2021 IEEE International Conference on Robotics and Automation (ICRA 2021)
May 31 - June 4, 2021, Xi'an, China

Efficient RGB-D Semantic Segmentation for Indoor Scene Analysis

Daniel Seichter, Mona Köhler, Benjamin Lewandowski, Tim Wengefeld and Horst-Michael Gross

2021 IEEE International Conference on Robotics and Automation (ICRA) | 978-1-7281-9077-8/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICRA48506.2021.9561675

***Abstract*— Analyzing scenes thoroughly is crucial for mobile robots acting in different environments. Semantic segmentation can enhance various subsequent tasks, such as (semantically assisted) person perception, (semantic) free space detection, (semantic) mapping, and (semantic) navigation. In this paper, we propose an efficient and robust RGB-D segmentation approach that can be optimized to a high degree using NVIDIA TensorRT and, thus, is well suited as a common initial processing step in a complex system for scene analysis on mobile robots. We show that RGB-D segmentation is superior to processing RGB images solely and that it can still be performed in real time if the network architecture is carefully designed. We evaluate our proposed Efficient Scene Analysis Network (ESANet) on the common indoor datasets NYUv2 and SUNRGB-D and show that we reach state-of-the-art performance while enabling faster inference. Furthermore, our evaluation on the outdoor dataset Cityscapes shows that our approach is suitable for other areas of application as well. Finally, instead of presenting benchmark results only, we also show qualitative results in one of our indoor application scenarios.**

I. Introduction

Semantic scene perception and understanding is essential for mobile robots acting in various environments. In our research projects, covering public environments from supermarkets [1], [2] to hospitals [3], [4] and domestic applications [5], [6], our robots need to perform several tasks in parallel such as obstacle avoidance, semantic mapping, navigation to semantic entities, and person perception. Most of the tasks require to be handled in real time given limited computing and battery capabilities. Hence, an efficient and shared initial processing step can facilitate subsequent tasks. Semantic segmentation is well suited for such an initial step, as it provides precise pixel-wise information that can be used for numerous subsequent tasks.

In this paper, we propose an efficient and robust encoder-decoder-based semantic segmentation approach that can be embedded in complex systems for semantic scene analysis such as shown in Fig. 1. The segmentation output enriches the robot's visual perception and facilitates subsequent processing steps by providing individual semantic masks. For our person perception [7], computations can be restricted to image regions segmented as person, instead of processing the entire image. Furthermore, the floor class indicates free space that can be used for inpainting invalid depth pixels as well as serves as additional information for avoiding even small obstacles below the laser. For mapping [8], we can include semantics and ignore image regions segmented as dynamic classes such as person.

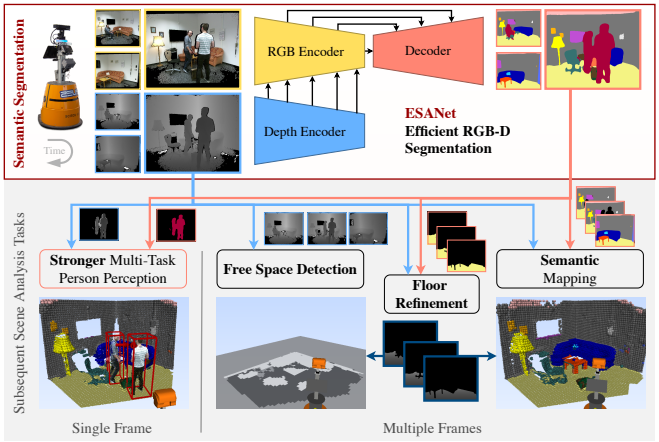


Fig. 1: Our proposed efficient RGB-D segmentation approach can serve as a common preprocessing step for subsequent tasks such as person perception, free space detection to avoid even small obstacles below the laser, or semantic mapping.

Our segmentation approach relies on both RGB and depth images as input. Especially in indoor environments, cluttered scenes may impede semantic segmentation. Incorporating depth images can alleviate this effect by providing complementary geometric information, as shown in [9], [10], [11]. In contrast to processing RGB images solely, this design decision comes with some additional computational cost. However, in this paper, we show that two carefully designed shallow encoder branches (one for RGB and one for depth data) can achieve better segmentation performance while still enabling faster inference (application of network) than a single deep encoder branch for RGB images solely. Moreover, our Efficient Scene Analysis Network (ESANet) enables much faster inference than most other RGB-D segmentation methods, while performing on par or even better as shown by our experiments.

We evaluate our ESANet on the commonly used indoor datasets NYUv2 [12] and SUNRGB-D [13] and further present qualitative results in our indoor application. Instead of only focusing on mean intersection over union (mIoU) as evaluation metric for benchmarking, we also strive for fast inference on embedded hardware. However, rather than reporting the inference time on high-end GPUs, we measure inference time on our robot's NVIDIA Jetson AGX Xavier. We designed our network architecture such that it

Authors are with Neuroinformatics and Cognitive Robotics Lab, Technische Universität Ilmenau, 98693 Ilmenau, Germany. daniel.seichter@tu-ilmenau.de

This work has received funding from the German Federal Ministry of Education and Research (BMBF) to the project MORPHIA (grant agreement no. 16SV8426) and from the Carl Zeiss Foundation to the project E4SM (grant agreement no. P2017-01-005).

can be executed as a single optimized graph using NVIDIA TensorRT. Moreover, by evaluating on the outdoor dataset Cityscapes [14], we show that our approach can also be applied to other areas of application.

The main contributions of this paper are

- an efficient RGB-D segmentation approach, which can serve as initial processing step to facilitate subsequent scene analysis tasks and is characterized by:
  - a carefully designed architecture that can be optimized to a high degree using NVIDIA TensorRT and, thus, enables fast inference
  - an efficient ResNet-based encoder that utilizes a modified basic block that is computationally less expensive while achieving higher accuracy
  - a decoder that utilizes a novel learned upsampling
- a detailed ablation study to the fundamental parts of our approach and their impact on segmentation performance and inference time
- qualitative results in a complex system for robotic scene analysis proving the applicability and robustness.

Our code as well as the trained networks are publicly available at: `https://github.com/TUI-NICR/ESANet`

II. Related Work

Common network architectures for semantic segmentation follow an encoder-decoder design. The encoder extracts semantically rich features from the input and performs downsampling to reduce computational effort. The decoder restores the input resolution and, finally, assigns a semantic class to each input pixel.

A. RGB-D Semantic Segmentation

Depth images provide complementary geometric information to RGB images and, thus, improve segmentation [9], [10]. However, incorporating depth information into RGB segmentation architectures is challenging as depth introduces deviating statistics and characteristics from another modality.

In [15], depth information is used to project the RGB images into a 3D space. However, processing the resulting 3D data, leads to significantly higher computational complexity. [16], [17], [18], [19], [20] design specifically tailored convolutions, taking into account depth information. Nevertheless, these modified convolutions often lack optimized implementations and, thus, are slow and not applicable for real-time segmentation on embedded hardware.

The majority of approaches for RGB-D segmentation [9], [21], [10], [11], [22], [23], [24], [25] simply use two branches, one for RGB and one for depth data and fuse the feature representations later in the network. This way, each branch can focus on extracting modality-specific features, such as color and texture from RGB images and geometric, illumination-independent features from depth images. Fusing these modality-specific features leads to stronger feature representations. Instead of fusing only low-level or high-level features, [9] shows that the segmentation performance increases if the features are fused at multiple stages. Typically, the features are fused once at each resolution stage with the last fusion at the end of both encoders. Using only one decoder for the combined features reduces the computational effort. FuseNet [9] and RedNet [10] fuse the depth features into the RGB encoder, which follows the intuition that the semantically richer RGB features can be further enhanced using complementary depth information. SA-Gate [22] combines RGB and depth features and fuses the recalibrated features back into both encoders. In order to make the two encoders independent of each other, ACNet [11] uses an additional, virtual, third encoder that obtains modality-specific features from the two encoders and processes the combined features. Instead of fusing in the encoder, the modality-specific features can also be used to refine the features in the common decoder via skip connections as in RDFNet [23], SSMA [24] and MMAF-Net [25].

However, none of the aforementioned methods focus on efficient RGB-D segmentation for embedded hardware. Using deep encoders such as ResNets with 50, 101 or even 152 layers results in high inference times and, therefore, makes them inappropriate for deploying to mobile robots.

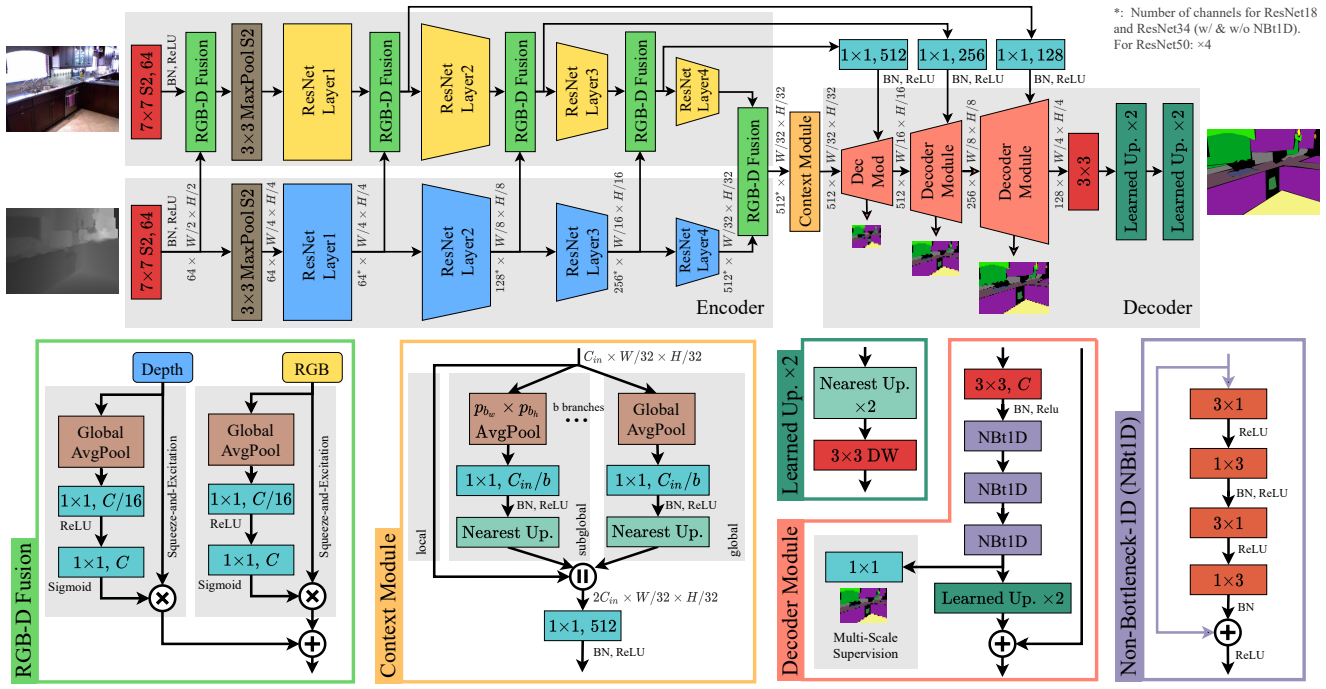
B. Efficient Semantic Segmentation

In contrast to RGB-D segmentation, recent RGB approaches [26], [27], [28], [29], [30], [31], [32] also address reducing computational complexity to enable real-time segmentation. Most efficient segmentation approaches propose specifically tailored network architectures, which reduce both the number of operations and parameters to enable faster inference while still retaining good segmentation performance. Approaches, such as ERFNet [26], LEDNet [27], or DABNet [28] introduce efficient encoder blocks by replacing expensive 3×3 convolutions with more light-weight variants such as factorized, grouped, or depth-wise separable convolutions. Nevertheless, although requiring more operations and memory, SwiftNet [30] and BiSeNet [31] are still faster than many other methods, while achieving higher segmentation performance by simply using a pretrained ResNet18 as encoder. This can be deduced to utilizing early and high downsampling in the encoder, a light-weight decoder, and using standard 3×3 convolutions, which are currently implemented more efficiently than grouped or depth-wise convolutions and have large representational power.

Following SwiftNet and BiSeNet, our approach also uses a ResNet-based encoder. However, in order to further reduce inference time, we exchange the basic block in all ResNet layers with a more efficient block, based on factorized convolutions.

III. Efficient RGB-D Segmentation

The architecture of our Efficient Scene Analysis Network (ESANet) for RGB-D semantic segmentation is depicted in Fig. 2 (top). It is inspired by the RGB segmentation approach SwiftNet [30], i.e., a shallow encoder with a pretrained ResNet18 backbone and large downsampling, a context module similar to the one in PSPNet [33], a shallow decoder with skip connections from the encoder, and final upsampling by a factor of 4. However, SwiftNet does not



Legend: $\boxed{k_w \times k_h, C}$: Convolution with kernel size $k_w \times k_h$ and C output channels, S2: Stride 2, BN: Batch Normalization, Up.: Upsampling, DW: Depthwise, Ⓘ: Concatenation

Fig. 2: Overview of our proposed ESANet for efficient RGB-D segmentation (top) and specific network parts (bottom).

incorporate depth information at all. Therefore, our ESANet uses an additional encoder for depth data. This depth encoder extracts complementary geometric information that is fused into the RGB encoder at several stages using an attention mechanism. Furthermore, both encoders use a revised architecture enabling faster inference. The decoder is comprised of multiple modules, each is upsampling the resulting feature maps by a factor of 2 and is refining the features using convolutions as well as by incorporating encoder features. Finally, the decoder maps the features to the classes and rescales the class mapping to the input resolution.

Our entire network features simple components implemented in PyTorch [34]. We do not use complex structures or specifically tailored operations as these are often incompatible for converting to ONNX [35] or NVIDIA TensorRT and, thus, result in slower inference time.

In the following, we explain each part of our network design in detail as well as its motivation. Fig. 2 (bottom) depicts the exact structure of our network modules.

A. Encoder

The RGB and depth encoder both use a ResNet architecture [36] as backbone. For efficiency reasons, we do not replace strided convolutions by dilated convolutions as in PSPNet [33] or DeepLabv3 [37]. Thus, the resulting feature maps at the end of the encoder are 32 times smaller than the input image. For a trade-off between speed and accuracy, we use ResNet34 but also show results for ResNet18 and ResNet50. We replace the basic block in each layer of ResNet18 and ResNet34 with a spatially factorized version. More precisely, each 3×3 convolution is replaced by a 3×1 and a 1×3 convolution with a ReLU in-between. The so-called *Non-Bottleneck-1D-Block* (NBt1D) is depicted in Fig. 2 (violet) and was initially proposed in ERFNet [26] for another network architecture. In our experiments, we show that this block can also be used in ResNet and simultaneously reduces inference time and increases segmentation performance.

B. RGB-D Fusion

At each of the five resolution stages in the encoders (see Fig. 2), depth features are fused into the RGB encoder. The features from both modalities are first reweighted with a Squeeze and Excitation (SE) module [38] and then summed element-wisely, as shown in Fig. 2 (light green). Using this channel attention mechanism, the model can learn which features of which modality to focus on and which to suppress, depending on the given input. In our experiments, we show that this fusion mechanism notably improves segmentation.

C. Context Module

Due to the limited receptive field of ResNet [33], we additionally incorporate context information by aggregating features at different scales using several branches in a context module similar to the Pyramid Pooling Module in PSPNet [33] (see Fig. 2 orange). Since NVIDIA TensorRT only supports pooling with fixed sizes, we carefully designed the context module such that the pooling sizes are always a factor of the input resolution of the context module and no adaptive pooling is required. Note that, depending on the image resolution of the respective dataset, the number of existing factors and, thus, the branches b and pooling sizes $p_{b_w} \times p_{b_h}$ differ. Our experiments show that this additional context module improves segmentation.

D. Decoder

As shown in Fig 2, our decoder is comprised of three decoder modules (depicted in red in Fig 2). Our decoder module extends the one of SwiftNet [30], which is comprised of a 3×3 convolution with a fixed number of 128 channels and a subsequent bilinear upsampling. However, our experiments show that for indoor RGB-D segmentation a more complex decoder is required. Therefore, we use 512 channels in the first decoder module and decrease the number of channels in each 3×3 convolution as the resolution increases. Moreover, we incorporate three additional Non-Bottleneck-1D-blocks to further increase segmentation performance.

Finally, we upsample the feature maps by a factor of 2. We do not use transposed convolutions for upsampling as they are computationally expensive and often introduce undesired gridding artifacts to the final segmentation, as shown in Fig. 3 (right). Moreover, instead of using bilinear interpolation, we propose a novel light-weight learned upsampling method (see Fig. 2 dark green), which achieves better segmentation results: In particular, we first use nearest neighbor upsampling to enlarge the resolution. Afterwards, a 3×3 depthwise convolution is applied to combine adjacent features. We initialize the kernels such that the whole learned upsampling initially mimics bilinear interpolation. However, our network is able to adapt the weights during training and, thus, can learn how to combine adjacent features in a more useful manner, which improves segmentation performance.

Although being upscaled, the resulting feature maps still lack fine-grained details that were lost during downsampling in the encoders. Therefore, we design skip connections from encoder to decoder stages of the same resolution. To be precise, we take the fused RGB-D encoder feature maps, project them with a 1×1 convolution to the same number of channels used in the decoder, and add them to the decoder feature maps. Incorporating these skip connections results in more detailed semantic segmentations.

Similar to [30], [39], we only process feature maps in the decoder until they are $4\times$ smaller than the input images and use a 3×3 convolution to map the features to the classes of the respective dataset. Two final learned upsampling modules restore the resolution of the input image.

Instead of calculating the training loss only at the final output scale, we add supervision to each decoder module. At each scale, a 1×1 convolution computes a segmentation of a smaller scale, which is supervised by the down-scaled ground truth segmentation.

IV. Experiments

We evaluate our approach on two commonly used RGB-D indoor datasets, namely SUNRGB-D [13] and NYUv2 [12] and present an ablation study to essential parts of our network. In order to demonstrate that our approach is suitable for other areas of application as well, we also show results on the Cityscapes [14] dataset, the most widely used outdoor dataset for semantic segmentation. Finally, instead of reporting benchmark results only, we present qualitative results when using our approach in a robotic indoor application.

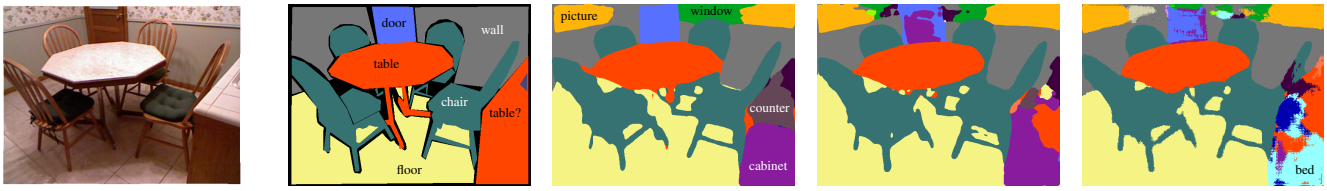
A. Implementation Details & Datasets

We trained our networks using PyTorch [34] for 500 epochs with batches of size 8. For optimization, we used both SGD with momentum of 0.9 and Adam [40] with learning rates of $\{0.00125, 0.0025, 0.005, 0.01, 0.02, 0.04\}$ and $\{0.0001, 0.0004\}$, respectively, and a small weight decay of 0.0001. We adapted the learning rate using PyTorch's one-cycle learning rate scheduler. To further increase the number of training samples, we augmented the images using random scaling, cropping, and flipping. For RGB images, we also applied slight color jittering in HSV space.

The best models were chosen based on the mean intersection over union (mIoU). We used bilinear upsampling to rescale the resulting class mapping to the size of the ground truth segmentation before computing the argmax for the final segmentation mask.

NYUv2 & SUNRGB-D: NYUv2 contains 1,449 indoor RGB-D images, of which 795 are used for training and 654 for testing. We used the common 40-class label setting. SUNRGB-D has 37 classes and consists of 10,335 indoor RGB-D images, including all images of NYUv2. There are 5,285 training and 5,050 testing images. Our ablation study is based on NYUv2 as it is smaller and, thus, leads to faster trainings. However, according to [41], training on a subset is sufficient for a reliable model selection. For both datasets, we used a network input resolution of 640×480 and applied median frequency class balancing [42]. As the input to the context module has a resolution of 20×15 due to the downsampling of 32, we used $b = 2$ branches, one with global average pooling and one with a pooling size of 4×3.

Cityscapes: This dataset contains 5,000 images with fine-grained annotation for 19 classes. The images have a high resolution of 2048×1024. There are 2,975 images for training, 500 for validation, and 1,525 for testing. Cityscapes also provides 20k coarsely annotated images, which *we did not use for training*. We computed corresponding depth images from the disparity images. Since we set the network input



RGB image | Ground Truth | **Learned Up. (ours)** | Bilinear Up. | Transp. Conv. (ACNet [11])

Fig. 3: Qualitative comparison of upsampling methods on NYUv2 test set (same colors as in Fig. 1 and Fig. 6).

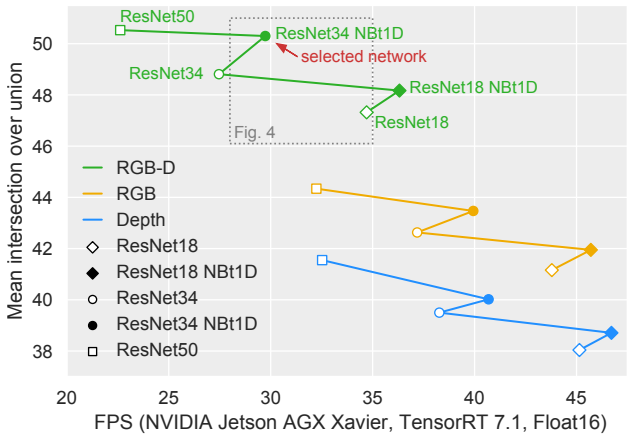


Fig. 4: Comparison of RGB-D to RGB and depth networks (single encoder) and different backbones on NYUv2 test set.

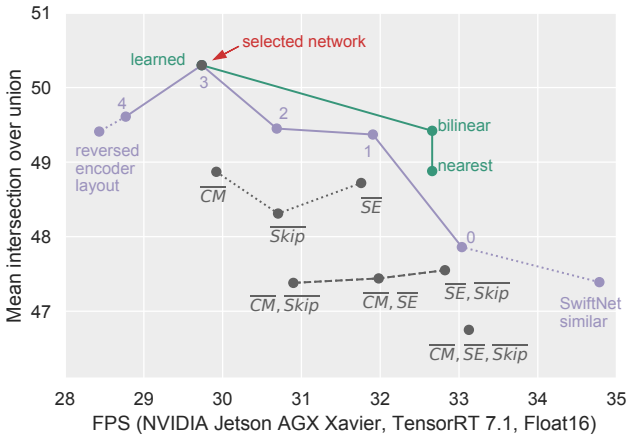


Fig. 5: Ablation Study on NYUv2 test. Each color indicates modifying one aspect: purple: number of NBt1D blocks in decoder module, dark green: upsampling method, and gray: usage of specific network parts with $\overline{CM}$: no context module, $\overline{Skip}$: no encoder-decoder skip connections, and $\overline{SE}$: no Squeeze-and-Excitation before fusing RGB and depth.

resolution to 1024×512, the input to our context module has a resolution of 32×16, which allows $b = 4$ branches in the context module, one with global average pooling and the others with pooling sizes of 16×8, 8×4, and 4×2.

For further details and other hyperparameters, we refer to our implementation available on GitHub.

B. Results on NYUv2 & SUNRGB-D

Fig. 4 compares our RGB-D approach on NYUv2 to single-modality baselines for RGB and depth (single encoder) as well as evaluates different encoder backbones. As expected, neither processing depth data nor RGB data alone reach the segmentation performance of our proposed RGB-D network. Remarkably, the shallow ResNet18-based RGB-D network performs better than the much deeper ResNet50-based RGB network while still being faster. Moreover, replacing ResNet's basic block with Non-Bottleneck-1D (NBt1D) block can further improve both segmentation and inference time. Note that ResNet50 incorporates bottleneck blocks, which cannot be replaced the same way.

Tab. I lists the results of our RGB-D approach for both indoor datasets. For the larger SUNRGB-D dataset, a similar trend can be observed. Compared to the state of the art, our smaller ESANet achieves similar segmentation results as the often much deeper networks. Besides focusing on segmentation performance alone, we also strive for low inference time on the embedded hardware of our robots. Therefore, we measured the inference time for all available approaches on a NVIDIA Jetson AGX Xavier using NVIDIA TensorRT. For our carefully designed ESANet, NVIDIA TensorRT enables up to $5\times$ faster inference compared to PyTorch. As shown in Tab. I (last column), our approach enables much faster inference while performing on par or even better than other approaches. For our application, we choose ESANet with ResNet34 backbone and Non-Bottleneck-1D (NBt1D) block (printed in bold in Tab. I) as it offers the best trade-off between inference time and performance. The last row in Tab. I further indicates that additional pretraining on synthetic data, such as SceneNet [43], should be preferred to deeper backbones, especially if the target dataset is small.

C. Ablation Study on NYUv2

Fig. 5 shows the ablation study for fundamental parts of our network architecture and justifies our design choices. Furthermore, it indicates the impact of each part when it is necessary to adapt our selected network to deviating real-time requirements.

As shown in purple, a shallow decoder similar to SwiftNet [30] is not as good as more complex decoders. Therefore, we gradually increased the number of additional NBt1D blocks in the decoder module. Apparently, a fixed number of three blocks in each decoder module performs better than a different number or a reversed layout of the encoder's design.

In dark green, different upsampling methods in the decoder are displayed. Although increasing inference time, the learned upsampling improves mIoU by 0.9. Moreover, as shown in Fig. 3, the obtained segmentation contains more fine-grained details compared to using bilinear interpolation. It further prevents gridding artifacts introduced by transposed convolutions as used in ACNet [11] or RedNet [10].

As shown in gray in Fig. 5, a context module, encoder-decoder skip connections, as well as reweighting modality-specific features with Squeeze-and-Excitation before fusion, independently improve segmentation performance. Incorporating all three network parts leads to the best result.

D. Results on Cityscapes

To demonstrate that our approach is applicable to other areas such as outdoor environments as well, in Tab. II, we further present an evaluation on the Cityscapes dataset.

We first focus on the smaller resolution of 1024×512 as it is commonly used for efficient segmentation. Moreover, since most approaches rely on RGB as input solely, we start by comparing a single-modality RGB version of our approach. Efficient approaches with custom architectures such as ERFNet [26], LEDNet [27], and ESPNetv2 [32] are quite

Method	Backbone	NYUv2	SUN-RGB-D	FPS
FuseNet [9]	2× VGG16	-	37.29	†
RedNet [10]	2× R34	-	46.8	26.0
SSMA [24]	2× mod. R50	-	44.43	12.4
MMAF-Net [25]	2× R50	-	45.5	N/A
RedNet [10]	2× R50	-	47.8	22.1
RDFNet [23]	2× R50	47.7*	-	7.2
ACNet [11]	3× R50	48.3	48.1	16.5
SA-Gate [22]	2× R50	50.4	49.4*	11.9
SGNet [19]	R101	49.0	47.1	N/A ▽
Idempotent [21]	2× R101	49.9	47.6	N/A ▽
2.5D Conv [16]	R101	48.5	48.2	N/A ▽
MMAF-Net [25]	2× R152	44.8	47.0	N/A ▽
RDFNet [23]	2× R152	50.1*	47.7*	5.8
ESANet-R18	2× R18	47.32	46.24	34.7
ESANet-R18-NBt1D	2× R18 NBt1D	48.17	46.85	36.3
ESANet-R34	2× R34	48.81	47.08	27.5
ESANet-R34-NBt1D	2× R34 NBt1D	50.30	48.17	29.7
ESANet-R50	2× R50	50.53	48.31	22.6
ESANet (pre. SceneNet)	2× R34 NBt1D	51.58	48.04	29.7

TABLE I: Mean intersection over union of our ESANet compared to state-of-the-art methods on NYUv2 and SUNRGB-D test set ordered by SUNRGB-D performance and backbone complexity. FPS is reported for NVIDIA Jetson AGX Xavier (Jetpack 4.4, TensorRT 7.1, Float16). Legend: R: ResNet, *: additional test-time augmentation, i.e., flipping or multi-scale (not timed), N/A: no implementation available, †: includes operations, which are not supported by TensorRT, and ▽: expected to be slower due to complex backbone.

	Method	1024×512 Val	1024×512 Test◇	1024×512 FPS	2048×1024 Val	2048×1024 Test◇	2048×1024 FPS
RGB	ERFNet [26]	-	69.7	49.9	-	-	-
	LEDNet [27]	-	70.6*	38.5	-	-	-
	ESPNetv2 [32]	66.4	66.2	47.4	-	-	-
	SwiftNet [30]	70.2	-	64.5	75.4	75.5	20.8
	BiSeNet [31]	-	-	-	74.8	74.7	20.0
	PSPNet [33]	-	-	-	-	81.2*	1.8
	DeepLabv3 [37]	-	-	-	79.3	81.3*	0.9
	ESANet-R18-NBt1D	71.48	-	37.2	77.95	-	9.8
	ESANet-R34-NBt1D	72.70	72.87	32.3	78.47	77.56	8.3
	ESANet-R50	73.88	-	24.9	79.23	-	6.5
RGB-D	SSMA [24]	-	-	-	82.19*	82.31*	2.2
	SA-Gate [22]	-	-	-	81.7	82.8	2.1
	LDFNet [44]	68.48	71.3	25.3	-	-	-
	ESANet-R18-NBt1D	74.65	-	28.9	79.25	-	7.6
	ESANet-R34-NBt1D	75.22	75.65	23.4	80.09	78.42	6.2
	ESANet-R50	75.66	-	16.9	79.97	-	4.0

TABLE II: Mean intersection over union of our ESANet on Cityscapes for both common input resolutions compared to state-of-the-art methods. FPS is reported for NVIDIA Jetson AGX Xavier (Jetpack 4.4, TensorRT 7.1, Float16). Legend: ◇: test server result, *: trained with additional coarse data.

fast but also perform notably worse than our ESANet. Compared to ERFNet, LEDNet, and ESPNetv2, SwiftNet [30] is both faster and achieves higher mIoU. Nevertheless, with an input resolution of 1024×512, our ESANet-R34-NBt1D still exceed 30 FPS while outperforming all other efficient approaches by at least 2.2 mIoU. Incorporating depth further increases segmentation performance. However, the performance gain is not as high as for the indoor dataset NYUv2. We assume that this can be deduced to the fact that the disparity images of Cityscapes are not as precise as the indoor depth images of NYUv2 and SUNRGB-D. Compared to the RGB-D approach LDFNet [44] with similar inference time, we achieve notably higher mIoU.

For completeness, we also evaluated our networks on the full resolution of 2048×1024. Compared to other methods, our ESANet lies in between mobile (SwiftNet, BiSeNet) and non-mobile approaches for both mIoU and inference time.

However, compared to SwiftNet (RGB, 2048×1024), our ESANet-R34-NBt1D achieves similar segmentation performance and slightly faster inference while processing RGB-D inputs with the smaller input resolution of 1024×512.

E. Application on our Robots

Instead of evaluating on benchmark datasets only, we further present qualitative results with a Kinect2 sensor [45], [46] in one of our indoor applications. We deployed our proposed ESANet-R34-NBt1D to our robot in order to accomplish the complex system for semantic scene analysis shown in Fig. 1. The obtained segmentation masks enrich the robot's visual perception enabling stronger person perception and robust semantic mapping including a refined floor representation which indicates free space. Fig. 6 provides an insight into the entire system. For further qualitative results and a comparison to non-semantic scene perception, we refer to the attached video or our repository on GitHub.

V. Conclusion

In this paper, we have presented an efficient RGB-D segmentation approach, called ESANet, which is characterized by two enhanced ResNet-based encoders utilizing the Non-Bottleneck-1D block, an attention-based fusion for incorporating depth information, and a decoder utilizing a novel learned upsampling. On the indoor datasets NYUv2 and SUNRGB-D, our ESANet performs on par or even better while enabling much faster inference compared to other state-of-the-art methods. Thus, it is well suited for embedding in a complex system for scene analysis on mobile robots given limited hardware.

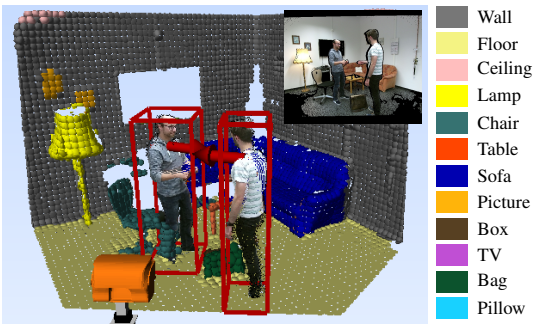


Fig. 6: Application in our robotic scene analysis system.

REFERENCES

[1] H.-M. Gross, *et al.*, "TOOMAS: Interactie shopping guide robots in everyday use – final implementation and experiences from long-term field trials," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*. IEEE, 2009, pp. 2005–2012.

[2] B. Lewandowski, *et al.*, "Socially compliant human-robot interaction for autonomous scanning tasks in supermarket environments," in *IEEE Int. Symp. on Robot and Human Interactive Communication (RO-MAN)*. IEEE, 2020, pp. 363–370.

[3] H.-M. Gross, *et al.*, "Mobile robot companion for walking training of stroke patients in clinical post-stroke rehabilitation," in *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2017, pp. 1028–1035.

[4] T. Q. Trinh, *et al.*, "Autonomous mobile gait training robot for orthopedic rehabilitation in a clinical environment*," in *IEEE Int. Conf. on Robot and Human Interactive Communication (RO-MAN)*, 2020, pp. 580–587.

[5] H.-M. Gross, *et al.*, "Robot companion for domestic health assistance: Implementation, test and case study under everyday conditions in private apartments," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*. IEEE, 2015, pp. 5992–5999.

[6] H. M. Gross, *et al.*, "Living with a mobile companion robot in your own apartment - final implementation and results of a 20-weeks field study with 20 seniors," in *IEEE Int. Conf. on Robotics and Automation (ICRA), Montreal, Canada*. IEEE, 2019, pp. 2253–2259.

[7] D. Seichter, *et al.*, "Multi-task deep learning for depth-based person perception in mobile robotics," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 10 497–10 504.

[8] E. Einhorn and H.-M. Gross, "Generic 2D/3D SLAM with NDT maps for lifelong application," in *Europ. Conf. on Mobile Robots (ECMR)*, 2013.

[9] C. Hazirbas, *et al.*, "FuseNet: Incorporating Depth into Semantic Segmentation via Fusion-based CNN Architecture," in *Asian Conference on Computer Vision (ACCV)*, 2016, pp. 213–228.

[10] J. Jiang, *et al.*, "RedNet: Residual Encoder-Decoder Network for indoor RGB-D Semantic Segmentation," *arXiv preprint arXiv:1806.01054*, 2018.

[11] X. Hu, *et al.*, "ACNet: Attention Based Network to Exploit Complementary Features for RGBD Semantic Segmentation," *IEEE Int. Conf. on Image Processing (ICIP)*, 2019.

[12] N. Silberman, *et al.*, "Indoor Segmentation and Support Inference from RGBD Images," in *Europ. Conf. on Computer Vision (ECCV)*, 2012.

[13] S. Song, *et al.*, "SUN RGB-D: A RGB-D Scene Understanding Benchmark Suite," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 567–576.

[14] M. Cordts, *et al.*, "The Cityscapes Dataset for Semantic Urban Scene Understanding," *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 3213–3223, 2016.

[15] Y. Zhong, *et al.*, "3D Geometry-Aware Semantic Labeling of Outdoor Street Scenes," in *Int. Conf. on Pattern Recognition (ICPR)*, 2018, pp. 2343–2349.

[16] Y. Xing, *et al.*, "2.5D Convolution for RGB-D Semantic Segmentation," in *IEEE Int. Conf. on Image Processing (ICIP)*, 2019, pp. 1410–1414.

[17] Y. Xing, *et al.*, "Malleable 2.5D Convolution: Learning Receptive Fields along the Depth-axis for RGB-D Scene Parsing," in *Europ. Conf. on Computer Vision (ECCV)*, 2020, pp. 1–17.

[18] W. Wang and U. Neumann, "Depth-Aware CNN for RGB-D Segmentation," in *Europ. Conf. on Computer Vision (ECCV)*, 2018, pp. 144–161.

[19] L.-Z. Chen, *et al.*, "Spatial Information Guided Convolution for Real-Time RGBD Semantic Segmentation," *arXiv preprint arXiv:2004.04534*, pp. 1–11, 2020.

[20] Y. Chen, *et al.*, "3D Neighborhood Convolution: Learning Depth-Aware Features for RGB-D and RGB Semantic Segmentation," in *Int. Conf. on 3D Vision (3DV)*, 2019, pp. 173–182.

[21] Y. Xing, *et al.*, "Coupling Two-Stream RGB-D Semantic Segmentation Network by Idempotent Mappings," in *IEEE Int. Conf. on Image Processing (ICIP)*, 2019, pp. 1850–1854.

[22] X. Chen, *et al.*, "Bi-directional Cross-Modality Feature Propagation with Separation-and-Aggregation Gate for RGB-D Semantic Segmentation," in *Europ. Conf. on Computer Vision (ECCV)*, 2020, pp. 561–577.

[23] S. Lee, *et al.*, "RDFNet: RGB-D Multi-level Residual Feature Fusion for Indoor Semantic Segmentation," *Int. Conference on Computer Vision (ICCV)*, pp. 4990–4999, 2017.

[24] A. Valada, *et al.*, "Self-supervised model adaptation for multimodal semantic segmentation," *Int. Journal of Computer Vision (IJCV)*, 2019.

[25] F. Fooladgar and S. Kasaei, "Multi-Modal Attention-based Fusion Model for Semantic Segmentation of RGB-Depth Images," *arXiv preprint arXiv:1912.11691*, pp. 1–12, 2019.

[26] E. Romera, *et al.*, "ERFNet: Efficient Residual Factorized ConvNet for Real-Time Semantic Segmentation," *IEEE Transactions on Intelligent Transportation Systems (ITS)*, pp. 263–272, 2018.

[27] Y. Wang, *et al.*, "LEDnet: A Lightweight Encoder-Decoder Network for Real-Time Semantic Segmentation," in *IEEE Int. Conference on Image Processing (ICIP)*, 2019, pp. 1860–1864.

[28] G. Li, *et al.*, "DABNet: Depth-wise Asymmetric Bottleneck for Real-time Semantic Segmentation," *British Machine Vision Conference (BMVC)*, 2019.

[29] S.-Y. Lo, *et al.*, "Efficient dense modules of asymmetric convolution for real-time semantic segmentation," in *ACM Int. Conf. on Multimedia in Asia*, 2019, pp. 1–6.

[30] M. Oršić, *et al.*, "In Defense of Pre-trained ImageNet Architectures for Real-time Semantic Segmentation of Road-driving Images," *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 12 607–12 616, 2019.

[31] C. Yu, *et al.*, "BiSeNet: Bilateral segmentation network for real-time semantic segmentation," in *Europ. Conf. on Computer Vision (ECCV)*, 2018, pp. 325–341.

[32] S. Mehta, *et al.*, "ESPNetv2: A Light-weight, Power Efficient, and General Purpose Convolutional Neural Network," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 9190–9200.

[33] H. Zhao, *et al.*, "Pyramid scene parsing network," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 2881–2890.

[34] A. Paszke, *et al.*, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems (NIPS)*. Curran Associates, Inc., 2019, pp. 8024–8035.

[35] J. Bai, *et al.*, "Onnx: Open neural network exchange," https://github.com/onnx/onnx, 2019.

[36] K. He, *et al.*, "Deep residual learning for image recognition," *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.

[37] L.-C. Chen, *et al.*, "Rethinking Atrous Convolution for Semantic Image Segmentation," *arXiv preprint arXiv:1706.05587*, 2017.

[38] J. Hu, *et al.*, "Squeeze-and-excitation networks," in *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 7132–7141.

[39] L.-C. Chen, *et al.*, "Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation," in *Europ. Conf. on Computer Vision (ECCV)*, 2018, pp. 801–818.

[40] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," in *Int. Conf. Learning Representation (ICLR)*, 2015.

[41] J. Bornschein, *et al.*, "Small Data, Big Decisions: Model Selection in the Small-Data Regime," in *Int. Conf. on Machine Learning (ICML)*, 2020.

[42] D. Eigen and R. Fergus, "Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture," *Int. Conf. on Computer Vision (ICCV)*, pp. 2650–2658, 2015.

[43] J. McCormac, *et al.*, "SceneNet RGB-D: Can 5M Synthetic Images Beat Generic ImageNet Pre-training on Indoor Segmentation?" *Int. Conf. on Computer Vision (ICCV)*, pp. 2697–2706, 2017.

[44] S.-W. Hung, *et al.*, "Incorporating Luminance, Depth and Color Information by a Fusion-Based Network for Semantic Segmentation," in *IEEE Int. Conf. on Image Processing (ICIP)*, 2019, pp. 2374–2378.

[45] Lingzhu Xiang, *et al.*, "Libfreenect2: Release 0.2," 2016. [Online]. Available: https://zenodo.org/record/50641

[46] F. J. Lawin, *et al.*, "Efficient multi-frequency phase unwrapping using kernel density estimation," in *Europ. Conf. on Computer Vision (ECCV)*, 2016, pp. 170–185.