Efficient and Robust Deep Networks for Semantic Segmentation

Gabriel L. Oliveira, Claas Bollen, Wolfram Burgard and Thomas Brox

Abstract— **This paper explores and investigates Deep Convolutional Neural Networks (DCNNs) architectures to increase efficiency and robustness of semantic segmentation tasks. The proposed solutions are based on Up-Convolutional Networks. We introduce three different architectures in this work. The first architecture, called Part-Net, is designed to tackle the specific problem of human body part segmentation and to provide robustness to overfitting and body part oclussion. The second network, called Fast-Net, is a network specifically designed to provide the lowest computation load without loosing representation power. Such architecture is capable of being run on mobile GPUs. The last architecture, called M-Net, aims to maximize the robustness characteristics of deep semantic segmentation approaches through multiresolution fusion. The networks achieve state-of-the-art performance on the PASCAL Parts Dataset and competitive results on the KITTI dataset for road and lane segmentation. Moreover, we introduce a new part segmentation dataset designed to bring semantic segmentation to highly realistic robotics scenarios, called Freiburg City Dataset. Additionally, we present results obtained with a ground robot and an unmanned aerial vehicle and a full system which explore the capabilities of human body part segmentation in the context of human-robot interaction.**

I. 1. Introduction

Convolutional Neural Networks (CNNs) are driving advances in many areas of visual perception, such as object detection [11]–[13], place recognition [38], localization [18] , visual odometry [1], [19] and classification [20], [37]. CNNs have the ability to learn effective hierarchical feature representations that characterize the typical variations observed in visual data, which makes them very well-suited for all visual classification tasks. While these previous approaches present state of the art results they still produce coarse inference and suffer from efficiency and robustness issues. The so-called up-convolutional networks extend CNNs towards fine inference, making pixel-wise prediction [8], [23].

In this paper we will explore and investigate techniques to increase efficiency and robustness of deep learning architectures for semantic segmentation. We introduce three different architectures. The first architecture, called Part-Net [30], is designed to tackle the specific problem of human body part segmentation and to provide robustness to overfitting and occlusion. The second network, called Fast-Net, is a network designed to provide a low computational power semantic segmentation architecture. The third architecture, called M-Net, aims to maximize the robustness of deep architectures using multiresolution fusion. We are interested in predicting high resolution segmentation masks as shown in Figure 1, in contrast to predicting a single class label per image.

In contrast to usual classification CNNs, which contract the high-resolution input to a low-resolution output, up-convolutional networks can take an abstract low-resolution input and predict a high-resolution output, such as a full-size image [8]. In [23], an up-convolutional network was attached to a classification network, which resolves the above-mentioned limitation: the contractive network part includes large receptive fields, while the up-convolutional part provides high localization accuracy. In this work, we demonstrate the power of up-convolutional networks to solve a wide range of segmentation tasks and propose architectures to solve human part segmentation and road and lane detection. Additionally a novel architecture based on multiresolution fusion is introduced to increase the robustness of these networks.

To evaluate the architectures we tested our techniques on the following datasets: PASCAL parts [6], Freiburg Sitting People and Freiburg People in Disaster [30]. We also tested on the KITTI-Road dataset [10] for road and lane segmentation and on Freiburg City dataset for realistic human body part segmentation in urban scenarios. Further experiments with a full robotic system were carried out to measure the behavior of our human body part segmentation approach in a human-robot interaction task.

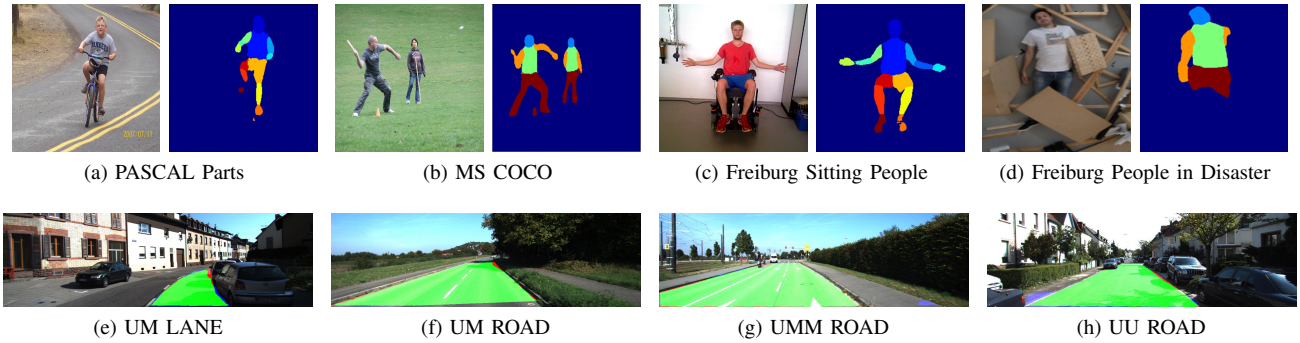
The results show that the proposed architectures achieved their desired goals with substantial improvements in efficiency and robustness. For road and lane segmentation, we show competitive results. In the other test cases we outperform the compared techniques in terms of computational performance and accuracy.

The paper is organized as follows. We first discuss related work in Section 2. In Section 3, we present the proposed architectures. Experimental results are described in Section 4. Ongoing work and possible future research directions are discussed in Section 5.

II. 2. Related Work

In the last two years deep learning approaches have achieved state of the art performance in semantic segmentation. Previous approaches rely on pre or post-processing and encode segmentation relations using Conditional Random Fields (CRFs) [4], [25], [32]. [32] presented an approach that couples local image features with a CRF and an image classification approach to combine global image classification with local segmentation. Another branch of CRFs called Hierarchical Conditional Random Fields (HCRF) has been introduced by [4]. They proposed a technique called harmony

All authors are with the Department of Computer Science at the University of Freiburg, 79110 Freiburg, Germany. This work has been supported by the European Commission under ERC-StG-PE7-279401-VideoLearn, ERC-AG-PE7-267686-LIFENAV, FP7-610603-EUROPA2, by the Freiburg Graduate School of Robotics, and by the Baden-Württemberg-Stiftung.



(a) PASCAL Parts (b) MS COCO (c) Freiburg Sitting People (d) Freiburg People in Disaster

(e) UM LANE (f) UM ROAD (g) UMM ROAD (h) UU ROAD

Fig. 1: Input image (left) and the corresponding mask (right) predicted by our networks for human body part segmentation, in the top row. The bottom row shows obtained results for road and lane segmentation.

potential to overcome a problem of classical HCRFs, which do not allow multiple classes to be assigned to a single region. [25] used an alternative approach for people detection and segmentation, in which they merge the outputs of a top-down part detector in a generalized eigen problem, producing pixel groupings.

The first method capable of processing arbitrarily-sized inputs was introduced in [26]. They proposed a fully convolutional network (FCN) to recognize strings of digits. This network was restricted to one-dimension input strings and the authors used Viterbi decoding to obtain the recognition outputs. Another pioneering work is from [41]. They extended a FCN to estimate a two-dimensional score detection map for the four corners of postal address blocks. Aforementioned methods were applied to detection and had only two or three layers. Recently much deeper fully convolutional networks were applied to semantic segmentation [31], image restoration [9] and detection [35]. [31] proposed a recurrent neural network for scene segmentation, which works on image patches. [35] proposed a CNN sliding window approach which simultaneously recognizes, locates and detects objects.

In contrast to previous FCN networks, the architecture proposed by [23] allows training the network end-to-end for semantic segmentation tasks using the whole image as input. The proposed method does not require any pre or post-processing method, unlike earlier approaches which rely on segmenting image patches and additional techniques to perform segmentation [29], [31]. The concept of [23] shows better performance and provides the state-of-the-art results in generic semantic segmentation problems. It replaces the fully connected layers of a deep classification network, e.g. VGG [37], by convolutional layers that produce coarse score maps. Successive up-convolutional refinements allows them to increase the resolution of these score maps. This kind of architecture was used for other problems. These new tasks include optical flow [7], depth estimation [21], edge detection [42] and region proposal generation [12].

The latest methods building on the FCN architecture are [3], [22], [30], [33], [34]. Our previous paper [30] discusses the Part-Net architecture, where we focused on human part segmentation and optimized the use of dropout to prevent over-fitting increasing robustness to occluded parts. [22] proposed a new architecture called ParseNet which focuses on global pooling and can model global context directly. [33] presented the U-Net, an up-convolutional network for pixel classification of microscopy images. [34] presented a FCN architecture for detection and tracking of liquids. The authors also utilized a long short-term memory (LSTM) recurrent cell to incorporate temporal information to the FCN based approach. [3] proposed a variation of the FCN architecture focusing on computational efficiency called SegNet. Their main contribution is related to improving the network's performance by the use of pooling indices computed in the max-pooling step to perform upsampling. The authors claim such approach eliminates the need for learning to upsample and reduces the systems memory requirements. Our low computation requirement architecture (Fast-Net) is different from Seg-Net with regard to the proposed architectural changes and to the identification and suppression of bottlenecks of FCNs.

Several works have been proposed on animal part segmentation and person keypoint prediction [16], [36], [39], [43], [44]. [43] performed part detection based on region proposals that are classified using a CNN. The approach was tested on a bird part segmentation dataset. [39] developed an approach that learns an end-to-end human keypoint detector for pose estimation using a CNN. [16] presents a sliding window approach for part localization with CNNs. They employ a CNN at each position of the window to detect human body parts. This requires thousands of CNN evaluations per image and makes the approach comparatively slow. None of these approaches has been applied to human body part segmentation.

Road and lane segmentation benchmarks are currently dominated by the following deep learning approaches [2], [5], [28]. [2] introduced a road scene segmentation approach that learns a classifier based on hand-crafted features, creating the training samples for a CNN network. The network learns specific domain features based on the machine-generated annotations. [5] introduced convolutional patch networks, which are CNNs designed for patch segmentation, allowing pixel-wise labeling. This technique also explicitly provides

spatial information of the patch to the network, allowing incorporation of a spatial prior to the network. [28] presented a CNN architecture in combination with deconvolutions. The author proposed a multi-patch technique that learns region-specific features, each patch region is trained on a separate network. This method currently provides the best results on the KITTI benchmark. While being less deep than our architectures, the proposed network is computational costly and is not able to provide interactive frame rates.

In this paper, we propose multiple architectures which explore specific task features (Part-Net), an architecture designed to provide maximum trade-off between computational burden and segmentation accuracy (Fast-Net) and an architecture created to provide higher robustness through multiresolution fusion (M-Net). Our approach is built upon the architecture from [23] and introduces several useful modifications. In particular, our architectures explore all available pooling layers for refinement, present a new refinement stage and propose a new dropout layer to further extend the robustness of such architectures to occlusion and clutter scenarios. We also share similar objectives with [3] and [22] in terms of better computational performance and segmentation robustness, respectively. We show in Section 4 a set of experiments directly comparing our architectures to the aforementioned techniques.

III. 3. Methodology

Semantic segmentation associates each pixel i of an input image x to one of N_{cl} class labels. In our approach, this task is learned end-to-end from a training set D, which consists of N input images x_n and their corresponding ground truth segmentation y_n, which is given as a N_{cl}-channel image with one-hot encoding. The learned network model $f(x;\theta)$ is described by a parameter vector θ, which is optimized during the training procedure according to the loss function

$$\mathcal{L}(\theta; D) = \sum_{n=1}^{N} \sum_{k=1}^{N_{cl}} \log(y_n) \mathrm{smax}(f_k(x_n;\theta)) \qquad (1)$$

where smax is the softmax function

$$\mathrm{smax}(\mathrm{a_k}) = \frac{\exp(\mathrm{a_k})}{\sum_{\mathrm{l}=1}^{\mathrm{N_{cl}}} \exp(\mathrm{a_l})} \qquad (2)$$

over all classes. After training, we set the output at each pixel to the class label k with the largest activation $f_k(x;\theta)$.

A. 3.1 Part-Net

Part-Net is a network designed to provide highly accurate human body part segmentation. The main novelties of such architecture is the refinement procedure and the feature map dropout module proposed by us in [30].

The network architecture is shown in Fig. 2. The parameters of the contracting part of the network are initialized with the parameters of the VGG classification network [37] to save training time.

The proposed refinement architecture is composed of multiple layers, where each layer combines the upsampled output of its previous layer with the pooled features of the corresponding layer of the contracting network part. The former provides the preliminary class scores at the coarse resolution, whereas the latter contributes information for refining the resolution. The combination of both is described in Fig. 3. The coarse score map is fed into an up-convolutional layer, i.e., it is upsampled by a factor 2 via bilinear interpolation followed by a convolution. We use a ReLU activation function after each up-convolutional operation to avoid the vanishing gradient problem. The feature map from the contracting network part is fed into a convolutional layer followed by dropout to improve the robustness to over-fitting. Finally, the output of both streams are summed element-wise to yield the output of the refinement layer. This output is the input for the next refinement layer. Each layer increases the resolution of the segmentation by a factor 2.

With this refinement architecture we manage to obtain a high quality output at the resolution of the input image. This is in contrast to [23], who stopped their refinement after three layers, because they did not observe any improvement afterwards. A full description of the architecture is presented at Table I and results of an ablation study are shown in Section 4.

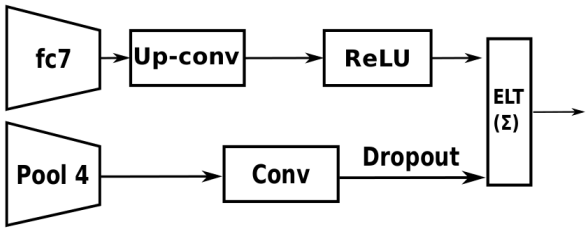


Fig. 3: Description of the first refinement layer. Successive refinement layers have the same architecture, but take different inputs. The upper stream takes the output from the contractive network (fc7-conv) or from the previous refinement layer as input. It applies an up-convolution followed by a ReLU. The lower stream takes high-resolution features from the corresponding layer in the contractive network as input.

1) Feature Map Dropout:: To make Part-Net more robust and specialized to human part segmentation we implement a feature map dropout technique. The method consists of extending the dropout to the entire feature map, not just to some pixels like on the standard dropout. Feature map dropout is based on the Spatial Dropout method [39] where the authors also aim to increase robustness of CNNs, but in this case, for human pose estimation. Spatial correlation is a singular characteristic in human body part segmentation which must be explored. Hence, dropout must also be correlated. Feature map dropout performs a standard Bernoulli trial per output feature during training, yet propagates the dropout value across the entire feature map.

Given a network with L hidden layers and $l \in \{1, ..., L\}$. Let z^l be the vector of inputs into layer l and let y^l denote the vector of outputs from layer l. W^l and b^l are the weights

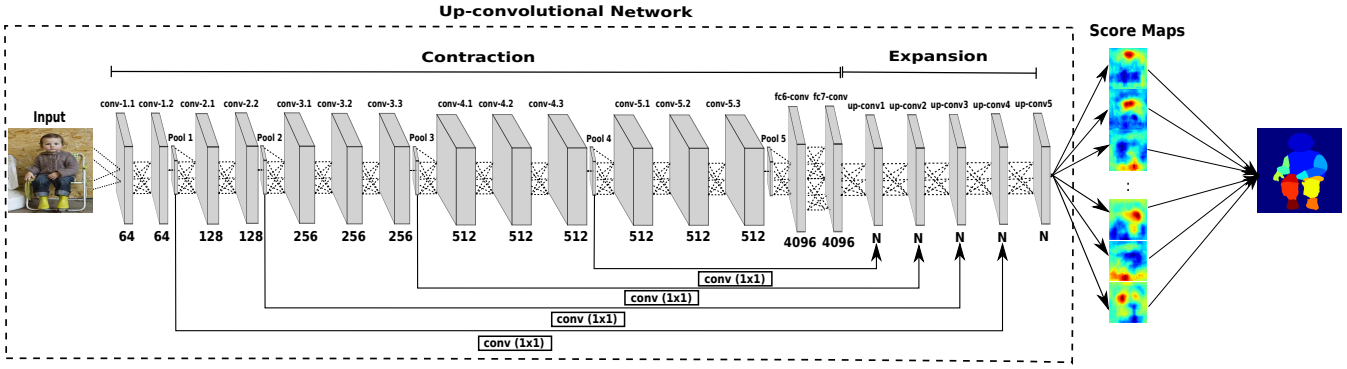


Fig. 2: Part-Net architecture. Only convolutional, pooling, and up-convolutional layers are visualized. Up-convolutional layers have size $N = N_{cl}$. We call the network part up to fc7-conv the *contractive network part*, whereas the part after fc7-conv is called the *expansive network part*.

name	kernel size	stride	pad	output size
data	-	-	-	$300 \times 300 \times 3$
conv1_1	3×3	1	100	$498 \times 498 \times 64$
conv1_2	3×3	1	1	$498 \times 498 \times 64$
pool1	2×2	2	0	$249 \times 249 \times 64$
conv2_1	3×3	1	1	$249 \times 249 \times 128$
conv2_2	3×3	1	1	$249 \times 249 \times 128$
pool2	2×2	2	0	$125 \times 125 \times 128$
conv3_1	3×3	1	1	$125 \times 125 \times 256$
conv3_2	3×3	1	1	$125 \times 125 \times 256$
conv3_3	3×3	1	1	$125 \times 125 \times 256$
pool3	2×2	2	0	$63 \times 63 \times 256$
conv4_1	3×3	1	1	$63 \times 63 \times 512$
conv4_2	3×3	1	1	$63 \times 63 \times 512$
conv4_3	3×3	1	1	$63 \times 63 \times 512$
pool4	2×2	2	0	$32 \times 32 \times 512$
conv5_1	3×3	1	1	$32 \times 32 \times 512$
conv5_2	3×3	1	1	$32 \times 32 \times 512$
conv5_3	3×3	1	1	$32 \times 32 \times 512$
pool5	2×2	2	0	$16 \times 16 \times 512$
fc6-conv	7×7	1	0	$10 \times 10 \times 4096$
fc7-conv	1×1	1	0	$10 \times 10 \times 4096$
Up-conv1	4×4	2	0	$22 \times 22 \times N_{cl}$
Up-conv2	4×4	2	0	$46 \times 46 \times N_{cl}$
Up-conv3	4×4	2	0	$94 \times 94 \times N_{cl}$
Up-conv4	4×4	2	0	$190 \times 190 \times N_{cl}$
Up-conv5	4×4	2	0	$382 \times 382 \times N_{cl}$
output	-	-	-	$300 \times 300 \times N_{cl}$

TABLE I: Part-Net architecture in more detail. The Up-conv layers refer to each refinement step. For brevity reasons ReLUs, dropout and some layers from the up-convolution step are not shown.

and biases at layer l. $(*)$ denote the element-wise product and $r^{(l)}$ is the vector of independent Bernoulli random variables that has probability p of being 1. Feature Map Dropout is then expressed as

$$r_z^{(l)} \sim \text{Bernoulli}(p)$$
$$\tilde{y}^{(l)} = r^{(l)} * y^{(l)}$$
$$z_i^{(l+1)} = w_i^{(l+1)} \tilde{y}^{(l)} + b_i^{(l+1)}$$
$$y_i^{(l+1)} = f(z_i^{(l+1)})$$

The variable $\tilde{y}^{(l)}$ is called thinned vector of outputs. This sets apart the feature map dropout from the standard dropout. The resulting thinner network $\tilde{y}^{(l)}$ has the entire feature maps zeroed. For instance, in a convolution layer of size $(1, 64, 20, 20)$, and a dropout of 0.5, approximately 32 of the 64 feature channels will be zeroed after the input passes the dropout layer.

Part-Net experiments are presented in Section 4 and ratify the effectiveness of our architecture to human body part segmentation. It yelds state-of-the-art results but does not provide interactive frame rate computation. Our next architecture aims to mitigate this limitation.

B. 3.2 Fast-Net

Fast-Net is an up-convolutional architecture which tackles the main inefficiencies of such architectures. Our goal is to provide the best trade-off between performance and segmentation accuracy. Fast-Net is shown in Fig. 4 and a detailed specification of the individual network layers are given in Table II.

1) Optimizing the Use of Parameters:: The main motivation to design Fast-Net is to make up-convolutional networks efficient in terms of memory and runtime, while keeping high quality segmentation. Such requirements emerged from the need to provide accurate and efficient segmentation for road and lane detection. To meet these requirements we optimized the number of network parameters.

2) Parameter reduction:: The fully convolutional network from [23] and Part-Net use the VGG-16 classification network as basis for the contraction side of the network. This network has 4096 filters with 7×7 spatial size. The large

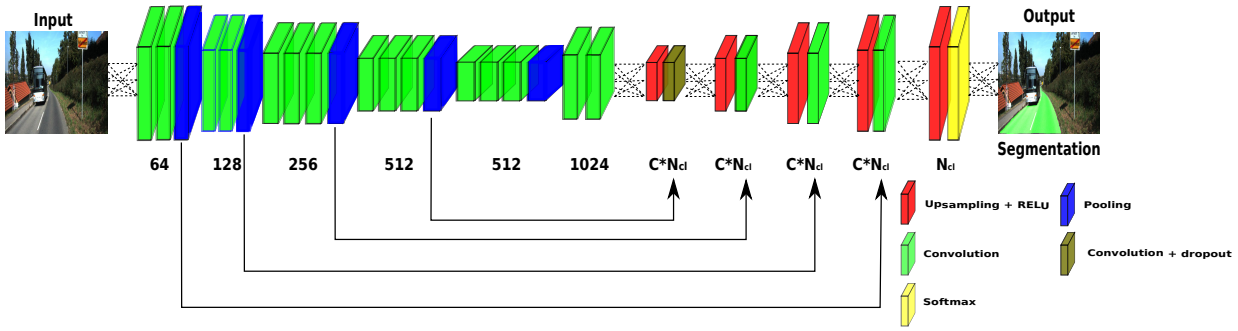


Fig. 4: Fast-Net architecture. Up-convolutional layers have size equal to $C * N_{cl}$. C stands for the scalar factor of filters augmentation.

name	kernel size	stride	pad	output size
data	-	-	-	$500 \times 500 \times 3$
conv1_1	3×3	1	10	$518 \times 518 \times 64$
conv1_2	3×3	1	1	$518 \times 518 \times 64$
pool1	2×2	2	0	$259 \times 259 \times 64$
conv2_1	3×3	1	1	$259 \times 259 \times 128$
conv2_2	3×3	1	1	$259 \times 259 \times 128$
pool2	2×2	2	0	$130 \times 130 \times 128$
conv3_1	3×3	1	1	$130 \times 130 \times 256$
conv3_2	3×3	1	1	$130 \times 130 \times 256$
conv3_3	3×3	1	1	$130 \times 130 \times 256$
pool3	2×2	2	0	$65 \times 65 \times 256$
conv4_1	3×3	1	1	$65 \times 65 \times 512$
conv4_2	3×3	1	1	$65 \times 65 \times 512$
conv4_3	3×3	1	1	$65 \times 65 \times 512$
pool4	2×2	2	0	$33 \times 33 \times 512$
conv5_1	3×3	1	1	$33 \times 33 \times 512$
conv5_2	3×3	1	1	$33 \times 33 \times 512$
conv5_3	3×3	1	1	$33 \times 33 \times 512$
pool5	2×2	2	0	$17 \times 17 \times 512$
FC-conv	3×3	1	0	$15 \times 15 \times 1024$
FC-conv2	1×1	1	0	$15 \times 15 \times 1024$
conv-N_{cl}	1×1	1	0	$1 \times 1 \times N_{cl}$
Up-conv1	4×4	2	0	$40 \times 40 \times CN_{cl}$
Up-conv2	4×4	2	0	$82 \times 82 \times CN_{cl}$
Up-conv3	4×4	2	0	$166 \times 166 \times CN_{cl}$
Up-conv4	4×4	2	0	$294 \times 294 \times CN_{cl}$
Up-conv5	4×4	2	0	$590 \times 590 \times N_{cl}$
output	-	-	-	$500 \times 500 \times N_{cl}$

TABLE II: Fast-Net architecture in more detail. The Up-conv layers refer to each refinement step.

number of filters and its corresponding size, constitutes the main concentration of parameters and consequently the major computational load of the network. Another related computational bottleneck is the extensive use of padding, Table II shows that Fast-Net can process an input image of 500×500 pixels with almost the same feature map dimensions as Part-Net with an input image of size 300×300.

To address this problem, we reduced the number of parameters by reducing the number of FC-conv filters from 4096 to 1024. In addition, we reduce the size of the filters from 7×7 to 3×3. The proposed reduction of network parameters makes our approach more efficient than the previously proposed dense segmentation architectures. From such a reduction, one must expect a significant drop in classification accuracy. However, we use some of the saved parameters at another part of the network to keep the high accuracy and even improve it compared to the baseline network.

3) New refinement to improve system accuracy:: In order to make Fast-Net capable of producing segmentation masks comparable or superior to the network before the parameter reduction, we give more parameters to the up-convolutional side of the network. Previous up-convolutional networks like [23] and Part-Net have a $one-to-one$ mapping, where each refinement has the same number of filters and classes (N_{cl}). We based our new distribution of parameters on those of the U-nets [33]. U-nets have a variable number of filters that is equal to the corresponding number of filters of the layer at the contraction part. Such approach was successfully applied to medical images, yet it was never shown to work with natural images because it is quite large and slow. In order to provide more parameters to the expansive part without hurting the computational time we propose a similar approach that uses multiple filters per class like U-net, but without the considerable increase of the network parameters. We increase the number of filters by a scalar C. The additional parameters to the expansive side provide further capacity to Fast-Net to produce more accurate upsampled masks and consequently better segmentation predictions. The small increase in the number of parameters hardly increases the computational cost and makes Fast-Net more robust to scale.

The new architecture has $C * N_{cl}$ filters in almost all expansive layers except two layers. These are the convolutional layer between the contraction and expansion part of the network (conv-N_{cl}) and the last layer of the architecture. For these layers $C = 1$, to maintain the network efficiency and to produce class scores, respectively. The main purpose of the layer between the contraction and expansion side is to maintain the network efficiency, while the last one has as goal to make the network calculate the loss over only the useful classes.

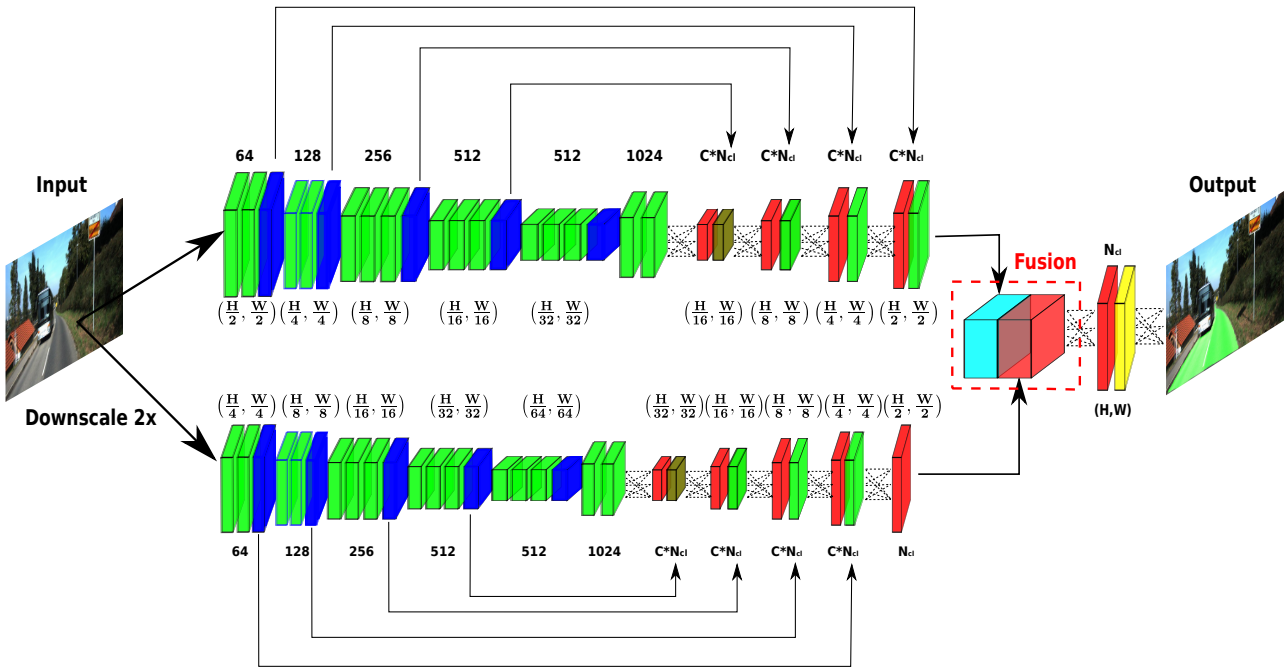


Fig. 5: Description of the M-Net architecture. The architecture has two streams, high and low. The low stream is half the resolution of the high stream and is an auxiliary stream that is fused to the main stream according to the resolution.

C. 3.3 M-Net

Multiresolution Net (M-Net) is an architecture inspired by inter network refinement of multiresolution streams. The main goal of the M-Net architecture is to incorporate characteristics of multiple resolutions into a single architecture. This is done by the fusion of features from a low resolution network into a high resolution one. Lower resolution networks have low false positive detection rates. By combining low resolution features from an auxiliary network into the predictions of our main network stream we are capable of providing high resolution segmentation masks together with low false positive detection rates. Such approach can leverage the semantic information of up-convolutional architectures and reinforce predictions which are consistent among resolutions.

Such approach is inspired by the coloring network of [15], where the authors proposed a multi-stream network for image coloring. Here, the two streams are Fast-Net architectures which are fused to improve the predicted mask from the high resolution stream. The complete architecture is shown in Figure 5. The key element of this architecture is its fusion layer.

1) Fusion Layer:: To include low resolution features in high resolution predictions we introduce a fusion layer. We formulate the fusion layer as:

$$y^{fusion} = \sigma\left(b + W\left[y^{high} + y^{low}\right]\right) \tag{3}$$

where y^{fusion} is the fused feature, y^{high} is the high resolution features, y^{low} is the low resolution features and W and b are the weights and biases. The fusion can be interpreted as a element-wise sum of high-level and low-level features maps. Incorporating low resolution features to a high resolution network makes M-Net more robust towards false positive detections.

D. 3.4 Data Augmentation

Data augmentation is needed for every tested application, due to the reduced number of training examples provided by these datasets. Very deep networks, like the ones proposed in this work, require a large number of examples. To this end, we employ a series of data transformations to provide more examples. In particular, we implemented:

- Scaling: Scale the image by a factor from 0.7 and 1.4
- Rotation: Rotate the image by an angle of up to 30 degrees
- Color: Add a value between -0.1 and 0.1 to the hue channel of the HSV representation.

In Section 4 we show in a dedicated experiment the importance of data augmentation, which is different from the fidings of [23].

In the case of body part segmentation all augmentation transformations positively impact the results. However, for the specific case of road and lane segmentation, rotation and cropping transformations are undesirable, since the network is expected to learn spatial priors of the road.

E. 3.5 Network Training

Training was performed in a multi-stage process, except for M-Net where we explored single stage training without VGG initialization of the contraction side. Without such initialization, training time increases by a factor of two. For the other techniques we initialized the contracting part of the

network with the 16 layer version of the VGG architecture [37].

The networks were trained by backpropagation using Stochastic Gradient Descent (SGD) with momentum. Each minibatch consisted of just one image. The learning rate and momentum for Part-Net were fixed to $1e^{-10}$ and 0.99, respectively. However for Fast-Net and M-net, we reduced the padding of the first convolutional layer from 100 to 10 pixels (slightly faster training), used Xavier initialization, increased learning rate from 1e-10 to 1e-9 and decreased the momentum from 0.99 to 0.90. We also changed the fixed learning rate (L_r) by a polynomial learning policy

$$L_r = L\,(1 - i/max_i)^p\,, \tag{4}$$

where L is the base learning rate, i is the learning step and p is the power index. The new policy converges faster than the fixed learning rate policy. On average the multi-stage training required half the number of iterations to obtain the same results.

The multi-stage training was done one refinement stage at time and each refinement took 24 hours. Thus, training the whole network takes 5 days on a single GPU. For M-Nets we needed twice this amount.

IV. 4. Experiments

We evaluated the performance of our networks on two different problems. First we present a series of experiments focusing on human body part segmentation. The second set of results focused on road and lane segmentation. Our full source code is publicly available at [1].

For human part segmentation we tested our method on PASCAL Parts dataset, Freiburg Sitting People dataset, Freiburg People in Disaster dataset and Freiburg City Dataset. On the first three datasets we report quantitative results and compare to results obtained with the state-of-the-art FCN baseline [23]. We fine-tuned the FCN for each dataset on the same training data that was used for training our network. Moreover, we conducted experiments in a direct robotics context with a ground robot and an unmanned aerial vehicle. A full robotic system experiment is presented to measure the behaviour of our human body part technique when exposed to an interaction scenario between a human and a robot.

Subsequently, we evaluated the performance of Fast-Net on real driving data from the KITTI benchmark dataset. We present a series of evaluations in terms of runtime, accuracy, and scale robustness. The implementation was based on the publicly available Caffe [17] deep learning toolbox, and all experiments were carried out with a system containing an NVIDIA Titan X GPU.

A. 4.1 PASCAL Parts dataset

The PASCAL Parts dataset [6] includes annotations for 20 PASCAL super classes and part annotations for each of them. We focused on the person subset of this dataset, which consists of 3539 images. The annotations even include eyes and ears, which may not seem relevant in a robotics context for now. Therefore we merged labels to two granularity levels: coarse and fine. For the coarse version we consider four labels (head, torso, arms, legs). In the finer version, we have 14 labels and distinguish also between the left and right side of the person (head, torso, upper right arm, lower right arm, right hand, upper left arm, lower left arm, left hand, upper right leg, lower right leg, right foot, upper left leg, lower left leg and left foot).

We cannot compare our results with other approaches since there is no paper reporting human body part segmentation. To the best of our knowledge, the only results reported so far are [24], [40], though none of them have reported results on the person category. Therefore, we present the first quantitative results on person part segmentation for the PASCAL Parts dataset.

As metrics, we chose pixel accuracy and intersection over union. Let n_{ij} be the number of pixels of class i predicted to belong to class j, where $t_i = \sum_j n_{ij}$ be the total number of pixels of class i. The pixel accuracy $Acc = \sum_i n_{ii} / \sum_i t_i$ takes into account also the prediction of background pixels. Background prediction is important to avoid false positives.

The downside of pixel accuracy as a sole measure, however, is the dominance of the background in the metric. More than three quarters of the images is background. Therefore, along with pixel accuracy, we also report the intersection over union (IOU), which is a popular metric for computer vision datasets. It is defined as $IOU = (1/N) \sum_i n_{ii}/(t_i + \sum_j n_{ji} - n_{ii})$. Unlike pixel accuracy, IOU does not take the background detection into account and solely measures the semantic segmentation of the parts. However, it does penalize false positive pixel assignments.

1) Coarse body parts:: We first predicted the coarse segmentation with four body part classes. We randomly divided the dataset into 70% training and 30% testing. Table III shows the results. There is a 5% percentage points improvement over the state of the art in both metrics. Further comparison against other FCN approaches are provided in Table XVIII.

TABLE III: Results on PASCAL dataset with 4 body parts.

Method	Accuracy	IOU
FCN [23]	71.30	57.35
Part-Net - No dropout	74.60	61.20
Part-Net - With dropout	**76.58**	**63.03**

Additionally, we also performed experiments without the feature map dropout at the refinement part of the network. Table III presents our results for the network without feature map dropout at the expansive part of the network and with dropout. The addition of the dropout layer brings a considerable gain, in terms of better mean pixel accuracy and IOU. This result confirmed that a spatially correlated dropout can benefit from the strong spatial correlation of human body parts. Since the results showed a consistent gain when including feature map dropout compared to standard

[1] http://lmb.informatik.uni-freiburg.de/resources/binaries/

dropout, all the following experiments with Part-Net were run with feature map dropout.

TABLE IV: Results on the PASCAL dataset with 14 body parts.

Method	Accuracy	IOU
FCN [23]	75.60	53.12
Part-Net	**77.00**	**54.18**

2) Detailed body parts:: When predicting all 14 body parts, we randomly divided the dataset into 80% training and 20% testing. Figure 6 shows a set of results obtained by Part-Net. The results are organized column-wise, where each column is an example and the rows correspond to input image, ground truth and results obtained using the FCN of [23]. The last row shows the results using our network. The results of Part-Net are closer to the ground truth than the FCN baseline. Table IV contains the corresponding quantitative numbers. We outperform the FCN baseline by 1% percentage point in both metrics. The smaller improvement on the more complex task indicates that there was not enough training data to exploit the larger capacity of our network. For the experiments reported so far, we did not make use of any data augmentation. In the next section we discuss the importance of data augmentation, especially for more complex tasks.

B. 4.2 Effect of Data Augmentation

Apart from the usual mirroring and cropping, we applied two types of augmentations to our training data: spatial augmentations and color augmentation; see the detailed description in Section 3.4. Table VI shows the impact of these types of data augmentation for 4 body parts, while Table V presents the results for all 14 body parts. What can be noticed from the fine granularity results is the difficulty of the network to segment the extremities, mainly based on its low area and wide range of possible orientations. Table VII summarizes the IOU along with the pixel accuracy for the fine granularity level. Clearly, both types of data augmentation improved results significantly. These results emphasize the importance of a solid data augmentation technique when approaching relatively complex tasks with limited training set sizes. The relative improvement of data augmentation was more prevalent on the more difficult task with 14 classes, which can be attributed to the fact that, a more difficult task requires more training data.

TABLE VI: Augmentation results Accuracy and IOU on the PASCAL dataset with 4 body parts.

Method	Acc.	IOU				
		Head	Torso	Arms	Legs	All
FCN [23]	71.30	70.74	60.62	48.44	50.38	57.35
Part-Net	76.58	75.08	64.81	55.61	56.72	63.03
Part-Net(Spatial)	82.18	80.49	74.39	67.17	70.39	73.00
Part-Net(Spatial+Color)	**85.51**	**83.24**	**79.41**	**73.73**	**76.52**	**78.23**

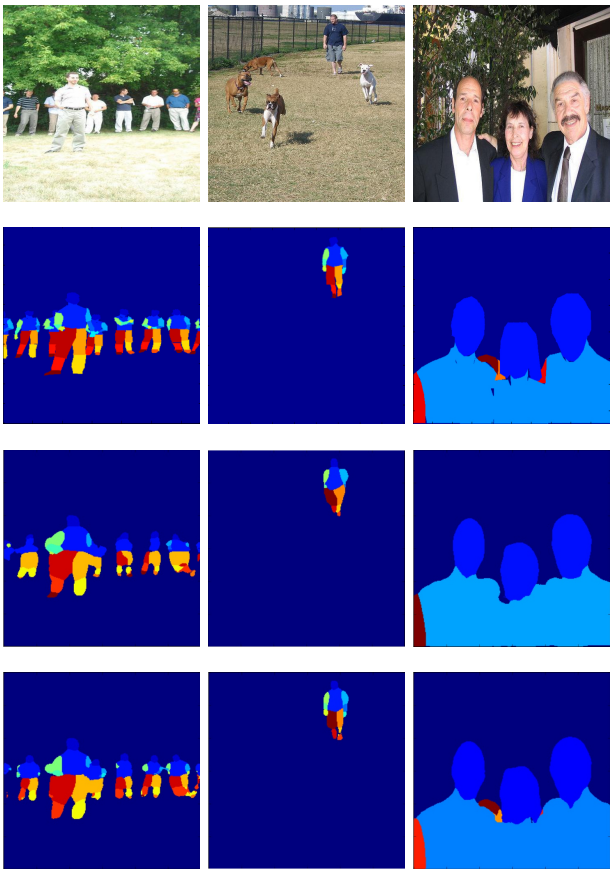


Fig. 6: Qualitative results on the PASCAL dataset (task with 14 body parts). **First row:** Input image. **Second row:** Ground truth. **Third row:** Result predicted with FCN [23]. **Fourth row:** Result predicted by our network. Part-Net produces more accurate segmentation masks, not only for single person segmentation but also when there are multiple persons in the image.

TABLE VII: Augmentation summary on the PASCAL dataset with 14 body parts.

Method	Accuracy	IOU
FCN [23]	75.60	53.12
Part-Net	77.00	54.18
Part-Net (Spatial)	84.19	66.93
Part-Net (Spatial + Color)	**88.20**	**71.71**

C. 4.3 Effect of Layer Refinement

In order to measure the effects of each refinement on the segmentation quality, we perform experiments testing all possibilities of previous layers inclusion to the proposed architecture. The experiments include the IOU values per set of refinements for PASCAL coarse and fine granularities, see Table VIII.

The results show that the inclusion of all pooling layers in the refinement process is beneficial for segmentation, providing a total gain above 20% percentage points. Such conclusion is opposite to [23], who pointed out that after fusing predictions from $pool3$ only minor improvements

TABLE V: Augmentation results (IOU) on the PASCAL dataset with 14 body parts.

Method	Head	Torso	L U arm	L LW arm	L hand	R U hand	R LW arm	R hand	R U leg	R LW leg	R foot	L U leg	L LW leg	L foot	Mean
FCN [23]	74.0	66.2	56.6	46.0	34.1	58.9	44.1	31.0	49.3	44.5	40.8	48.5	47.6	41.2	53.1
Part-Net (Spatial)	81.8	78.0	69.5	63.1	59.0	71.2	63.0	58.7	65.4	60.6	52.0	67.9	60.3	50.0	66.9
Part-Net (Spatial+Color)	**84.0**	**81.5**	**74.1**	**68.0**	**64.0**	**75.4**	**67.4**	**61.9**	**72.4**	**67.1**	**56.9**	**73.0**	**66.1**	**57.7**	**71.7**

R = right, L = left, U = upper, LW = lower.

TABLE VIII: Refinement impact for PASCAL dataset with 4 and 14 body parts. The results constitute the IOU values per refinement stage.

Stage	IOU-4 parts	IOU-14 parts
No Refinement	57.01	51.69
Pool4	70.24	64.15
Pool4 + Pool3	73.31	66.80
Pool4 + Pool3 + Pool2	74.65	67.92
Pool4 + Pool3 + Pool2 + Pool1	**78.23**	**71.70**

were noticed. In our experiments all refinements present significant gains. The experiments show that $pool2$ and $pool1$ are responsible for an improvement of more than 4.9 percentage points with respect to the IOU metric. Therefore, exploiting all pooling layers can provide better segmentation masks with only a small computational burden.

D. 4.4 Freiburg Sitting People Part Segmentation Dataset

To evaluate Part-Net in a robotics scenario, we created a new dataset [2] that provides high resolution segmentation masks for people in sitting position, specifically people in wheelchairs. Part segmentation can be of great interest for robotics, since it provides to robots detailed body location, allowing fine interaction. Figure 7a, presents an input sample image and Figure 7b its ground truth segmentation, while Figure 7c shows the segmentation prediction. The dataset has 200 images of six different people from multiple viewpoints and wide range of orientations. The ground truth annotation contains the 14 body part classes as used for the PASCAL parts dataset.

(a) Input Image (b) Groundtruth (c) Predicted Mask

Fig. 7: Results on the Freiburg Sitting People dataset. Groundtruth in a different color mapping.

Due to the unavailability of a large amount of data, we chose two different testing scenarios. First we trained our network on the PASCAL parts dataset, and used the full sitting people dataset for testing. Alternatively, we randomly chose two people from the dataset for training (along with the data from PASCAL parts) and the remaining four as the test set. Results are shown in Table IX. The network generalized well to the new datasets. The improvement over the FCN baseline was much larger than the difference between the network that had seen sitting people for training and the one that had not. Nonetheless, providing training data that is specific to the task helped improve the performance.

TABLE IX: Results with and without training on the Freiburg people dataset.

Method	Accuracy	IOU
FCN [23]	59.69	43.17
Part-Net (Trained on PASCAL)	78.04	59.84
Part-Net (Training with 2 people, Testing with 4)	**81.78**	**64.10**

E. 4.5 KITTI Road/Lane Dataset

The KITTI Visual Benchmark Suite [10] is a dataset designed to benchmark optical flow, odometry data, object detection, and road/lane detection. The road benchmark consists of 600 frames of 375×1242 pixels and constitutes the main benchmark dataset for road and lane segmentation. The data was acquired in five different days. The main goal of the Fast-Net architecture is to provide high quality semantic segmentation predictions with the lowest possible computational burden. The road and lane segmentation problem motivate the design of such architecture based on its real-time requirement. M-Net was designed to provide more robustness to false positive detections, consequently better segmentation.

The dataset has three different categories of road scenes: single-lane road with markings (UM), single-lane road without markings (UU), and multi-lane road with markings (UMM). In this paper, we deal with road and ego-lane detection. We do not differentiate between the road categories, but the ego-lane problem is trained separately. The dataset provides ground truth for training and an evaluation website for testing[3]. The KITTI online evaluation system allows for anonymous submission of results, thereby some of the top ranked methodologies do not have a corresponding publication.

1) Road Detection:: There are four road segmentation tasks for the KITTI benchmark: UM, UU, UMM and URBAN ROAD. URBAN ROAD is the category that summarizes

[2] http://lmb.informatik.uni-freiburg.de/resources/datasets/PartSeg.html

[3] http://www.cvlibs.net/datasets/kitti/eval_road.php

TABLE X: Results for the road KITTI dataset.

Benchmark	MaxF	AP	PRE	REC	FPR	FNR
UM Fast-Net	92.20%	88.85%	92.57%	**91.83%**	3.36%	**8.17%**
UM M-Net	**93.31%**	**90.54%**	**95.11%**	91.57%	**2.14%**	8.43%
UMM Fast-Net	**95.52%**	92.86%	95.37%	**95.67%**	5.10%	**4.33%**
UMM M-Net	95.20%	**92.96%**	**96.03%**	94.38%	**4.28%**	5.62%
UU Fast-Net	92.65%	**89.20%**	92.85%	92.45%	2.32%	7.55%
UU M-Net	**92.97%**	88.38%	**93.36%**	**92.58%**	**2.15%**	**7.42%**
URBAN Fast-Net	93.83%	90.47%	94.00%	**93.67%**	3.29%	**6.33%**
URBAN M-Net	**94.09%**	**90.78%**	**95.13%**	93.08%	**2.63%**	6.92%

TABLE XI: Results on UM road KITTI dataset.

Method	MaxF	AP	PRE	REC	FPR	FNR	Time
DDN [28]	**93.65%**	88.55%	94.28%	93.03%	2.57%	6.97%	2s
M-Net	93.31%	**90.54%**	**95.11%**	91.57%	**2.14%**	8.43%	130ms
Fast-Net	92.20%	88.85%	92.57%	91.83%	3.36%	8.17%	**83ms**
CNN1 (anonymous)	91.73%	92.08%	91.10%	92.36%	4.11%	7.64%	2s
CNN (anonymous)	91.22%	91.35%	91.22%	91.23%	4.00%	8.77%	2s

TABLE XII: Results on UU road KITTI dataset.

Method	MaxF	AP	PRE	REC	FPR	FNR	Time
M-Net	**92.97%**	88.38%	**93.36%**	**92.58%**	**2.15%**	**7.42%**	130ms
Fast-Net	92.65%	89.20%	92.85%	92.45%	2.32%	7.55%	**83ms**
Fast-Net Low	91.89%	**89.44%**	92.59%	91.20%	2.38%	8.80%	**52ms**
DDN [28]	91.76%	86.84%	93.06%	90.50%	2.20%	9.50%	2s
CNN1 (anonymous)	89.70%	90.61%	89.41%	89.99%	3.47%	10.01%	2s

TABLE XIII: Results on UMM road KITTI dataset.

Method	MaxF	AP	PRE	REC	FPR	FNR	Time
Fast-Net	**95.52%**	92.86%	95.37%	95.67%	5.10%	4.33%	**83ms**
M-Net	95.20%	**92.96%**	**96.03%**	94.38%	**4.28%**	5.62%	130ms
DDN [28]	94.17%	92.70%	96.73%	91.74%	3.41%	8.26%	2s
FCN_LC [27]	94.09%	90.26%	94.05%	94.13%	6.55%	5.87%	**30ms**
Fast-Net Low	93.89%	92.62%	94.57%	93.22%	5.89%	6.78%	52ms

TABLE XIV: Results on URBAN_ROAD KITTI dataset.

Method	MaxF	AP	PRE	REC	FPR	FNR	Time
M-Net	**94.09%**	**90.78%**	**95.13%**	93.08%	**2.63%**	6.92%	130ms
Fast-Net	**93.83%**	90.47%	94.00%	93.67%	3.29%	6.33%	**83ms**
DDN [28]	93.43%	89.67%	95.09%	91.82%	2.61%	8.18%	2s
Fast-Net Low	92.39%	90.24%	93.03%	91.76%	3.79%	8.24%	**52ms**
CNN1 (anonymous)	91.98%	92.44%	91.08%	92.89%	5.01%	7.11%	2s

all three different road scene categories. Table X shows our results for road segmentation with Fast-Net and M-Net. The results confirm the capacity of M-Net to provide more accurate segmentation masks and to incorporate the false positive robustness of low resolution networks to high resolution ones. The individual methods are ranked according to their pixel-wise maximum F-measure on the bird's-eye view space. The other provided measurements are: Average Precision (AP), Precision (PRE), Recall (REC), False Positive Rate (FPR) and False Negative Rate (FNR).

The single-lane with markings category (UM) is formed by images taken from a marked urban two-way road and has 95 images for training and 96 images for testing. Our approach ranks second and third for this category, as seen in Table XI. While all top results for this category report processing times of about 2 seconds, the Fast-Net architecture has an average runtime of 83 milliseconds and M-Net of 130 milliseconds. This makes Fast-Net the only approach among the top performing methods that is capable of interactive frame rates.

For the single-lane road without markings (UU) we have 98 images for training and 100 images for testing. Our method ranks first and second for this set; see Table XII. The resolution used for most of the experiments was 500×500 but we also tested with a 300×300 resolution. More tests on the impact of resolution will be discussed in Section 4.6.

The third setting is composed of images taken from the urban multi-lane marked road (UMM), which has 96 images for training and 94 for testing. Results are shown in Table XIII, where the proposed network compares favorably to all existing techniques, as well. For this scenario a CNN method (FCN_LC) reports faster processing times but inferior

accuracy in all metrics.

The final metric of the KITTI evaluation benchmark is one that combines all three experimental settings (URBAN_ROAD). Table XIV presents our results and the best results available. M-Net achieves the highest accuracy among all top methods while Fast-Net presents comparable results to M-Net with the smallest runtime of all compared techniques. Figure 8 show a qualitative comparison of the results obtained with our architecture and the top three results. We chose the techniques that ranked best on URBAN_ROAD techniques for comparison, since this metric can provide a better overall measurement of how well a method behaves. The proposed architecture shows low occurrences of false positive predictions and sharp segmentation of edges, while enabling interactive frame rates.

2) Lane Detection:: Lane detection is a challenging task due to low inter-class variability. Without context, the ego-lane is hard to distinguish from other asphalt parts of the road. The KITTI dataset uses single-lane road marked images to segment lanes. Table XV shows our results. In contrast to road detection, there are methods with runtimes in the millisecond range. Fast-Net, which is the only one among the top techniques that was not specially designed for this task, achieves the best accuracy. Figure 9 presents some qualitative comparisons with the 3 best approaches. We can notice from the qualitative results that Fast-Net presents high fidelity ego-lane segmentation and low false positive and negative detections.

F. 4.6 Performance Tests

Since the runtime depends much on the hardware and resolution, we present results on various GPUs and at different resolutions for the Fast-Net architecture. We tested seven different desktop GPUs and two mobile ones; see Table XVI. Even on older GPUs, such as the GTX 680, the architecture achieves more than 10 frames per second and fits into the GPU memory. The tests also reveal that the network is as fast on a GTX 980, GTX TITAN X and on the recently release GTX 1070.

We further extend our experiments to mobile GPUs and proved Fast-Net is capable of runtime at speeds superior to one frame per second in modern mobile GPUs, such as TX1, and, to the best of our knowledge this is the first up-convolutional network capable of local processing in low power mobile GPUs. Such capability can leverage many new applications of semantic segmentation in robotics and improve the on-board perception of autonomous platforms.

We also varied the resolution. For the experiment we used a machine with a TITAN X GPU and tested inputs ranging from 150×150 to 500×500; see Table XVII. As expected, the runtime increases with higher resolutions. However, even at a 500×500 resolution, our system is still more efficient than any top result for road detection.

G. 4.7 Baseline Comparison

To compare to the state of the art, we trained all the networks with rotation, scale and color augmentations. We selected the top available deep architectures based on the reported results on semantic segmentation and on the methodologies which tackle, to a certain degree, efficiency and robustness. The results are reported from PASCAL PARTS with four body parts. We explored the parameter space to achieve the best baseline performance. Table XVIII summarizes our results using the corresponding metrics, such as Intersection over Union (IOU), Mean Pixel Accuracy (PA), Precision (PRE), Recall (REC), False Positive Rate (FPR), False Negative Rate (FNR). The time is reported for a forward pass through the network. The results demonstrate that all three networks outperform all the previous state-of-the-art approaches. Fast-Net does not only consistently outperform all previous methods in all metrics but also its runtime is only a millisecond slower than the fastest architecture, however largely outperforming such architecture.

We also tested giving more parameters to the fusion layers of the M-Net architecture. We aim to measure the impact of a wider search space for fusion and corresponding increment of the upsampling filters to our approach. To this end, we increased the fusion and up-convolution layers to 256 filters; we call such modification M-Net heavy. The results show that having more parameters in the fusion layers can make the network provide the highest IOU, 2.56 percentage points higher than Fast-Net. The heavy version of M-Net also displays the highest precision and recall values and the lowest false positive and negative rates. The only encountered limitation is the higher forward pass time.

H. 4.8 Real-World Experiments

In this section we present experimental results on actual robots. The collected datasets are able to benchmark three main challenges of semantic segmentation, which are scale, occlusion and crowded environments.

1) Range Experiments:: We explicitly made a dataset to measure robustness of our networks when exposed to multiple scales. PASCAL parts and KITTI does not provide any specific data for measuring range robustness. The robot

TABLE XVI: Runtime depending on the GPU, resolution of 300×300.

GPU	Forward Pass Time (ms)
TK1	1440
TX1	599
GTX 680	97.4
K-40	108
GTX TITAN	96.8
GTX 970	66.4
GTX 980	**51**
GTX TITAN X	**52.2**
GTX 1070	**50.3**

TABLE XVII: Runtime depending on the resolution for a TITAN X GPU.

Resolution	Forward Pass Time (ms)
150×150	30
200×200	35.6
300×300	52.2
500×500	83

(a) Fast-Net

(b) DNN [28]

(c) CNN1 (anonymous)

(d) CNN (anonymous)

Fig. 8: Segmentation results for road segmentation extracted from the KITTI benchmark. The proposed architecture shows low occurrences of false positive predictions and sharp segmentation of edges. Green corresponds to correct segmentation, red to false negative and blue to false positive detections.

TABLE XV: Results on UM lane KITTI dataset.

Method	MaxF	AP	PRE	REC	FPR	FNR	Time
Fast-Net	**89.88%**	87.52%	92.01%	87.84%	1.34%	12.16%	83ms
ANM (anonymous)	89.11%	81.11%	88.68%	89.54%	2.01%	10.46%	60ms
PCA-Lane-S (anonymous)	87.01%	74.16%	87.31%	86.70%	2.22%	13.30%	**30ms**
S (anonymous)	85.15%	76.52%	88.61%	81.95%	1.85%	18.05%	100ms

Fig. 9: Lane predictions on the KITTI dataset compared with the 3 next best approaches. Top Left: Fast-Net, Top Right: ANN, Botton Left: PCA-Lane-S, Botton Right: S. In the images green is the correct segmentation, red false negative detection and blue false positive detection.

used for the experiments is the Obelix robot, Figure 10a. Obelix is a robot designed for autonomous navigation in pedestrian environments thus is useful to mimic the human perspective for perception tasks.

In our experiments, we captured images from a Bumblebee stereo camera. The testing phase consisted of training all the architectures using the PASCAL parts dataset [6] and testing on the range data. The dataset comprises outdoor images of two different people at distances ranging from 0.8 m to 6.0 m, captured every 20 cm, as shown at Figure 11. Figure

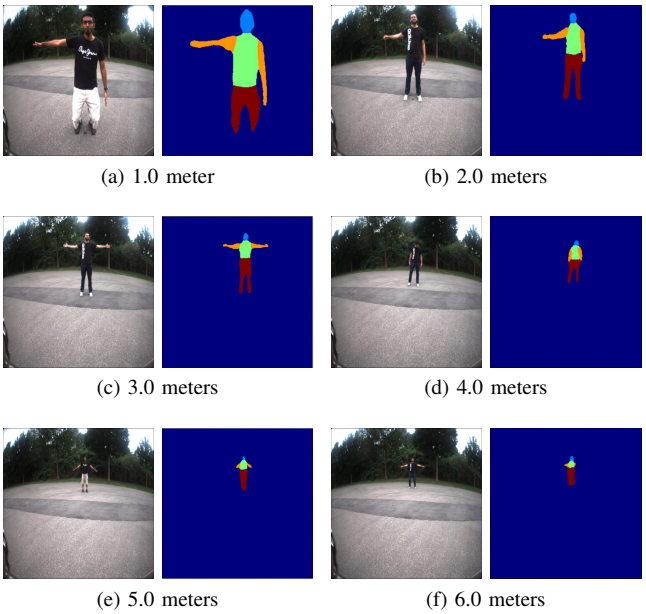
TABLE XVIII: Performance of Part-Net, Fast-Net and M-Net in comparison to the state-of-the-art, from coarse PASCAL PARTS.

Baseline	IOU	PA	PRE	REC	FPR	FNR	Time
FCN [23]	57.35	71.79	77.28	67.92	15.08	27.12	$150ms$
SegNet [3]	45.22	44.82	49.88	80.71	45.21	9.57	**47.7ms**
ParseNet [22]	64.25	70.02	74.66	78.95	19.86	15.89	$88ms$
Part-Net	78.23	85.47	86.00	87.78	11.50	10.10	$225ms$
Fast-Net	81.92	88.81	88.74	90.04	9.51	8.44	**48.7ms**
M-Net	78.15	84.95	86.29	87.60	11.69	10.15	$130ms$
M-Net (Heavy)	**84.62**	**91.51**	**91.47**	**90.57**	**7.1**	**8.29**	$345ms$

(a) Obelix ground robot.

(b) AR.DRONE 2.0 aerial platform.

Fig. 10: Robotics platforms used in our tests.



(a) 1.0 meter (b) 2.0 meters

(c) 3.0 meters (d) 4.0 meters

(e) 5.0 meters (f) 6.0 meters

Fig. 11: Qualitative results of the range experiment with the Obelix robot. The lower resolution at one point does not allow detection of small body parts anymore. However, the larger parts, such as the torso and head, are still detected correctly even at 6m distance.

12 presents our quantitative results for Part-Net and Fast-Net. Fast-Net consistently performs better than Part-Net, specially for longer distances. The smaller filters (3×3) at the last two layers of the contraction side of the network provide a smaller field of view and present a gain for longer distances. For distances beyond 4 meters, Fast-Net largely outperforms Part-Net.

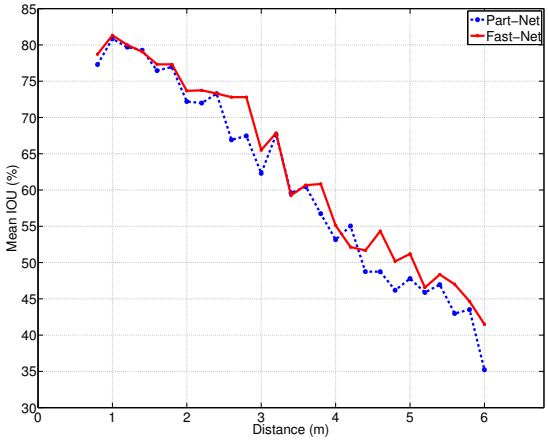


Fig. 12: Range segmentation results. The proposed approach consistently performs better than Part-Net, specially for longer distances.

2) Freiburg City Dataset:: We collected data with the Obelix robot in the Freiburg city center. The dataset is constituted of 52574 images and presents scenes with occlusion, radial distortion, motion blur, glare, low light conditions, people at multiple scales and crowds. The effort to build such dataset is to bring semantic segmentation to a highly realistic robotics scenario.

The dataset is used as a testing set, Part-Net was used for these experiments, trained on the PASCAL dataset with 4 body parts. We choose Part-Net based on its robustness to segmenting unseen environments, as confirmed in the Freiburg People in Disaster experiments. Figure 13 shows examples of segmentation under motion-blur, glare, occlusions and multiple instances. In the first row we present results with a single instance in a sequential order. The second row of Figure 13 depicts segmentation examples in presence of crowds, heavy occlusion and radial distortion. The first

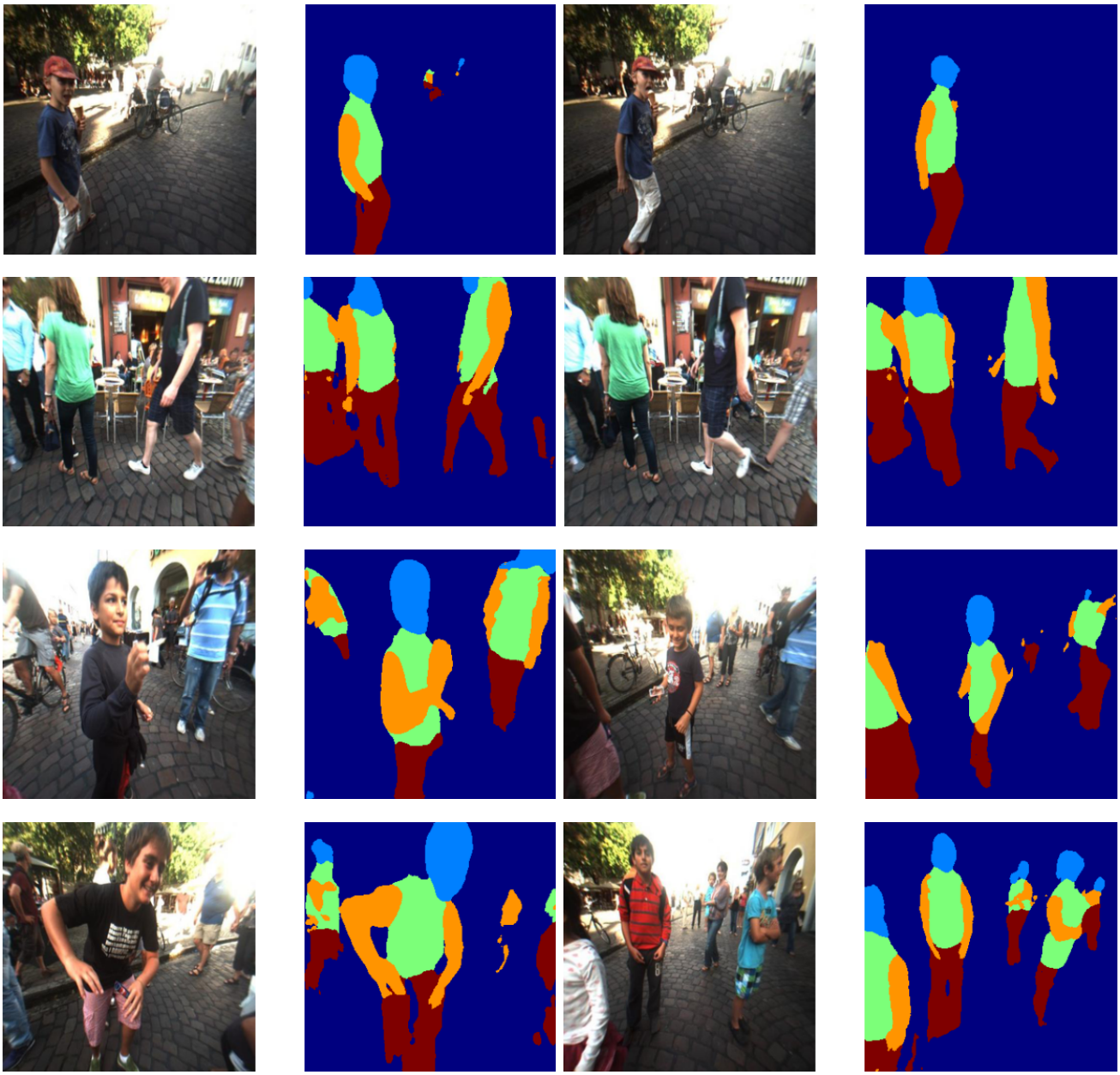
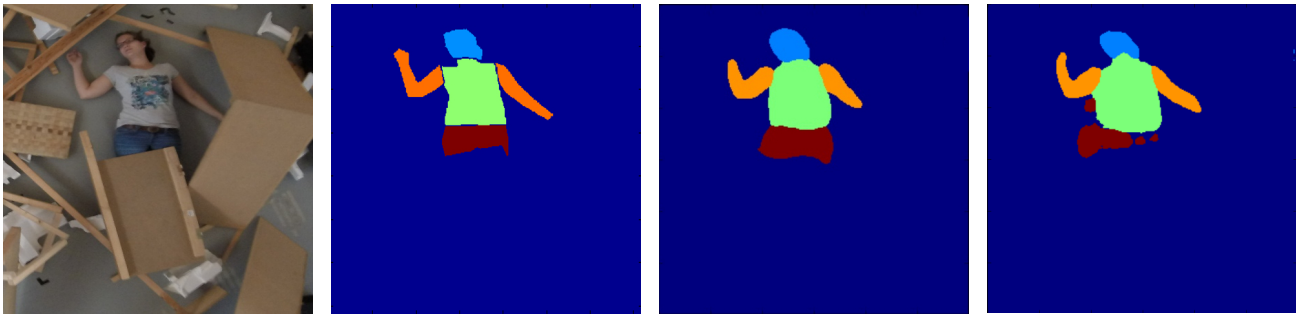


Fig. 13: Segmentation examples on multiple scenarios on Freiburg City Dataset. The first row presents segmentation of a single instance in a sequence. Second row exemplifies an example of segmentation for multiple instances in a sequence. The third and fourth rows show segmentation results under glare, occlusion and multiple instances presence.

example in the third row of Figure 13 shows segmentation under glare and the second example segmentation under occlusion, where the segmented person on the left has most of his body occluded, specially the head. The row displays an unusual pose for segmentation and another example of severe occlusion, where the person has only half of his body visible.

The main outcome from such experiment is that Part-Net can robustly transfer its representation from a controlled dataset, in our case PASCAL Parts, to a real world environment and still produce accurate segmentation masks. Part-Net can segment body parts under heavy occlusion and been robust to unusual light conditions. Such difficult conditions are not present in the PASCAL Parts dataset.

3) Freiburg People in Disaster:: For testing the robustness of our networks to occlusion and clutter in other robotics scenario we created the Freiburg People in Disaster dataset. We used an AR.DRONE 2.0 aerial platform(Figure 10b). The platform lacks a high definition downward facing camera, therefore we mounted a GoPro HERO 4 to collect the Freiburg People in Disaster dataset. The dataset consists of images and corresponding segmentation masks for a set of 4 people in an environment that mimics a disaster scenario, including heavy clutter and occlusions. Figure 14 shows an example



(a) Input Image (b) Groundtruth (c) Part-Net (d) Fast-Net

Fig. 14: Predictions of our networks for the Freiburg People in Disaster Dataset.

with the ground truth and the results obtained using Part-Net and Fast-Net approaches.

Since the dataset is too small to split into training and a test set, we only used it for testing networks trained on PASCAL Parts with 4 body parts. Table XIX presents the results for this dataset. Part-Net performs 28% better than the state of the art FCN, Fast-Net is slightly worse than Part-Net but four times faster. The results obtained with Part-Net and Fast-Net are already good enough for use in practice.

TABLE XIX: Results for Freiburg People in Disaster dataset.

Method	Head	Torso	Arms	Legs	IOU
FCN [23]	52.71	62.49	35.04	43.25	43.20
Fast-Net	78.02	78.21	63.90	64.70	70.73
Part-Net	**80.56**	**79.45**	**63.93**	**64.91**	**71.99**

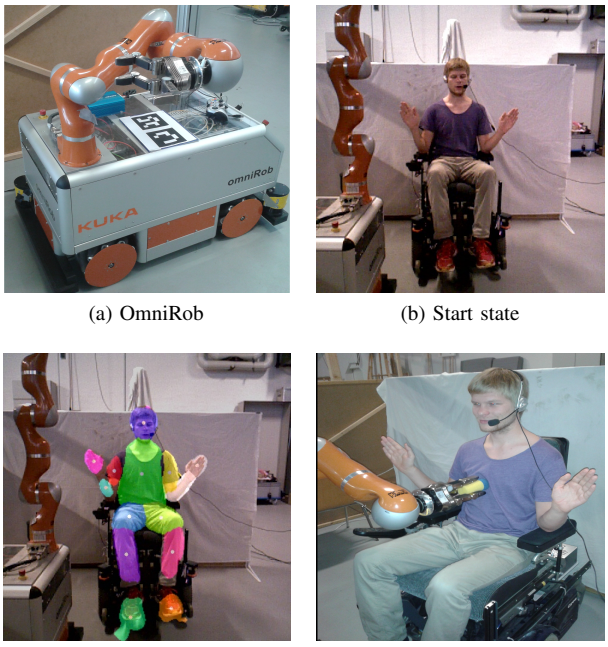
I. 4.9 Robotic Interaction Experiment:

In this section we present the use of the human part segmentation with a robot scratching a paralysed person. Our experimental setup consists of a KUKA OmniRob equipped with a 7-DOF light-weight arm, Figure 15(a).

Our experiment consists of a person in a wheel chair requesting a specific part of his body to be scratched. The person determines the spot which needs to be scratched via speech. For this we use an open source continuous, speaker-independent speech recognition: pocketsphinx [14].

We implement pocketsphinx with a simple grammar, which is activated by the keyword start. Pocketsphinx then converts the speech into simple orders like "scratch my left hand". For the other body parts there are speech commands which are defined similarly. So whenever pocketsphinx recognizes a speech sequence which fits to its grammar, it sends the specific command to our central Robot Operating System (ROS) interface.

Our ROS interface receives as input also a stream of images from a camera. This camera takes images of the person in the wheel chair throughout the whole experiment. These images are given into Part-Net. The Part-Net predicts an outcome based on the input image. This output is sent back to the ROS interface. From every body part the mean position of the segmented pixels is computed and together with the speech command the destination of the robot arm is determined. Then the ROS interface directs the robot arm to reach its destination and, in case of success, to scratch the person. Figure 16 illustrates this process.



(a) OmniRob (b) Start state

(c) Segmentation output (d) Final state

Fig. 15: Scratching experiment - Process: (a) shows the omniRob platform used for these experiments (b) illustrates the start state of our experiment. The robot is in its home position. (c) shows the output of our segmentation tool. In (d) the robot arm has reached the goal (in this scenario the torso) and scratches the person.

We ran a total number of 40 attempts on four different test persons. One attempt is when the person gives an order to scratch. Each of the 14 body parts has been at least once the part which was requested to be scratched.

We distinguish four different outcomes of the experiment: (1) speech recognition fails, (2) segmentation fails, (3) robot arm movement not possible and (4) success. The first three

TABLE XX: Results for the scratching experiment. (1) speech recognition fails, (2) segmentation fails, (3) robot arm movement not possible, (4) success.

	(1)	**(2)**	**(3)**	**(4)**
number	10	5	13	12

categories reflect which part of the experiment fails, while the last category is the amount of sucessful attempts. Table XX shows that our scratching experiment is successful in 30% of all cases. Considering its variable tasks and the restricted working space of the robot arm this is a good result. Furthermore, we conclude that the Part-Net works well in combination with robots as it is responsible for only 18% of the failures.

Figure 17 shows examples of the segmentation in the scratching experiment. In the first row high accurate segmentations for the different body parts are presented while the second row depicts failed segmentations. In the last two rows segmentation results that include a direct interaction with the robot arm are shown. We observe that the body parts which are not easy to segment (e.g. legs in the second row), are absorbed by the robot arm. It can also be seen that the segmentation during the experiment is much harder.

V. 5. Conclusion and Future Works

We presented three different deep learning architectures which are focused on efficiency and robustness aspects. Part-Net presents state of the art performance for human body part segmentation and robustness to occlusion and clutter, however cannot provide interactive frames rates for robotics tasks. Fast-Net provides not only state of the art results on multiple segmentation tasks but also makes efficient use of resources available on embedded platforms and provides near-real time performance on modern GPUs. M-Net extends up-convolutional networks to multiresolution fusion and presents results with increased robustness. To the best of our knowledge the presented architectures are the first to tackle efficiency and robustness aspects of semantic segmentation using up-convolutional networks in real world scenarios. In addition, we shown results on human part segmentation and road and lane segmentation problems. Our approach advances the state of the art in all the following datasets, namely PASCAL Parts, Freiburg Sitting People, Freiburg People in Disaster and Freiburg Range. We also presented competitive results for KITTI-Road and KITTI-Lane datasets.This work also introduced the Freiburg City Dataset to provide realistic robotics scenarios for human body part segmentation and a full robotic system designed to measure the behavior of our human body part segmentation approach in a human-robot interaction task. This experiment confirms the applicability and robustness of our technique on a mobile robot scenario.

Future work will investigate new architectures to perform human part segmentation in videos taking the temporal context into account. The ability to do human part segmentation from videos can have many robotics applications, like learning from demonstration. Another possible line of research will include instance segmentation. A robust and efficient instance segmentation method could provide further perception capabilities to new applications, for instance people segmentation and tracking in a smart home scenario.

References

[1] Pulkit Agrawal, Joao Carreira, and Jitendra Malik. Learning to see by moving. In *ICCV*, 2015.

[2] Jose M. Alvarez, Theo Gevers, Yann LeCun, and Antonio M. Lopez. Road scene segmentation from a single image. In *Proceedings of the 12th European Conference on Computer Vision - Volume Part VII*, ECCV'12, pages 376–389, Berlin, Heidelberg, 2012.

[3] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *arXiv preprint arXiv: 1511.00561*, 2015.

[4] Xavier Boix, Josep M. Gonfaus, Joost van de Weijer, Andrew D. Bagdanov, Joan Serrat, and Jordi Gonzalez. Harmony potentials. *IJCV*, pages 83–102, 2012.

[5] Clemens-Alexander Brust, Sven Sickert, Marcel Simon, Erik Rodner, and Joachim Denzler. Convolutional patch networks with spatial prior for road detection and urban scene understanding. In *International Conference on Computer Vision Theory and Applications (VISAPP)*, 2015.

[6] Xianjie Chen, Roozbeh Mottaghi, Xiaobai Liu, Sanja Fidler, Raquel Urtasun, and Alan Yuille. Detect what you can: Detecting and representing objects using holistic models and body parts. In *CVPR*, 2014.

[7] A. Dosovitskiy, P. Fischer, E. Ilg, P. Hausser, C. Hazırba, V. Golkov, P. v.d. Smagt, D. Cremers, and T. Brox. Flownet: Learning optical flow with convolutional networks. In *ICCV*, Dec 2015.

[8] Alexey Dosovitskiy, Jost Tobias Springenberg, and Thomas Brox. Learning to generate chairs with convolutional neural networks. In *CVPR*, 2015.

[9] David Eigen, Dilip Krishnan, and Rob Fergus. Restoring an image taken through a window covered with dirt or rain. In *ICCV*, pages 633–640, 2013.

[10] Jannik Fritsch, Tobias Kuehnl, and Andreas Geiger. A new performance measure and evaluation benchmark for road detection algorithms. In *International Conference on Intelligent Transportation Systems (ITSC)*, 2013.

[11] Ross Girshick. Fast R-CNN. *arXiv preprint arXiv:1504.08083*, 2015.

[12] Ross Girshick. Faster R-CNN: Towards real-time object detection with region proposal networks. *NIPS*, 2015.

[13] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014.

[14] David Huggins-Daines, Mohit Kumar, Arthur Chan, Alan W Black, Mosur Ravishankar, Alex Rudnicky, et al. Pocketsphinx: A free, real-time continuous speech recognition system for hand-held devices. In *Acoustics, Speech and Signal Processing, 2006. ICASSP 2006 Proceedings. 2006 IEEE International Conference on*, volume 1, pages I–I. IEEE, 2006.

[15] Satoshi Iizuka, Edgar Simo-Serra, and Hiroshi Ishikawa. Let there be color!: Joint end-to-end learning of global and local image priors for automatic image colorization with simultaneous classification. *ACM Transactions on Graphics (Proc. of SIGGRAPH 2016)*, 35(4), 2016.

[16] Ajrun Jain, Jonathan Tompson, Mykhaylo Andriluka, Graham W. Taylor, and Christoph Bregler. Learning human pose estimation features with convolutional networks. In *ICLR*, 2014.

[17] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv:1408.5093*, 2014.

[18] Alex Kendall, Matthew Grimes, and Roberto Cipolla. Posenet: A convolutional network for real-time 6-dof camera relocalization. In *ICCV*, 2015.

[19] K. Konda and R. Memisevic. Learning visual odometry with a convolutional network. In *VISAPP*, 2015.

[20] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, pages 1097–1105, 2012.

[21] Fayao Liu, Chunhua Shen, Guosheng Lin, and Ian Reid. Learning depth from single monocular images using deep convolutional neural fields. *IEEE T. Pattern Analysis and Machine Intelligence*, 2015.

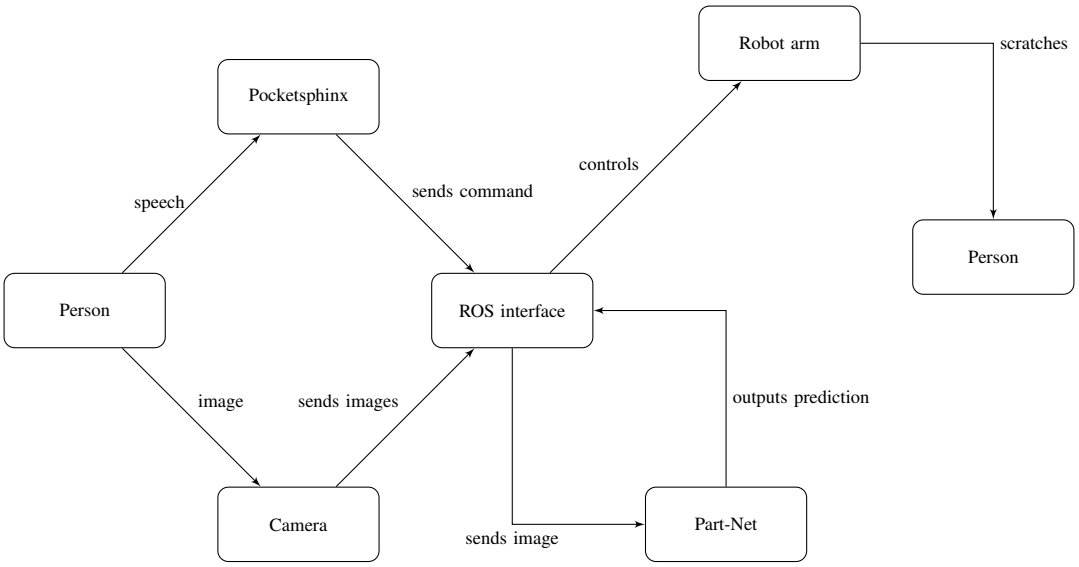


Fig. 16: Overview of our scratching system.

[22] Wei Liu, Andrew Rabinovich, and Alexander C. Berg. Parsenet: Looking wider to see better. *arXiv preprint arXiv: 1506.04579*, 2015.

[23] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. *CVPR*, nov 2015.

[24] Wenhao Lu, Xiaochen Lian, and Alan Yuille. Parsing semantic parts of cars using graphical models and segment appearance consistency. In *BMVC*, 2014.

[25] Michael Maire, Stella X. Yu, and Pietro Perona. Object detection and segmentation from joint embedding of parts and pixels. In *ICCV*, 2011.

[26] Ofer Matan, Christopher J.C. Burges, Yann Le Cun, and John S. Denker. Multi-digit recognition using a space displacement neural network. In *Neural Information Processing Systems*, pages 488–495. Morgan Kaufmann, 1992.

[27] Caio César Teodoro Mendes, Vincent Frémont, and Denis Fernando Wolf. Exploiting fully convolutional neural networks for fast road detection. In *ICRA*, 2016.

[28] Rahul Mohan. Deep deconvolutional networks for scene parsing. *CoRR*, abs/1411.4101, 2014.

[29] Feng Ning, Damien Delhomme, Yann Lecun, Fabio Piano, LÃl'on Bottou, and Paolo Emilio Barbano. Toward automatic phenotyping of developing embryos from videos. *IEEE Transactions on Image Processing*, 14:1360–1371, 2005.

[30] G. L. Oliveira, Abhinav Valada, Claas Bollen, Wolfram Burgard, and Thomas Brox. Deep learning for human part discovery in images. *IEEE International Conference on Robotics and Automation (ICRA)*, 2016.

[31] P. H. O. Pinheiro and R. Collobert. Recurrent convolutional neural networks for scene labeling. In *Proceedings of the 31st International Conference on Machine Learning (ICML)*, 2014.

[32] Nils Plath, Marc Toussaint, and Shinichi Nakajima. Multi-class image segmentation using conditional random fields and global classification. In *ICML*, pages 817–824, New York, NY, USA, 2009. ACM.

[33] O. Ronneberger, P.Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, volume 9351 of *LNCS*, pages 234–241. Springer, 2015.

[34] Connor Schenck and Dieter Fox. Detection and tracking of liquids with fully convolutional networks. *CoRR*, abs/1606.06266, 2016.

[35] Pierre Sermanet, David Eigen, Xiang Zhang, Michael Mathieu, Rob Fergus, and Yann Lecun. Overfeat: Integrated recognition, localization and detection using convolutional networks. In *ICLR*, 2014.

[36] Marcel Simon, Erik Rodner, and Joachim Denzler. Part detector discovery in deep convolutional neural networks. In *ACCV*, volume 2, pages 162–177, 2014.

[37] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *ICLR*, 2015.

[38] Niko Suenderhauf, Sareh Shirazi, Adam Jacobson, Feras Dayoub, Edward Pepperell, Ben Upcroft, and Michael Milford. Place recognition with convnet landmarks: Viewpoint-robust, condition-robust, training-free. In *Proceedings of Robotics: Science and Systems*, July 2015.

[39] Jonathan Tompson, Ross Goroshin, Arjun Jain, Yann LeCun, and Christoph Bregler. Efficient object localization using convolutional networks. *CVPR*, 2015.

[40] Stavros Tsogkas, Iasonas Kokkinos, George Papandreou, and Andrea Vedaldi. Semantic part segmentation with deep learning. *CoRR*, abs/1505.02438, 2015.

[41] Ralph Wolf and John C. Platt. Postal address block location using a convolutional locator network. In *in Advances in Neural Information Processing Systems 6*, pages 745–752. Morgan Kaufmann Publishers, 1994.

[42] Saining Xie and Zhuowen Tu. Holistically-nested edge detection. In *ICCV*, 2015.

[43] Ning Zhang, Jeff Donahue, Ross Girshick, and Trevor Darrell. Part-based R-CNNs for fine-grained category detection. In *ECCV*, 2014.

[44] Ning Zhang, Manohar Paluri, Marc'Aurelio Ranzato, Trevor Darrell, and Lubomir D. Bourdev. PANDA: pose aligned networks for deep attribute modeling. In *CVPR*, pages 1637–1644, 2014.

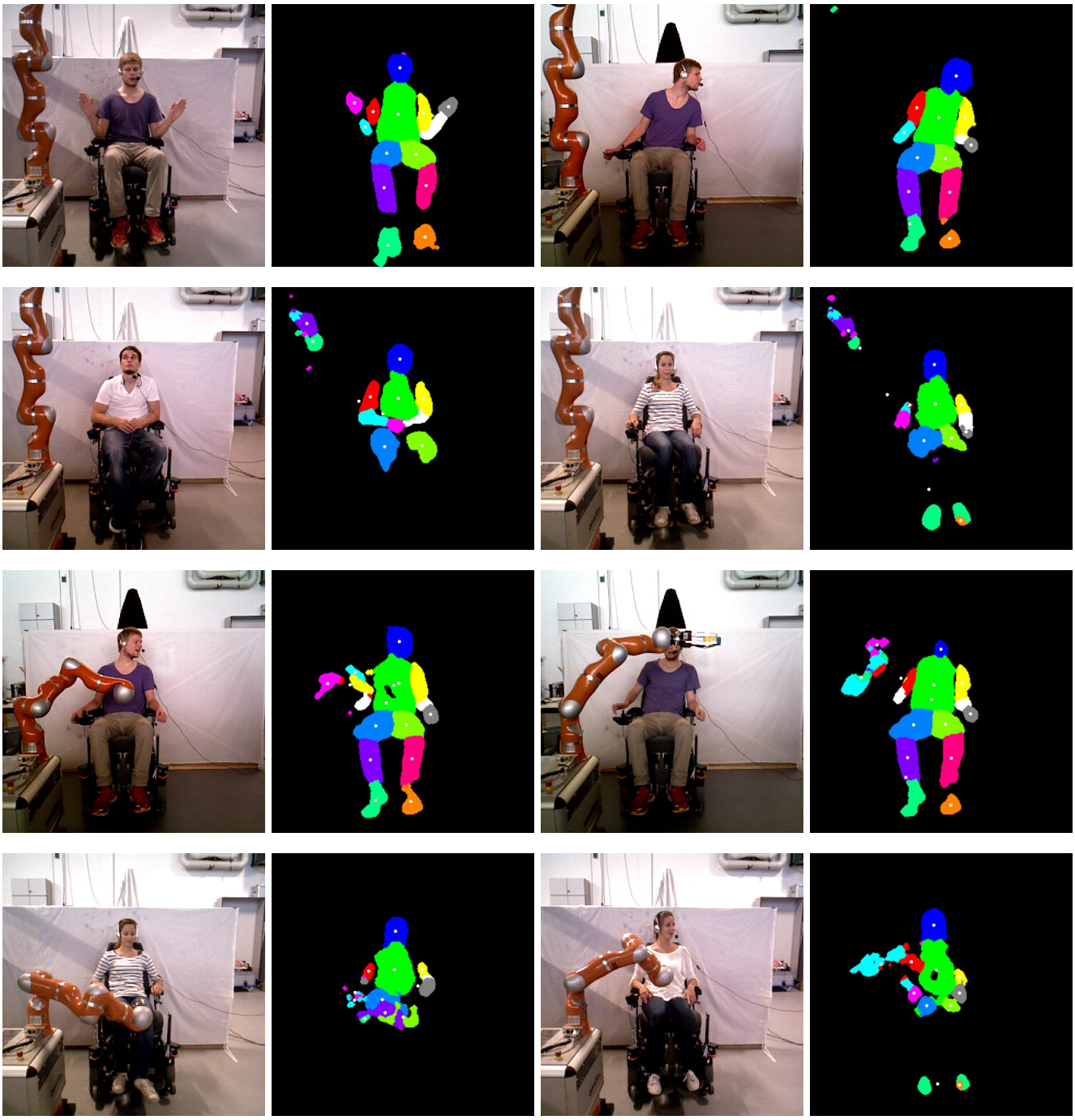


Fig. 17: Segmentation results for the scratching experiment. The white dots describe the computed means of the different body parts. The first row shows examples in which the segmentation approach performs accurately. In the second row it can be seen that the robot arm is also segmented, lowering the robustnes of Part-Net for this scenario. The last two rows present examples during the scratching experiment. While the results in the third row are less accurate than in the first row, they still produce reliable segmentation masks, whereas the last row, which presents failed examples.