer	500	-
Hidden layer	500	-
Output	20×48×48	-
Regression network f_r		
Depth input	24×24	-
Conv. layer c1	32×3×3	2×2
Conv. layer c211, c221	16×1×1	
Conv. layer c212, c222	8×3×3	
Pooling p231	-	2×2
Conv. layer 232	8×1×1	
Conv. layer 241	16×1×1	
Hidden layer fc	1200	-
Output	14×3	-

Table 1: Hyper-parameters chosen for the deep networks.

from a full size input depth image D and a full size segmentation map N to a joint location vector $\boldsymbol{z}$. Parameters θ_r are trained classically by minimizing the L_2 norm between output joint positions and ground truth joint positions. This training step is shown as red data flow in Figure 3.

Subsection 3.1 provides more details on step 2, the weakly supervised training procedure. The actual deep architectures employed will be described separately in Section 4.

3.1. *Weakly supervised fine-tuning of the segmentation network*

Supervised pre-training of the segmentation network results in a prediction model $f_s(\cdot, \theta_s)$. To address the domain shift between synthetic data and real data shot with depth sensors, the model is fine-tuned by training on real data. Since no ground truth segmentation maps exist for this data, we generate a loss function for training based on two sources:

- sparse information in the form of ground truth joint positions, and
- *a priori* information on the local distribution of part labels on human hands through a patch-wise restoration process, which aligns noisy predictions with a large dictionary of synthetic poses.

The weakly supervised training procedure is shown as green data flow in Figure 3. Each real depth image is passed through the pre-trained segmentation network f_s, resulting in a segmentation map. This noisy predicted map is restored through a restoration process f_{nn} described further below. The quality of the restored segmentation map is estimated by comparing it to the set of ground truth joint positions for that image. In particular, for each joint, a corresponding part-label is identified, and the barycenter of the corresponding pixels in the segmentation map is calculated. A rough quality measure for a segmentation map can be given as the sum (over joints) of the L_2 distances between barycenters and ground truth joint positions. The quality measure is used to determine whether the restoration process has lead to an improvement in segmentation quality, i.e. whether the barycenters of the restored map are closer to the ground truth joint positions than the barycenters of the original prediction. For images where this is the case, the segmentation network is updated for each pixel, minimizing NLL using the labels of the restored map as artificial "ground truth".

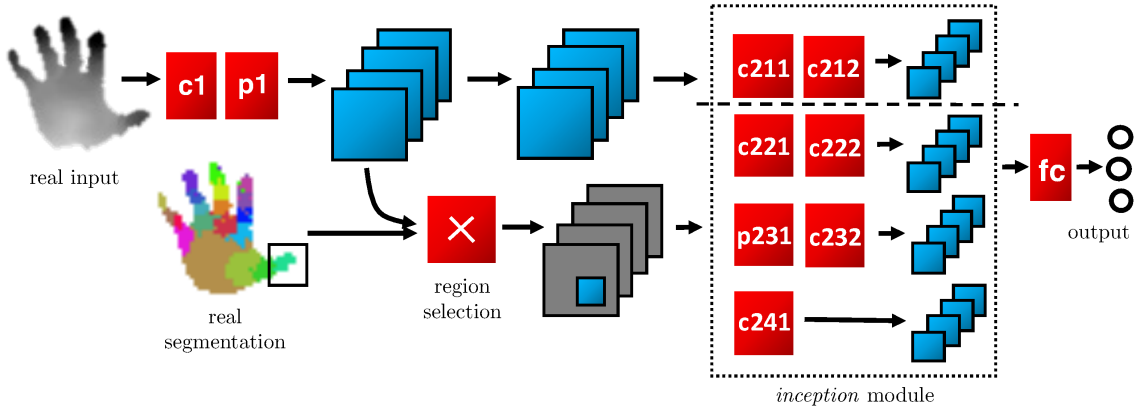


Figure 4: Organization of the regression network f_r. All functional modules are shown in red and the produced feature maps are shown in blue. The grey areas correspond to masked regions on the feature maps. Masked areas are shown as rectangles in the figure, but are of general shape.

Figure 5: Different segmentation results: (a) input image; (b) output of the segmentation network after supervised training; (c) after restoration; (d) output of the segmentation network after joint training; (e) ground truth segmentation; (f) estimated joint positions. The image itself was *not* part of the training set.

3.2. Patchwise restoration

We proceed patchwise, extracting patches of size $P\times P$ from a large set of synthetic segmentation images, resulting in a dictionary of patches $\boldsymbol{P}=\{\boldsymbol{p}^{(l)}\}$, $l\in\{1\dots N\}$. In our experiments, we used patches of size 27×27 and a dictionary of 36 million patches is extracted from the training set (see Section 5). As also reported in [33], the range of poses which can occur in natural motion is extremely large, making full global matching with pose datasets difficult. This motivates our patch-based approach, which aims to match at a patch-local level rather than matching whole images.

A given real input depth image D is aligned with this dictionary in a patchwise process using the intermediate representation. For each pixel j, a patch $\boldsymbol{q}^{(j)}$ is extracted from the segmentation produced by the network f_s, and the nearest neighbor $\nu(\boldsymbol{q}^{(j)})$ is found by searching the dictionary $\boldsymbol{P}$:

$$\boldsymbol{n}^{(j)} \triangleq \nu(\boldsymbol{q}^{(j)}) = \arg\min_{\boldsymbol{p}^{(l)}\in\boldsymbol{P}} d_H(\boldsymbol{q}^{(j)}, \boldsymbol{p}^{(l)}), \qquad (1)$$

where $d_H(\boldsymbol{q}, \boldsymbol{p})$ is the Hamming distance between the two patches $\boldsymbol{q}$ and $\boldsymbol{p}$. The search performed in (1) can be calculated efficiently using KD-trees.

In a naïve setting, a restored label for pixel j could be obtained by choosing the label of the center of the retrieved patch $\boldsymbol{n}^{(j)}$. This, however, leads to noisy restorations, which suggest the need for spatial context. Instead of chosing the center label of each patch only, we propose to use all of the labels in each patch. For each input pixel, the nearest neighbor results in a local window of size $W \times W$ are integrated. In particular, for a given pixel j, the assigned patches $\boldsymbol{n}^{(k)}$ of all neighbors k are examined and the position of pixel j in each patch is calculated. A label is estimated by voting, resulting in a mapping $f_{nn}(.)$:

$$f_{nn}(\boldsymbol{q}^{(j)}) = \arg\max_{l} \sum_{k \in W \times W} \mathbb{I}(l = \boldsymbol{n}^{(k, j@k)}) \qquad (2)$$

where $\boldsymbol{n}^{(k)}{=}\nu(\boldsymbol{q}^{(j)})$ is the nearest neighbor result for pixel k, $\boldsymbol{n}^{(j,m)}$ denotes pixel m of patch $\boldsymbol{n}^{(j)}$, $\mathbb{I}(\omega){=}1$ if ω holds and 0 else, and the expression $j@k$ denotes the position of pixel j in the patch centered on neighbor k.

This integration bears some similarity to [41], where information of nearest neighbor searches is integrated over a local window, albeit through averaging a continuous mapping. It is also similar to the patchwise integration performed in structured prediction forests [42].

If real-time performance is not required, the patch alignment process in Equation 1 can be regularized with a graphical model including pairwise terms favoring consistent assignments, for instance, a Potts model or a term favoring patch assignments with consistent overlaps. Interestingly, this produces only very small gains in performance, especially given the higher computational complexity. Moreover, the gains vanish if local integration (Equation 2) is added. More information is given in Section 5.

4. Architectures

The structure of the segmentation network f_s is motivated by the idea of performing efficient pixelwise image segmentation preserving the original resolution of the input, and is inspired by *OverFeat* networks which were proposed for object detection and localization [43, 44]. The network consists of 3 convolutional layers, followed by a fully connected layer (see Table 1). Max pooling, which typically follows convolutional layers, results in downsampling of feature maps, destroying precise spatial information. Instead, we perform pooling over $2{\times}2$ overlapping regions produced by shifting the feature maps obtained at the previous step by a single pixel along one or another axis. As opposed to patchwise training of pixel classification based on its local neighborhood, such an architecture is more computationally efficient, as it benefits from dense computations at earlier layers. Compared to recently introduced fully-convolutional [45] and deconvolutional networks [46], which tackle a similar problem in the context of semantic segmentation, the proposed network requires less upsampling and interpolation.

The regression network, taking a depth image as a single input and producing 3 coordinates for a given joint, also incorporates the information provided by the segmentation network during training. Structurally, it resembles an Inception module [47, 48] where the output of the first convolutional layer after max pooling as passed through several parallel feature extractors capturing information at different levels of localization. The organization of this module is shown in Figure 4. and the corresponding parameters are, as before, provided in Table 1. The output of the first convolutional layer c1 (followed by pooling p1) is aligned with the segmentation maps produced by the segmentation network. From each feature map, for a given joint we extract a localized region of interest filtered by the mask of a hand part to which it belongs (or a number of hand parts which are naturally closest to this joint). These masks are calculated by performing morphological opening on the regions having the corresponding class label in the segmentation map. Once the local region is selected, the rest of the feature map area is set to 0. The result, along with the original feature maps is then fed to the next layer, i.e. an Inception module. The rest of the training process is organized in such a way that the network's capacity is split between global structure of the whole image and the local neighborhood, and a subset of Inception $3{\times}3$ filters is learned specifically from the local area surrounding the point of interest.

Experiments showed that individual networks for each joint do perform better than networks sharing parameters over joints. We conjecture, that sharing parameters would be the optimal choice for smaller amounts of training data. As the number of examples increases, separating networks allows all layers to pick up the fine nuances required for regressing each individual joint.

Both networks have ReLU activation functions at each layer and employ batch normalization [49]. The regression network during test time uses the batch normalization parameters estimated on the training data, however, in the segmentation network, the batch normalization is performed across all pixels from the same image, for both training and test samples.

5. Experimental results

We evaluated the proposed method on the *NYU Hand Pose Dataset*, which was published in [1] and is publicly available[3]. It comprises 70,000 images captured with a depth sensor in VGA resolution accompanied by ground truth annotations of positions of hand joints.

A video illustrating the results of our method is available online[4]. It shows the obtained pose as well as the intermediate representation during testing and during training, i.e. after restoration.

To train the segmentation network, the synthetic training images were selected from our own dataset consisting of two subsets: (i) 170,974 synthetic training images rendered using the

[3] http://cims.nyu.edu/~tompson/NYU_Hand_Pose_Dataset.htm

[4] https://www.youtube.com/watch?v=7GMiExWKM8c

Method	— per pixel—		— per class —	
No restoration	51.03		39.38	
NN-search — no integration	48.76		39.72	
NN-search — integration w. equation (2)	**54.55**	**(+3.52)**	**46.38**	**(+7.00)**
CRF – Potts-like model	53.10	(+2.07)	43.64	(+4.26)
CRF – Hamming distance on overlapping patch area	52.45	(+1.42)	42.68	(+3.30)

Table 2: Restoration (=segmentation) accuracy on 100 manually labelled images of the NYU dataset (=NYU-100).

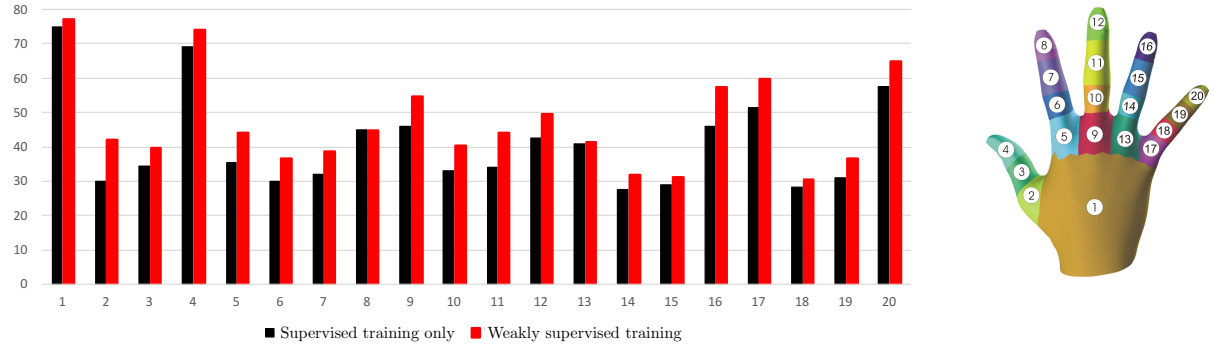


Figure 6: Classification accuracy of the segmentation network f_s for supervised training only (blue), and for weakly supervised training (red), where $1 \ldots 20$ is the number of a segment shown in the hand legend at the right.

commercial software "Poser", including ground truth; (ii) 500 labelled synthetic images plus ground truth reserved for testing.

In our experiments, we extract hand images of normalized metric size (taking into account depth information) and resize them to 48×48 pixels. The data is preprocessed by local contrast normalization with a kernel size of 9. To be able to predict absolute values for z-coordinates, the subtracted depth is then added back to the network output. For supervised training of the segmentation network and the regression network, the full set of 170,974 training images is used. A third of this set is used to extract patches for the patch alignment mapping f_{nn}, giving a dictionary of 36M patches of size 27×27 extracted from 56,991 images. Local integration as given in equation (2) was done using windows of size $W\times W = 17\times17$.

The segmentation network was initially trained for 100 epochs with SGD, batch size 1, initial learning rate 0.1 and learning rate decay 10^{-5} on the synthetic data and then fine-tuned for additional 10 epochs on a mixture of synthetic and restored real segmentations with the ratio 9:1.

Finally, the regression network is trained by gradient descent using the Adam [50] update rule, with learning rate set 0.05 and batch size of 64.

ConvNets were implemented using the Torch7 library [51]. NN-search using KD-trees was performed using the FLANN library [52].

5.1. Segmentation performance

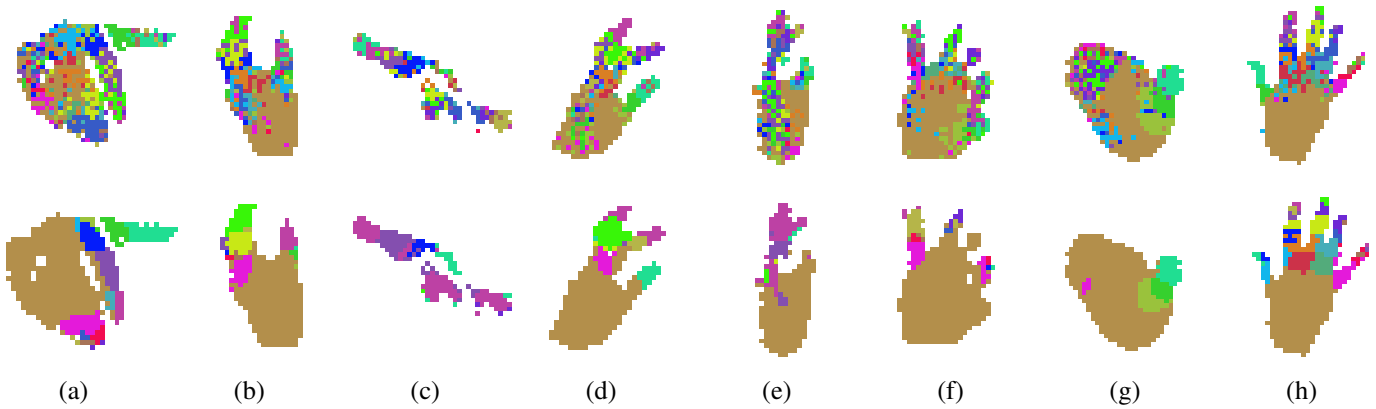
To evaluate the performance of the various segmentation methods, we manually labelled 100 images from the NYU dataset and report accuracy per pixel and per class in Table 2. These 100 images were solely used for evaluation and never entered any training procedure. Recall that pure segmentation performance is of course not the goal of this work (or of the targeted application), it is given as additional information only.

Purely supervised training on the synthetic dataset gives poor segmentation performance of 51.03% accuracy per pixel. We emphasize once more that training was performed on synthetic images while we test on real images, thus of an unseen distribution. This domain shift is clearly a problem, as accuracy on the synthetic dataset is very high, 90.16%. Using patchwise restoration of the predicted real patches with a large dictionary of synthetic patches gives a performance increase of $+3.5$ percentage points per pixel and $+7$ percentage points per class. This corroborates our intuition that the intermediate representation carries important structural information. Integration of patch-labels over a local window with equation (2) is essential– pure NN-search without integration performs poorly.

Figure 5 provides examples of input depth maps and different segmentations: after synthetic pre-training only, after off-line restoration and a prediction after weakly-supervised finetuning. We can see that in a majority of cases the restoration output is very close to ground truth maps. Figure 6 shows the distribution of the error over the different hand parts. The improvement is consistent, and high on the important fingertips.

Restoration failure cases, corresponding to 5-10% of the real data, are demonstrated and explained in Figure 7. At the finetuning stage, these samples are filtered out and excluded from training (see Section 3.1 for more detail).

We also compared local integration of equation (2) to potentially more powerful regularization methods by implementing a CRF-like discrete energy function. The goal of the optimization problem is to regularize the patchwise restoration process described in Section 3.2. Instead of chosing the nearest neigh-



(a) (b) (c) (d) (e) (f) (g) (h)

Figure 7: Visualization of the automatic rejections made by the quality check described in section 3.1. Examples of raw noisy segmentations of real images are shown on top and the corresponding maps after restoration are shown at bottom. These restoration failure cases may happen due to artifacts in depth maps (c), (e), (f), increased distance between the camera and the hand (g) and particularly noisy initial predictions (a), (d), (h). As a result, the restoration process may result in lower recall of finger segments (a)-(g) or, less often, false detections (h). Both cases can be automatically detected.

Method	Datasets used	— per pixel—		— per class —	
Fully supervised training only	synth. segmentations	51.03		39.38	
Semi-/weakly-supervised training	synth. segmentations + real joint positions	**57.18**	(+6.15)	**47.20**	(+7.82)

Table 3: The contribution of semi-/weakly-supervised training on segmentation accuracy.

bor in patch space for each pixel as described in equation (2), a solution is searched which satisfies certain coherence conditions over spatial neighborhoods. To this end, we create a global energy function $E(x)$ defined on a 2D-lattice corresponding to the input image to restore:

$$E(x) = \sum_i u(x_i) + \alpha \sum_{i \sim j} b(x_i, x_j)$$

where $i \sim j$ indices neighbors i and j. Each pixel i is assigned a discrete variable x_i taking values between 1 and $N{=}10$, where $x_i{=}l$ signifies that for pixel i the $l{-}th$ nearest neighbor in patch space is chosen. For each pixel i, a nearest neighbor search is performed using KD-trees and a ranked list of N neighbors is kept defining the label space for this pixel. The variable N controls the degree of approximation of the model, where $N{=}\infty$ allows each pixel to be assigned every possible patch of the synthetic dictionary.

The unary data term $u(x)$ guides the solution towards approximations with low error. It is defined as the Hamming distance between the original patch and the synthethic patch.

We tested two different pairwise terms $b(x_i, x_j)$:

Potts-like terms — a Potts model classically favors equality of labels of neighboring sites. In our setting, we favor equality of the center pixels of the two patches assigned to x_i and x_j.

Patch-overlap distance — the alternative pairwise term is defined as the Hamming distance between the two synthethic patches defined by x_i and x_j, in particular, the distance restricted to the overlapping area.

Inference was performed through message passing using the open-GM library [63], and the hyper-parameter α was optimized on a hold-out set. Interestingly, local patch integration with equation (2) outperformed the combinatorial models significantly while at the same time being much faster. We conjecture that the reason is that our patchwise integration corresponds to a high-order potential (2), whereas the binary terms in the CRF model are poorer. Calculating the global solution of poorer model seems to be less efficient than locally solving a high order problem. We see this as a further indication of the strong topological information carried by the label space of the intermediate representation.

Table 3 shows the contribution of semi/weakly-supervised training, where sparse annotation (joint positions) are integrated into the training process of the segmentation network f_s. This procedure achieves an improvement of $+6.15$ percentage points per pixel and $+7.82$ percentage points per class, an essential step in learning an efficient intermediate representation.

At first sight it might seem to be odd that the pretrained predictor (+6.15pp w.r.t.t. baseline) is better than the off-line restoration process (+3.52pp), on whose results it has been trained. However, let us recall that the predictor also uses a second source of information, namely the joint locations available on real images during the weakly-supervised fine-tuning step. Rejecting bad segmentations during training is responsible for the difference.

5.2. Hand joint position estimation

Table 4 illustrates the effect of incorporating segmentation information on the performance of the regression network, i.e. on the error in joint positions — the main goal of this work. In the

Method	2D error, mm	3D error, mm
Tompson et al. [1]	7.1	28.8
Oberweger et al. (DeepPrior) [21]	14.8	19.8
Oberweger et al. [10]	12.4	16.0
Bouchacourt et al. (DISCO) [53]	–	20.7
Xu et al. (Lie-X) [54]	–	14.5
Zhou et al. (DeepModel) [55]	–	16.9
Deng et al. (Hand3D) [56]	–	17.6
Guo et al. (REN) [57]	–	13.4
Wan et al. (Crossing Nets) [58]	–	15.5
Fourure et al. (JTSC) [59]	8.0	16.8
Zhang et al. [60]	–	18.3
Madadi et al. [61]	–	15.6
Oberweger et al. (DeepPrior++) [62]	–	12.3
Our baseline: direct regression (a)	13.3	17.3
Our baseline: cascade direct regression (two steps) (b)	12.6	16.9
Our baseline: semi-supervised network (c)	12.1	16.5
Our method: semi/weakly supervised network (d)	11.2	14.8

Table 4: Joint position estimation error on the NYU Hand Pose dataset (some values were estimated from plots if authors did not provide numerical values). Baseline (a) corresponds to the case of a single regression network where the segmentation results are not used. The cascaded baseline (b) is similar in architecture but features an additional refinement step (as proposed in [21]). Method (c) is based on the full pipeline with no restoration employed during training, i.e. the segmentation network is trained on synthetic data in supervised way and not fine-tuned on the real samples. Finally, method (d) corresponds to our proposed solution.

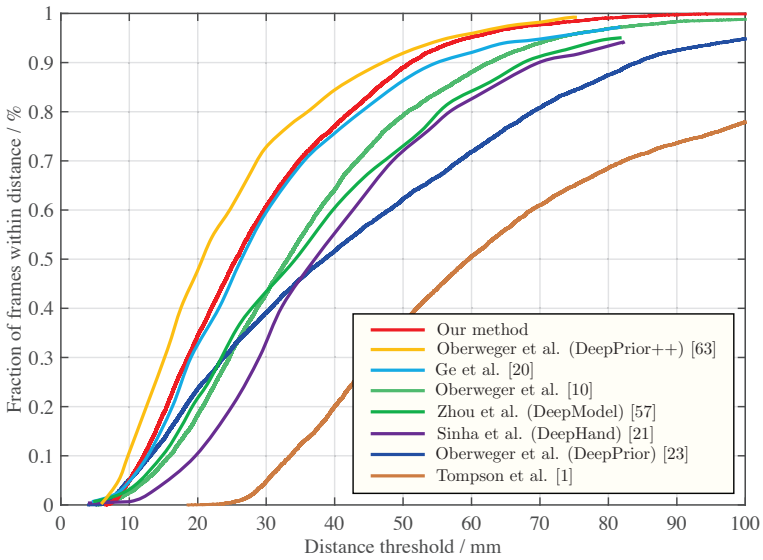


Figure 8: Joint estimation accuracy for the 3D case expressed as propotion of frames where all joints are localized within a given distance threshold in mm.

bottom part of the table we compare the proposed method to our own baselines, which are based on an ablation study of the proposed network. In particular, we compare with direct regression without the intermediate representation, and with a version where the intermediate representation is learned in a supervised way only.

The first part of the table provides comparison with the literature and is based on publicly available data (predictions of uv-coordinates of joint locations) released by the authors of corresponding methods. The error is expressed as mean distance in mm (in 2D or 3D) between predicted position of each joint and its ground truth location. In comparison to a single network regressor, the 2D mean error was improved by 15.7%. The second best model not involving segmentation (cascade regression) was inspired by the work of [18] and more recent [21], where the initial rough estimation of hand joints positions is then improved by zooming in.

We can see that the purely supervised baseline performs well compared to the state-of-the-art. Prior to augmenting the method with the unsupervised pipeline, we spent additional time on careful optimization of the baselines by tuning the architecture and the training regime. This appeared to be crucial,

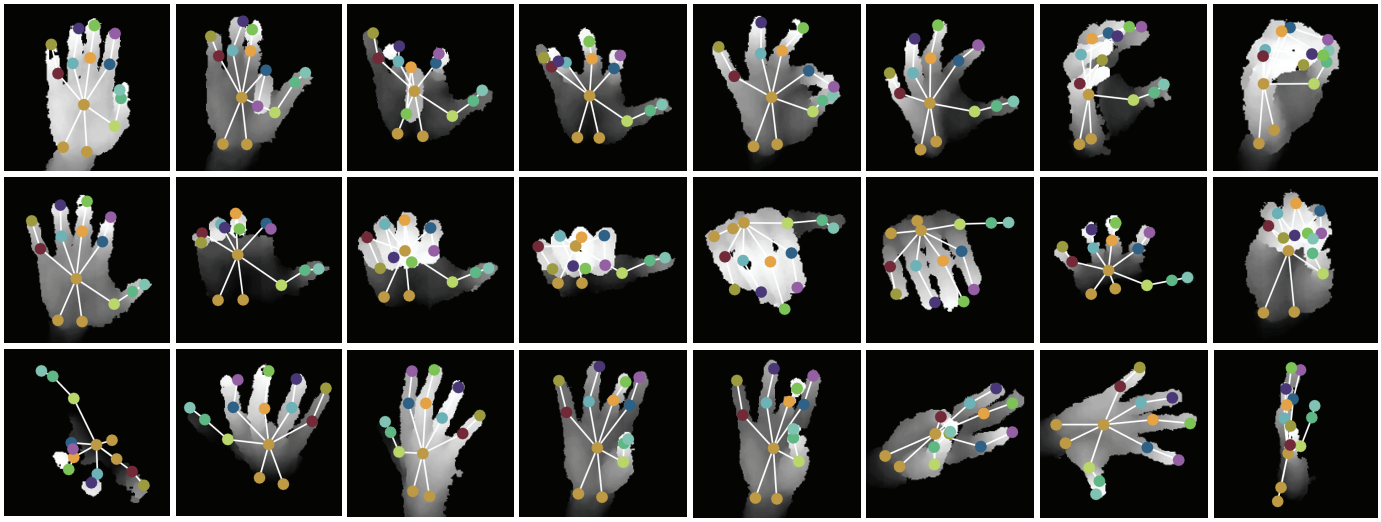


Figure 9: Visualization of estimated hand skeletons produced by our network trained in a semi/weakly-supervised fashion (depth images are sampled uniformly from the NYU Hand Pose dataset).

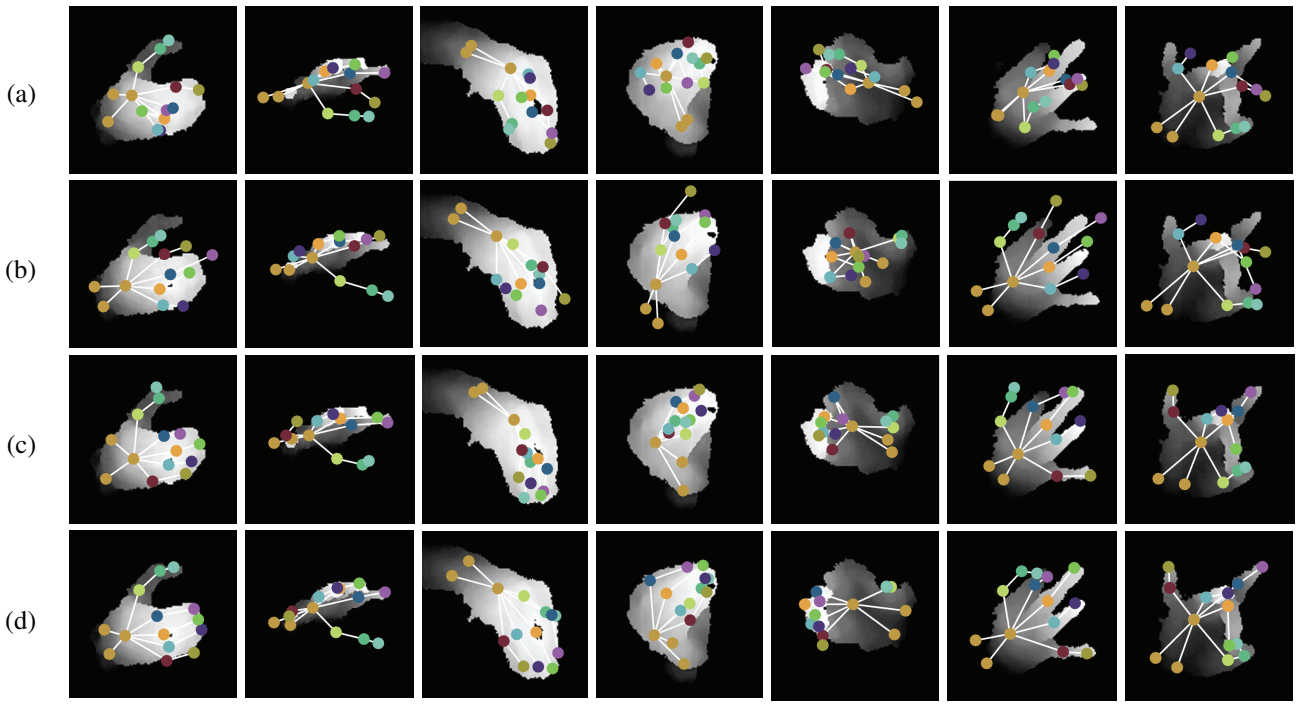


Figure 10: Visual comparison of results produced on the NYU Hand Pose dataset by different 3D pose estimation methods: (a) DeepPrior [21], (b) Oberweger et al. [10], (c) our method, based on the network trained in a semi/weakly-supervised fashion, (d) ground truth.

in particular the choice of batch normalization [64] and Adam optimization [65].

Figure 9 visualizes the estimated pose skeletons corresponding to 24 input depth maps randomly sampled from the test set. Figure 10 illustrates performance of the proposed method on challenging examples in comparison with a number of state-of-the-art methods. Figure 8 plots quantitative performance expressed as the number of frames with all joints being localized within a given distance threshold in 3D. In these figures, we compare the proposed method with several recent state-of-the-art approaches. For the 2D hand pose estimation method proposed in [1], we follow [21] and augment estimated x and y coordinates with depth values from the input depth maps. In those cases when predicted locations fall on the background, we set the corresponding z-coordinate to the median depth value of the hand.

We should note here, that the quality of network outputs can be further improved by optimization through inverse kinematics, as it has been done, for example, in [1]. However, the focus of this work is to explore the potential of pure learning approaches with no priors enforcing structure on the output. The bottom part of the table contains non deep learning methods. In a recent work [9] on optimization of hand pose estimation formulated as an inverse kinematics problem, the authors report performance similar to [1] in terms of 2D UV-error (no error in mm provided).

5.3. Computational complexity

All models have been trained and tested using GPUs, except the patchwise restoration process which is pure CPU code and not used at test time. Estimating the pose of a single hand takes 31 ms if the segmentation resolution is set to 24×24 pixels (which includes the forward passes of both networks f_s (12 ms) and f_r (18 ms)) and 58 ms for 48×48 segmentation outputs (40 ms for f_s, corresponding to the results reported in the experiments section of the paper). Training of the segmentation network requires up to 24 hours to minimize validation error, while the regression network is trained in 20 min on a single GPU. The results were obtained using a cluster configured with 2 x E5-2620 v2 Hex-core processors, 64 GB RAM and 3 x Nvidia GTX Titan Black cards with 6GB memory per card.

6. Conclusion

We presented a method for hand pose estimation based on an intermediate representation which is fused with raw depth input data. We showed that the additional structured information of this representation provides important cues for joint regression which leads to lower error. Weakly supervised learning of the mapping from depth to segmentation maps from a mixture of densely labelled synthetic data and from sparsely labelled real data is a key component of the proposed method. Weak supervision is dealt with by patch-wise alignment of real data to synthetic data performed in the space of the intermediate representation, exploiting its strong geometric and topological properties.

7. Acknowledgements

This work has been partly financed through the French grant Interabot, a project of type "Investissement's d'Avenir / Briques Génériques du Logiciel Embarqué", and by the ANR project SoLStiCe (ANR-13-BS02-0002-01), a project of the grant program "ANR blanc". G. Taylor acknowledges the support of NSERC, CFI, and NVIDIA.

References

[1] J. Tompson, M. Stein, Y. LeCun, K. Perlin, Real-time continuous pose recovery of human hands using convolutional networks, in: ACM Transactions on Graphics, 2014.

[2] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, A. Blake, Real-time human pose recognition in parts from single depth images, in: CVPR, 2011.

[3] C. Keskin, F. Kiraç, Y. Kara, L. Akarun, Hand pose estimation and hand shape classification using multi-layered randomized decision forests, in: ECCV, 2014.

[4] J. Tompson, A. Jain, Y. LeCun, C. Bregler, Joint training of a convolutional network and a graphical model for human pose estimation, in: NIPS, 2014.

[5] X. Chen, A. Yuille, Articulated pose estimation by a graphical model with image dependent pairwise relations, in: CVPR, 2014.

[6] M. Sun, P. Kohli, J. Shotton, Conditional regression forests for human pose estimation, in: CVPR, 2012.

[7] X. Chen, A. Yuille, Parsing occluded people by flexible compositions, in: CVPR, 2015.

[8] X. Chu, W. Ouyang, H. Li, X. Wang, Structured feature learning for pose estimation, in: arXiv:1603.09065v1, 2016.

[9] D. Tang, J. Taylor, K. Pushmeet, C. Keskin, T.-K. Kim, J. Shotton, Opening the black box: Hierarchical sampling optimization for estimating human hand pose, in: ICCV, 2015.

[10] M. Oberweger, P. Wohlhart, V. Lepetit, Training a feedback loop for hand pose estimation, in: ICCV, 2015.

[11] T. Sharp, C. Keskin, D. Robertson, J. Taylor, J. Shotton, D. Kim, C. Rhemann, I. Leichter, A. Vinnikov, Y. Wei, D. Freedman, P. Kohli, E. Krupka, A. Fitzgibbon, S. Izadi, Accurate, robust, and flexible real-time hand tracking, in: SIGCHI, 2015.

[12] D. Tang, T. Yu, T.-K. Kim, Real-time articulated hand pose estimation using semi-supervised transductive regression forests, in: ICCV, 2013.

[13] D. Tang, H. Chang, A. Tejani, T.-K. Kim, Latent regression forest: Structured estimation of 3d hand posture, in: CVPR, 2014.

[14] X. Sun, Y. Wei, S. Liang, X. Tang, J. Sun, Cascaded hand pose regression, in: CVPR, 2015.

[15] P. Li, H. Ling, X. Li, C. Liao, 3d hand pose estimation using randomized decision forest with segmentation index points, in: ICCV, 2015.

[16] G. Varol, J. Romero, X. Martin, N. Mahmood, M. Black, I. Laptev, C. Schmid, Learning from synthetic humans, 2017, cVPR.

[17] M. Jiu, C. Wolf, G. Taylor, A. Baskurt, Human body part estimation from depth images via spatially-constrained deep learning, Pattern Recognition Letters 50 (2014) 122–129.

[18] A. Toshev, C. Szegedy, DeepPose: Human pose estimation via deep neural networks, in: CVPR, 2014.

[19] A. Jain, J. Tompson, M. Andriluka, G. Taylor, C. Bregler, Learning human pose estimation features with convolutional networks, in: ICLR, 2014.

[20] A. Bulat, G. Tzimiropoulos, Human pose estimation via convolutional part heatmap regression, in: ECCV, 2016.

[21] M. Oberweger, P. Wohlhart, V. Lepetit, Hands deep in deep learning for hand pose estimation, in: CVWW, 2015.

[22] L. Ge, H. Liang, J. Yuan, D. Thalmann, Robust 3d hand pose estimation in single depth images: From single-view cnn to multi-view cnns, in: CVPR, 2016.

[23] A. Newell, K. Yang, J. Deng, Stacked hourglass networks for human pose estimation, in: ECCV, 2016.

[24] P. Pinheiro, R. Collobert, Recurrent convolutional neural networks for scene labeling, in: ICML, 2014.

[25] J. S. III, G. Rogez, Y. Yang, J. Shotton, D. Ramanan, Depth-based hand pose estimation: methods, data, and challenges, in: ICCV, 2015.

[26] M. Oberweger, G. Riegler, P. Wohlhart, V. Lepetit, Efficiently creating 3d training data for fine hand pose estimation, in: CVPR, 2016.

[27] O. Koller, H. Ney, R. Bowden, Deep hand: How to train a cnn on 1 million hand images when your data is continuous and weakly labelled, in: CVPR, 2016.

[28] I. Oikonomidis, N. Kyriazis, A. Argyros, Efficient model-based 3D tracking of hand articulations using Kinect, in: BMVC, 2011.

[29] M. de La Gorce, D. Fleet, N. Paragios, Model-based 3d hand pose estimation from monocular video, IEEE TPAMI 33 (9) (2014) 1793–1805.

[30] H. Liang, J. Yuan, D. Thalmann, Z. Zhang, Model-based hand pose estimation via spatial-temporal hand parsing and 3D fingertip localization, The Visual Computer 29 (2013) 837–848.

[31] A. Tagliasacchi, M. Schroder, A. Tkach, S. Bouaziz, M. Botsch, M. Pauly, Robust articulated-icp for real-time hand tracking, in: Eurographics Symposium on Geometry Processing, 2015.

[32] C. Qian, X. Sun, Y. Wei, X. Tang, J. Sun, Realtime and Robust Hand Tracking from Depth, in: CVPR, 2014.

[33] S. Sridhar, A. Oulasvirta, C. Theobalt, Interactive markerless articulated hand motion tracking using RGB and depth data, in: ICCV, 2013.

[34] J. Taylor, L. Bordeaux, T. Cashman, B. Corish, C. Keskin, E. Soto, D. Sweeney, J. Valentin, B. Luff, A. Topalian, E. Wood, S. Khamis, P. Kohli, T. Sharp, S. Izadi, R. Banks, A. Fitzgibbon, J. Shotton, Efficient and precise interactive hand tracking through joint, continuous optimization of pose and correspondences, in: ACM TOG - SIGGRAPH, 2016.

[35] J. Taylor, J. Shotton, T. Sharp, A. Fitzgibbon, The vitruvian manifold: Inferring dense correspondences for one-shot human pose estimation, in: CVPR, 2012.

[36] V. Athitsos, Z. Liu, Y. Wu, J. Yuan, Estimating 3D hand pose from a cluttered image, in: CVPR, 2003.

[37] R. A. Guler, G. Trigeorgis, E. Antonakos, P. Snape, S. Zafeiriou, I. Kokkinos, Densereg: Fully convolutional dense shape regression in-the-wild, 2017, cVPR.

[38] W. Xu, Z. Shi, M. Xu, K. Zhou, J. Wang, B. Zhou, J. Wang, Z. Yuan, Transductive 3D shape segmentation using sparse reconstruction, in: Transactions on Graphics, 2014.

[39] T. Devries, K. Biswaranjan, G. Taylor, Multi-task learning of facial landmarks and expression, in: 14th Canadian Conference on Computer and Robot Vision (CRV), 2014.

[40] Z. Zhang, P. Luo, C. C. Loy, X. Tang, Facial landmark detection by deep multi-task learning, in: ECCV, 2014.

[41] Y. Ganin, V. Lempitsky, N^4-Fields: Neural Network Nearest Neighbor Fields for Image Transforms, in: ACCV, 2014.

[42] P. Kontschieder, S. Bulo, M. Pelillo, H. Bischof, Structured labels in random forests for semantic labelling and object detection, in: ICCV, 2011.

[43] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, Y. LeCun, Overfeat: Integrated recognition, localization and detection using convolutional networks, in: ICLR, 2014.

[44] N. Neverova, C. Wolf, G. Taylor, F. Nebout, Hand segmentation with structured convolutional learning, in: ACCV, 2014.

[45] J. Long, E. Shelhamer, T. Darrell, Fully convolutional networks for semantic segmentation, in: CVPR, 2015.

[46] H. Noh, S. Hong, B. Han, Learning deconvolution network for semantic segmentation, in: ICCV, 2015.

[47] C. Szegedy, W. Liu, J. Yangqing, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. R. et al, Going deeper with convolutions, in: CVPR, 2015.

[48] M. Lin, Q. Chen, S. Yan, Network in network, in: ICLR, 2014.

[49] S. Ioffe, C. Szegedy, Batch normalization: Accelerating deep network training by reducing internal covariate shift, in: ICML, 2015.

[50] D. Kingma, J. L. Ba, Adam: A method for stochastic optimization, in: ICLR, 2015.

[51] R. Collobert, K. Kavukcuoglu, C. Farabet, Torch7: A matlab-like environment for machine learning, in: NIPS BigLearn Workshop, 2011.

[52] M. Muja, D. Lowe, Fast approximate nearest neighbors with automatic algorithm configuration, in: VISAPP, 2009.

[53] D. Bouchacourt, M. P. Kumar, S. Nowozin, Disco nets: Dissimilarity coefficient networks, in: NIPS, 2016.

[54] C. Xu, N. Govindarajan, Y. Zhang, L. Cheng, Liex: Depth image based articulated object pose estimation, tracking, and action recognition on lie groups, in: IJCV, 2016.

[55] K. Zhou, Q. Wan, W. Zhang, X. Xue, Y. Wei, Model-based deep hand pose estimation, in: IJCAI, 2016.

[56] X. Deng, S. Yang, Y. Zhang, P. Tan, L. Chang, H. Wang, Hand3d: Hand pose estimation using 3d neural network, in: arXiv:1704.02224, 2017.

[57] H. Guo, G. Wang, X. Chen, C. Zhang, F. Qiao, H. Yang, Region ensemble network: Improving convolutional network for hand pose estimation, in: ICIP, 2017.

[58] C. Wan, T. Probst, L. van Gool, L. Yo, Crossing nets: Dual generative models with a shared latent space for hand pose estimation, in: CVPR, 2017.

[59] D. Fourure, R. Emonet, E. Fromont, D. Muselet, N. Neverova, A. Tremeau, C. Wolf, Multi-task, multi-domain learning: application to semantic segmentation and pose regression, in: Neurocomputing, 2017.

[60] X. Zhang, C. Xu, Y. Zhang, T. Zhu, L. Cheng, Multivariate regression with grossly corrupted observations: A robust approach and its applications, in: arXiv:1701.02892, 2017.

[61] M. Madadi, S. Escalera, X. Baro, J. Gonzalez, End-to-end global to local cnn learning for hand pose recovery in depth data, in: arXiv:1705.09606, 2017.

[62] M. Oberweger, V. Lepetit, Deepprior++: Improving fast and accurate 3d hand pose estimation, in: ICCV workshop, 2017.

[63] B. Andres, T., Beier, J. Kappes, OpenGM: A c++ library for discrete graphical models, arXiv e-prints (2012).

[64] S. Ioffe, C. Szegedy, Batch normalization: Accelerating deep network training by reducing internal covariate shift, in: ICML, 2015.

[65] D. Kingma, J. Ba, Adam: A method for stochastic optimization, in: ICML, 2015.