2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
September 27 - October 1, 2021. Prague, Czech Republic

2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) | 978-1-6654-1714-3/21/$31.00 ©2021 IEEE | DOI: 10.1109/IROS51168.2021.9636518

# Real-Time Monocular Human Depth Estimation and Segmentation on Embedded Systems

Shan An[1], Fangru Zhou[2], Mei Yang[2], Haogang Zhu[1,*], Changhong Fu[3], and Konstantinos A. Tsintotas[4]

***Abstract*— Estimating a scene's depth to achieve collision avoidance against moving pedestrians is a crucial and fundamental problem in the robotic field. This paper proposes a novel, low complexity network architecture for fast and accurate human depth estimation and segmentation in indoor environments, aiming to applications for resource-constrained platforms (including battery-powered aerial, micro-aerial, and ground vehicles) with a monocular camera being the primary perception module. Following the encoder-decoder structure, the proposed framework consists of two branches, one for depth prediction and another for semantic segmentation. Moreover, network structure optimization is employed to improve its forward inference speed. Exhaustive experiments on three self-generated datasets prove our pipeline's capability to execute in real-time, achieving higher frame rates than contemporary state-of-the-art frameworks (114.6 frames per second on an NVIDIA Jetson Nano GPU with TensorRT) while maintaining comparable accuracy.**

## I. Introduction

Depth estimation of a scene has been studied for a long time in the computer vision field for various applications, such as augmented reality [1], scene reconstruction [2], and detection [3]. In the robotic community, it is used for different tasks, which are mainly related to obstacle avoidance, localization, and mapping [4], [5]. The ability of a robot to build a consistent map during its autonomous mission, widely known as Simultaneous Localization and Mapping (SLAM) [6], is strengthened when scale information is provided as robust visual odometry is generated [7]. Thus, depth-sensing is essential in any contemporary SLAM system [8], [9]. Commonly used sensors include LiDARs, binocular vision, etc., which are expensive and massive. However, in most resource-constrained platforms (e.g. a micro aerial vehicle), cameras have become the primary perception device due to their low cost and power consumption. As a result, in such cases, approaches that tackle the depth estimation task make use of a monocular camera.

Early studies were based on multi-scale features extracted from Convolutional Neural Networks (CNN) [10]. Firstly, they predicted coarse-scale depth information, and subsequently, they refined it through a fine-scaled network. Current pipelines are mostly based on deep learning methods. These are distinguished into three main categories, namely supervised, weakly-supervised, and unsupervised ones. These frameworks adopt the encoder-decoder network structure [11], [12], [13], which originates from Natural Language Processing (NLP). As the incoming camera stream arrives, the encoder extracts high-level, low-resolution features, while the decoder merges and upsamples them to produce the final high-resolution depth map. Despite their high performances, these techniques are known for their excess demand in computational resources due to their high complexity functionality [14], [15]. Having identified this drawback, researchers developed frameworks with reduced computational complexity for real-time applications on embedded platforms [16], [17], [18].

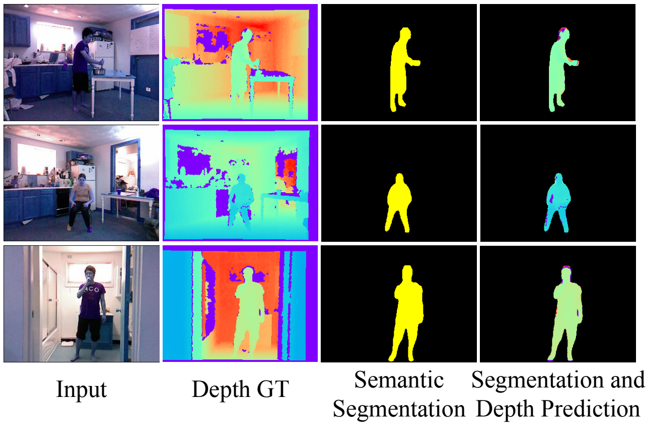


Fig. 1: Illustrative results of the proposed human depth estimation and segmentation framework. In the first column, the incoming RGB images are depicted. Since the encoder-decoder network structure consists of two branches, viz. depth prediction and semantic segmentation, the second column shows the metric depth data, while the third column presents the semantic segmentation results. The fourth column demonstrates the final segmentation and depth estimation of the foreground people instances.

In this paper, a straightforward network, which ensures real-time processing, for human depth estimation and segmentation is proposed. The former is an essential information for obstacle avoidance, while the latter can permit the system to achieve more complex task simultaneously as it provides

[1] Shan An and Haogang Zhu are with the School of Computer Science and Engineering, Beihang University, 100191 Beijing, China haogangzhu@buaa.edu.cn

[2] Fangru Zhou and Mei Yang are with Tech & Data Center, JD.COM Inc, 100108 Beijing, China zhoufangru@jd.com

[3] Changhong Fu is with the School of Mechanical Engineering, Tongji University, 201804 Shanghai, China changhongfu@tongji.edu.cn

[4] Konstantinos A. Tsintotas is with the Department of Production and Management Engineering, Democritus University of Thrace, 67132 Xanthi, Greece ktsintot@pme.duth.gr

[*] Corresponding Author

the crucial semantic data. Our pipeline utilizes MobileNetV1 [19] and Atrous Spatial Pyramid Pooling (ASPP) [20] as the encoder, while the decoder is composed of depthwise separable convolutions and upsampling modules. Furthermore, two branches, one for depth prediction and another for semantic segmentation, are proposed for the final estimation. A network structure optimization is employed as well as the TensorRT optimizer [21] to improve the forward inference speed. An example containing results produced by our network is illustrated in Fig. 1. Various approaches [11], [22], [12], [18], [23] have been developed on the NYU Depth v2 dataset [24], which is an indoor environment without humans, while the KITTI vision suite collection [25] is selected for the outdoor cases [8], [22], [13]. Thus, no suitable data-sequence was available for our method's evaluation. Therefore, to test the proposed framework, we generated three datasets based on the Cornell Activity [26] and the EPFL RGBD Pedestrian [27] image-sequences. Utilizing the provided depth information and through the well-known segmentation method MaskRCNN [28], we automatically predicted the people masks, which subsequently were used as ground truth for the segmentation branch. Finally, the proposed method is tested on these environments and compared against state-of-the-art approaches showing its improved performance. An implementation of the presented work is available, under the title "HDES-Net[1](Human Depth Estimation and Segmentation Network)".

The remainder of this work is structured as follows. A literature review is presented in Section II. In Section III, we describe our network design, whereas Section IV evaluates and discusses the experimental results. Finally, the last section is devoted to conclusions and future plans.

## II. Related Work

### A. Monocular depth estimation

Modern depth estimation methods use deep learning techniques trained over large-scale datasets. Following the popular encoder-decoder structure, the authors in [12] propose a network with multi-scale feature fusion and refinement to produce accurate object boundaries. A fast monocular depth estimation method is proposed by D. Wofk *et al.*, which utilizes MobileNet as the encoder and depthwise decomposition in the decoder [18]. This approach also utilizes the TVM compiler stack [29] intending to address the runtime inefficiencies, while NetAdapt [30] is adopted for post-training network pruning. Low complexity and low-latency are achieved, performing improved accuracy at 178 Frames Per Second (FPS) on an NVIDIA Jetson TX2 Graphics Processing Unit (GPU). Except for the per-pixel depth, a pipeline can infer a distribution over possible depths through discrete binary classifications [22]. A double refinement network uses iterative pixel shuffle for upsampling [31]. In this method, the authors aim to replace the traditional bilinear interpolation and propose to guide the intermediate depth branch using auxiliary losses. A geometric network is proposed in [13] to capture various structures of a scene, which is trained on uncalibrated videos.

### B. Semantic segmentation

Semantic segmentation refers to the process that labels each pixel of an image with a corresponding class of what is represented. Transferring and fine-tuning classification networks to Fully Convolutional Networks (FCN) [32] show that improved performance can be achieved without further machinery. Object interaction information is aggregated and fused to improve semantic segmentation performances [33]. Based on the common encoder-decoder architecture, U-Net [34] and SegNet [35] are widely used for semantic segmentation on medical [34] or satellite images [36]. The well-known framework DeepLab series proposed Atrous convolution for dense feature extraction [37], [38], ASPP [37] to encode objects, and a combination of CNN and fully-connected conditional random fields for accurate object boundary extraction [39]. DFANet [40] utilizes a lightweight backbone and multi-scale feature propagation to reduce parameters. This method exhibits sufficient performance and high inference speed. Similarly, LEDNet [41] proposes an asymmetric network for real-time semantic segmentation, while LiteSeg [42] explores ASPP to improve the segmentation results. An Efficient Spatial Pyramid (ESP) module is proposed by ESPNet [43], which uses a point-wise convolution and a pyramid of dilated convolutions to compose the final system. In a later work, the same authors proposed ESPNetv2 [44], where depthwise dilated separable convolutions are utilized to improve accuracy with fewer FLOPS.

## III. Proposed Method

In this section, we describe our network's design, which is a fully convolutional encoder-decoder network. The encoder extracts low-resolution, high-level abstract features from the provided visual sensory information, while through the decoder, a sufficiently high-resolution output is generated. An outline of the proposed pipeline is depicted in Fig. 2.

### A. The encoder

Commonly used networks, initially trained for image classification, such as VGG16 [45] and ResNet-50 [46], have shown their improved capability to extract features with high accuracy. As a result, they are usually selected as encoders. However, despite their high performances, these approaches have two major disadvantages. The first one is related to a large number of required computations, while the second one concerns the increased time of forward processing. Therefore, as we aim for a real-time application which is able to work on embedded platforms, they are incompatible with our system. MobileNetV1 [19] is selected as the backbone of the encoder to achieve a balanced ratio between accuracy and processing time. Furthermore, as the ASPP [20] module has different sizes of receptive fields, making it capable of extracting features at different scales, we add it after the backbone in our network. The depthwise separable convolutions are utilized to achieve lower execution times. We use

[1] https://github.com/AnshanTJU/HDES-Net

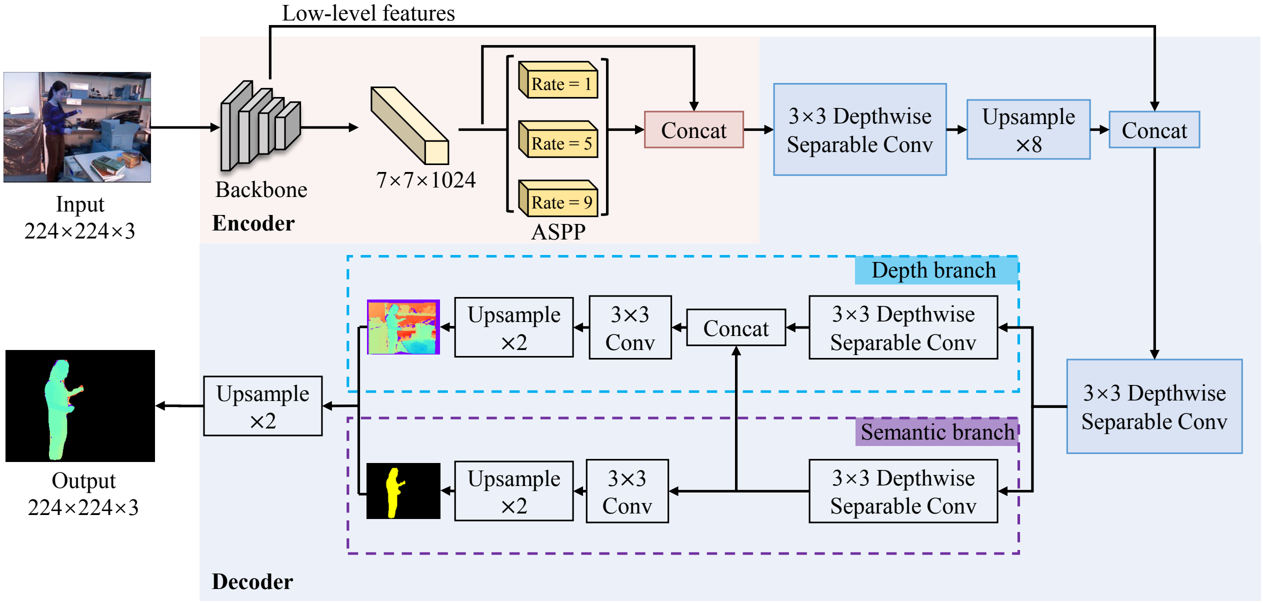


Fig. 2: An overview of the proposed pipeline. MobileNetV1 [19] is used as the network's backbone and combined with the Atrous Spatial Pyramid Pooling (ASPP) [20] module to form the encoder. This way, high-level features with abstract information are extracted. In contrast, the decoder fuses the high-level features with the low-level details in order to predict the final estimation results through the depth and semantic branches. Aiming to intuitively understand and analyze the feature resolution of each stage of the network, the input size is $224 \times 224$.

dilated convolutions [47] with a rate of $[1, 3, 6, 9]$ in ASPP to increase the receptive field while maintaining the feature's resolution.

### B. The decoder

As the encoder extracts features, the role of the decoder is to fuse and upsample them. The ones that come from ASPP are merged with a stride of $4$. This way, more detailed information is contained, which subsequently is used for depth estimation and semantic segmentation. Our decoder consists of two upsampling layers, which upsample the feature maps eight times and twice, respectively. Also, several depthwise separable layers perform $3 \times 3$ convolutions, reducing the number of output channels to $96$. For the prediction step, two branches are proposed, i.e., one for depth estimation and another for semantic segmentation. Both are composed of depthwise separable convolutions, standard convolutions, and upsampling layers. More specifically, the merged features are fed into the depthwise layer using a kernel size of $3$, stride length of $1$, and $96$ filters. The output of this layer is upsampled twice to obtain the image's depth result, the size of which is half of the input frame. Aiming to improve the performance, we add the features extracted from the depthwise separable convolutional layer of the semantic branch into the depth branch, as depicted in Fig. 2. Finally, a similar structure is adopted for the semantic segmentation pipeline where the output classes come from the standard convolution layer. The loss of the semantic segmentation branch is smooth $L1$ loss, while the depth estimation branch utilizes cross-entropy loss.

### C. Network structure optimization and acceleration

A network structure optimization strategy is performed over ASPP branches to improve the forward inference speed. Our main goal is to predict the depth and segment any human presenting in the input image within a distance of $10$ meters, which is the maximum depth of our datasets. Therefore, the scale is relatively fixed, excluding the cases where the target is too large, exceeding the frame's covers. Nevertheless, we retain the two parallel branches with dilation rates of $1$ and $9$, while we replace the remaining two with one presenting a dilation rate of $5$, as shown in Fig. 2. Moreover, the global average pooling is removed and replaced with a dilated convolution of rate $5$, which is more suitable for the human's scale in the image. In addition to the above strategy, we also use TensorRT SDK [21], a deep learning inference optimizer, for further acceleration.

## IV. Experimental Results

In this section, extensive experiments are conducted to demonstrate the effectiveness of the proposed architecture. At first, we introduce our benchmark, the training settings, and the evaluation metrics. Then, we provide ablation studies showing the effect of using the ASPP module, fusing the semantic information and network optimization. Finally, through quantitative and qualitative experimentation, we measure the method's overall performance.

### A. Experimental settings

*1) Benchmark introduction:* As there is no proper data-sequence related to simultaneous human depth estimation

TABLE I: Properties of the used datasets. A ratio of $9:1$ is selected in order to divide the training and test set.

| Dataset | | Description | Image resolution | # Training set | # Test set |
|---|---|---|---|---|---|
| Cornell Activity [26] | CAD-60 | Indoor, only one individual | $240 \times 320$ | 74575 | 5737 |
| | CAD-120 | | $480 \times 640$ | 60480 | 4653 |
| EPFL RGBD [27] | | Lab and corridor, multiple pedestrians | $424 \times 512$ | 4560 | 507 |

and segmentation in indoor scene, we generate three datasets based on the Cornell Activity [26] and EPFL RGBD [27]. The former is composed of CAD-60 and CAD-120 image-sequences, which both contain RGB-D visual information of humans performing activities. More specifically, CAD-60 has 60 videos involving 4 subjects with 12 activities on 5 different environments. Regarding CAD-120, it consists of 120 video of long daily activities involving 4 subjects, 10 high-level activities, 10 sub-activity, and 12 object affordance labels. The camera measurements are recorded via the Microsoft Kinect sensor. The EPFL RGBD Pedestrian dataset, which contains over 4000 RGB-D images, offers highly accurate depth maps thanks to a Kinect V2 module. Table I provides a brief description of each dataset used. We divide the data into the training set and the test set with a ratio of $9:1$. Subsequently, MaskRCNN [28] trained on COCO dataset [48] was utilized to generate semantic segmentation masks. We sample the annotations and verify them manually to ensure the correctness.

*2) Training:* For CAD-60 and CAD-120 we used the SGD optimizer with $10^{-4}$ weight decay and 0.9 momentum. The initial learning rate was set to $10^{-2}$ and decayed to one-tenth of the previous one, while performed every 60 epoch. Regarding the EPFL RGBD Pedestrian data-sequence, we adopted the Adam optimizer with $5 \times 10^{-4}$ as weight decay. The initial learning rate was set to $5 \times 10^{-4}$, while the decays to the half of the previous one as conducted every 100 epochs. The maximum number of iterations was 300 epochs. Network's implementation was made through the Pytorch framework [49] utilizing a batch size of 64, while an NVIDIA Tesla P40 GPU was used for the training procedure. During network's training and testing, the images were not resized. Source code and some demo videos of the presented work can be found at `https://github.com/AnshanTJU/HDES-Net`.

*3) Evaluation metrics:* Three metrics are selected for evaluating the overall performance. The RMSE (stands for Root Mean Squared Error in meters), $\delta_1$ (the percentage of predicted pixels where the relative error is within 25%), and the People IoU (Intersection over Union). The first two are chosen to evaluate the accuracy of human depth estimation, while the latter measures the semantic segmentation quality.

## *B. Ablation studies*

The ASPP module uses multiple dilated convolution branches with different rates to extract features at various scales. This way, it can provide a better representation of humans in the image. As shown in Table II, where we compare our network's performance under the ASPP inclusion, an improvement is observed in almost every metric.

TABLE II: The network's performance when Atrous Spatial Pyramid Pooling (ASPP) [20] is applied.

| Dataset | Metric | w/o ASPP | with ASPP |
|---|---|---|---|
| CAD-60 [26] | RMSE $\downarrow$ | 0.1529 | **0.1526** |
| | $\delta_1 \uparrow$ | 98.71% | **98.72%** |
| | People IoU $\uparrow$ | 96.80% | **96.82%** |
| CAD-120 [26] | RMSE $\uparrow$ | 0.3147 | **0.3140** |
| | $\delta_1 \uparrow$ | 99.97% | **97.98%** |
| | People IoU $\uparrow$ | 96.10% | **96.17%** |
| EPFL RGBD [27] | RMSE $\downarrow$ | 0.1484 | **0.1461** |
| | $\delta_1 \uparrow$ | 98.47% | **98.53%** |
| | People IoU $\uparrow$ | **96.08%** | 95.97% |

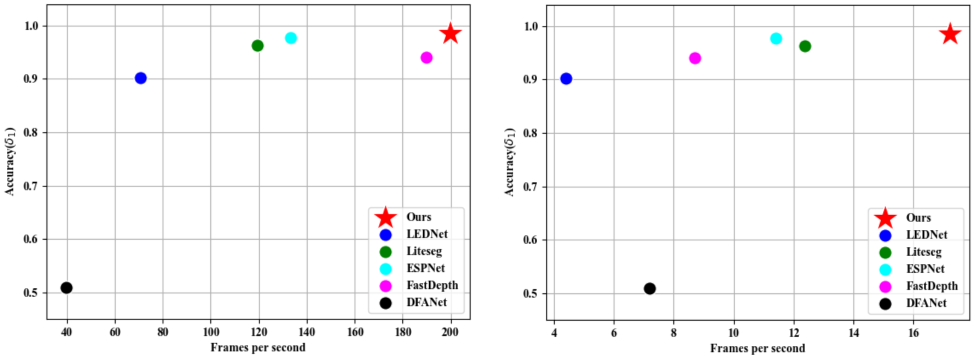
TABLE III: The network's performance when the features of the semantic branch are added into the depth branch.

| Dataset | Metric | w/o fuse | fuse |
|---|---|---|---|
| CAD-60 [26] | RMSE $\downarrow$ | 0.1545 | **0.1526** |
| | $\delta_1 \uparrow$ | 98.69% | **98.72%** |
| | People IoU $\uparrow$ | 96.70% | **96.82%** |
| CAD-120 [26] | RMSE $\uparrow$ | 0.3168 | **0.3140** |
| | $\delta_1 \uparrow$ | 97.90% | **97.98%** |
| | People IoU $\uparrow$ | 95.89% | **96.17%** |
| EPFL RGBD [27] | RMSE $\downarrow$ | 0.3185 | **0.1461** |
| | $\delta_1 \uparrow$ | 97.91% | **98.53%** |
| | People IoU $\uparrow$ | 95.91% | **95.97%** |

TABLE IV: The network's performance through the utilization of the network structure optimization technique.

| Dataset | Metric | w/o optimization | with optimization |
|---|---|---|---|
| CAD-60 [26] | RMSE $\downarrow$ | **0.1524** | 0.1526 |
| | $\delta_1 \uparrow$ | 98.72% | **98.72%** |
| | People IoU $\uparrow$ | **96.85%** | 96.82% |
| CAD-120 [26] | RMSE $\uparrow$ | **0.3133** | 0.3140 |
| | $\delta_1 \uparrow$ | 97.98% | **97.98%** |
| | People IoU $\uparrow$ | 96.14% | **96.17%** |
| EPFL RGBD [27] | RMSE $\downarrow$ | 0.1483 | **0.1461** |
| | $\delta_1 \uparrow$ | 98.50% | **98.53%** |
| | People IoU $\uparrow$ | **96.13%** | 95.97% |

Recall that we propose to incorporate the features from the semantic branch into the depth branch, the effect of this feature fusion operation is shown in Table III. The results show that feature fusion indeed brings performance



(a) NVIDIA P40 GPU

(b) NVIDIA Jetson Nano GPU

Fig. 3: The accuracy ($\delta_1$) and the runtime (measured in frames per second) when applied on NVIDIA P40 GPU (left) and NVIDIA Jetson Nano GPU (right) for various depth estimation frameworks. The EPFL [27] dataset is selected for evaluation. The input images are resized to $224 \times 224$.

TABLE V: Measuring the inference runtime when the network structure optimization is employed.

| Device | w/o optimization | with optimization |
|---|---|---|
| Intel Xeon E5-2640 2.40GHz CPU | 9.34 FPS | **13.80 FPS** |
| NVIDIA Tesla P40 GPU | 179.21 FPS | **199.93 FPS** |
| NVIDIA Jetson Nano GPU | 13.56 FPS | **17.23 FPS** |

gains on all three benchmarks. One can draw that semantic segmentation information can lead to more precise depth estimation. Especially in the EPFL dataset with multiple pedestrians in each image, the incorporation of pedestrians' segmentation information helps the depth estimation branch to better distinguish the depth of each pedestrian, thus significantly boost the depth estimation performance.

In Table IV, a performance comparison is presented, aiming to show the impact of the network structure optimization. Even if some of the three metrics are reduced, the overall performance is not decreased. The network with optimized structure shows a improved performance. Also, we compare the network's forward processing speed for both cases, i.e., when the network structure optimization is employed and without it. As shown in Table V, timings, measured in FPS, on different devices are significantly improved after optimization. Finally, Table VI compares the proposed network when different backbones are applied. As we can see, MobileNetV1 is the fastest one achieving nearly 200 FPS on a Tesla P40 GPU and 17.23 FPS on a Jetson Nano GPU.

### C. Comparison with the baseline techniques

This section compares our method against other representative pipelines, which are: LEDNet [41], LiteSeg [42], ESPNet [43], FastDepth [18] and DFANet [40]. Each of the methods mentioned above add a depth estimation branch for the joint prediction. In Table VII, we list the results obtained for each baseline method and the proposed network. Green values indicate the highest scores, while the blue ones denote the second highest. Since we aim to humans as the main object, People IoU is selected as a metric to demonstrate the performance regarding the semantic segmentation. By examining Table VII, one can observe the significantly high scores achieved by our method in every evaluated dataset. We succeed to excel among every other approach concerning the depth estimation on EPFL RGBD. However, our framework performs unfavourably against other pipelines when compared on CAD-120. The reason is that each dataset has a different maximum value of depth, which is 9.757, 12.4, and 8, for CAD-60, CAD-120, and EPFL RGBD, respectively. Our algorithm is mainly developed to estimate depth in indoor scenes. As a result, the accuracy of this value in a small range is relatively high. Compared to EPFL RGBD, CAD-120 has a broader depth range, so our framework does not have a significant advantage over CAD-120.

Table VIII compares the execution times needed for the proposed network and the baseline approaches when employed on different devices. Notice the increased reduction offered by our network reaching a score of 199.93 FPS and 17.23 FPS on a Tesla P40 GPU and a Jetson Nano GPU, respectively. As a final note, in Fig. 3a and Fig. 3b, our system is measured against other contemporary state-of-the-art solutions on the EPFL RGBD dataset through the accuracy $\delta_1$ over the processing speed (FPS). It is noteworthy that our method outperforms each baseline approach. ESPNet [43] and LiteSeg [42] achieve similar high scores regarding accuracy. Nevertheless, they are much slower. We also test our network optimized with TensorRT on a Jetson Nano GPU. The inference runtime comparison is shown in Table IX. We can see that the model achieves 114.16 FPS, which far exceeds the real-time requirements.

Qualitative results of the proposed network are illustrated in Fig. 4 and Fig. 5, where illustrative results show that our method can accurately segment humans and estimate their depth. There are some pixels with ignored depth values in images of the datasets. We check the ground truth information and get these pixels and, then, we assign these pixels to zero to generate the refined depth prediction results. In this way, it can be visually compared with ground truths more intuitively.

TABLE VI: Comparing the inference runtime of our network when different backbones are used.

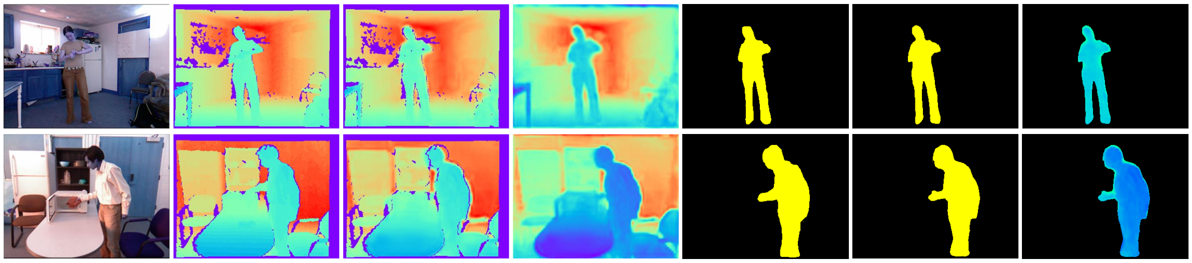
| Device | MobileNetV2 [50] | Resnet-18 [46] | Resnet-50 [46] | VGG16 [45] | MobileNetV1 [19] |
|---|---|---|---|---|---|
| Intel Xeon E5-2640 2.40GHz CPU | 11.24 FPS | 8.11 FPS | 6.49 FPS | 8.89 FPS | **13.80 FPS** |
| NVIDIA Tesla P40 GPU | 120.22 FPS | 169.90 FPS | 112.91 FPS | 170.64 FPS | **199.93 FPS** |
| NVIDIA Jetson Nano GPU | 14.30 FPS | 4.39 FPS | 3.46 FPS | 2.56 FPS | **17.23 FPS** |

TABLE VII: Comparative results of the baseline methods against the proposed method. Green denotes the best, while blue is second best.

| Dataset | Metric | LEDNet [41] | LiteSeg [42] | ESPNet [43] | FastDepth [18] | DFANet [40] | Our Proposed |
|---|---|---|---|---|---|---|---|
| CAD-60 [26] | RMSE ↓ | 0.2300 | 0.1823 | 0.1513 | 0.1559 | 0.2833 | 0.1526 |
| | $\delta_1$ ↑ | 96.28% | 98.25% | 98.77% | 98.66% | 94.39% | 98.72% |
| | People IoU ↑ | 97.31% | 94.35% | 95.06% | 94.50% | 90.17% | 96.82% |
| CAD-120 [26] | RMSE ↓ | 0.4076 | 0.2982 | 0.3249 | 0.3323 | 0.5259 | 0.3140 |
| | $\delta_1$ ↑ | 95.15% | 98.12% | 98.29% | 98.31% | 88.78% | 97.98% |
| | People IoU ↑ | 96.64% | 96.51% | 94.18% | 94.48% | 87.19% | 96.17% |
| EPFL RGBD [27] | RMSE ↓ | 0.3882 | 0.2206 | 0.1804 | 0.2663 | 0.8427 | 0.1461 |
| | $\delta_1$ ↑ | 90.25% | 96.33% | 97.70% | 94.07% | 50.92% | 98.53% |
| | People IoU ↑ | 96.50% | 93.96% | 90.75% | 92.12% | 58.13% | 95.97% |

TABLE VIII: Inference runtime comparison between the baseline and the proposed network.

| Device | LEDNet [41] | LiteSeg [42] | ESPNet [43] | FastDepth [18] | DFANet [40] | Our Proposed |
|---|---|---|---|---|---|---|
| Intel Xeon E5-2640 2.40GHz CPU | 13.66 FPS | 7.79 FPS | 17.65 FPS | **19.67 FPS** | 6.79 FPS | 13.80 FPS |
| NVIDIA Tesla P40 GPU | 70.73 FPS | 119.37 FPS | 133.09 FPS | 189.87 FPS | 39.56 FPS | **199.93 FPS** |
| NVIDIA Jetson Nano GPU | 4.39 FPS | 12.38 FPS | 11.42 FPS | 8.69 FPS | 7.18 FPS | **17.23 FPS** |



Input | Depth GT | Refined Depth Prediction | Depth Prediction | Semantic GT | Semantic Segmentation | Segmentation and Depth Prediction

Fig. 4: Qualitative results of CAD-60 dataset [26] (top) and CAD-120 dataset [26] (bottom).

TABLE IX: Measuring the inference runtime when TensorRT [21] is applied.

| Device | w/o TensorRT | with TensorRT |
|---|---|---|
| NVIDIA Jetson Nano GPU | 17.23 FPS | **114.16 FPS** |

## V. Conclusions and Future Work

As dense metric data allow a mobile robot to perform different tasks, such as obstacle avoidance and metric planning, to achieve a fully autonomous mission, in this paper, a real-time human depth estimation and segmentation network is proposed. Our approach relies on the information provided by a monocular camera, while adopts computational low deep learning techniques to execute in real-time. MobileNetV1, along with ASPP, is used to extract features at different scales, then fused and upsampled. This way, we ensure high accuracy scores, while the processing speed is accelerated through network structure optimization and TensorRT optimizer reaching 114.16 FPS on a Jetson Nano GPU. Our network is evaluated on three self-generated datasets demonstrating an improved performance compared to several state-of-the-art methods. In our plans, we aim to use a monocular camera to realize learning-based collision

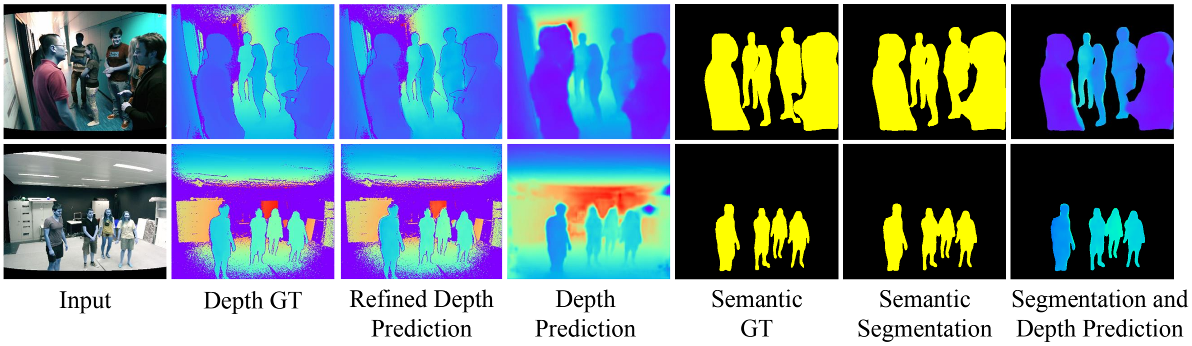


Fig. 5: Qualitative results for the proposed pipeline on EPFL RGBD dataset [27].

avoidance in crowds.

## VI. Acknowledgement

This work was supported by grants from the National Key Research and Development Program of China (Grant No. 2020YFC2006200).

## References

[1] C. Diaz, M. Walker, D. A. Szafir, and D. Szafir, "Designing for depth perceptions in augmented reality," in *Proc. IEEE Int. Symp. Mixed and Augmented Reality (ISMAR)*, 2017, pp. 111–122.

[2] U. Kusupati, S. Cheng, R. Chen, and H. Su, "Normal assisted stereo depth estimation," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[3] M. Mancini, G. Costante, P. Valigi, and T. A. Ciarfuglia, "Fast robust monocular depth estimation for obstacle detection with fully convolutional networks," in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*, 2016, pp. 4296–4303.

[4] ——, "J-MOD 2: Joint monocular obstacle detection and depth estimation," *IEEE Robotics Automation Letters*, vol. 3, no. 3, pp. 1490–1497, 2018.

[5] J.-H. Chen and K.-T. Song, "Collision-free motion planning for human-robot collaborative safety under cartesian constraint," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, 2018, pp. 1–7.

[6] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard, "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *IEEE Trans. robotics*, vol. 32, no. 6, pp. 1309–1332, 2016.

[7] Y. Almalioglu, M. R. U. Saputra, P. P. de Gusmao, A. Markham, and N. Trigoni, "GANVO: Unsupervised deep monocular visual odometry and depth estimation with generative adversarial networks," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, 2019, pp. 5474–5480.

[8] T. Roussel, L. Van Eycken, and T. Tuytelaars, "Monocular depth estimation in new environments with absolute scale," in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*, 2019, pp. 1735–1741.

[9] R. Mur-Artal and J. D. Tardós, "ORB-SLAM2: An open-source slam system for monocular, stereo, and rgb-d cameras," *IEEE Trans. Robotics*, vol. 33, no. 5, pp. 1255–1262, 2017.

[10] A. Bhoi, "Monocular depth estimation: A survey," *arXiv preprint arXiv:1901.09402*, 2019.

[11] N. Durasov, M. Romanov, V. Bubnova, P. Bogomolov, and A. Konushin, "Double refinement network for efficient monocular depth estimation." in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*, 2019, pp. 5889–5894.

[12] J. Hu, M. Ozay, Y. Zhang, and T. Okatani, "Revisiting single image depth estimation: Toward higher resolution maps with accurate object boundaries," in *Proc. IEEE Wint. Conf. Appli. Comput. Vision (WACV)*, 2019, pp. 1043–1051.

[13] K. Wang, Y. Chen, H. Guo, L. Wen, and S. Shen, "Geometric pretraining for monocular depth estimation," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, 2020, pp. 4782–4788.

[14] J. Liang, U. Patel, A. J. Sathyamoorthy, and D. Manocha, "Realtime collision avoidance for mobile robots in dense crowds using implicit multi-sensor fusion and deep reinforcement learning," in *Proc. Int. Conf. Autonomous Agents and Multi-Agents Systems (AAMAS)*, 2020.

[15] I. Kansizoglou, L. Bampis, and A. Gasteratos, "Deep feature space: A geometrical perspective," *arXiv preprint arXiv:2007.00062*, 2020.

[16] V. Nekrasov, T. Dharmasiri, A. Spek, T. Drummond, C. Shen, and I. Reid, "Real-time joint semantic segmentation and depth estimation using asymmetric annotations," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, 2019, pp. 7101–7107.

[17] M. Poggi, F. Aleotti, F. Tosi, and S. Mattoccia, "Towards real-time unsupervised monocular depth estimation on cpu," in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*, 2018, pp. 5848–5854.

[18] D. Wofk, F. Ma, T.-J. Yang, S. Karaman, and V. Sze, "FastDepth: Fast monocular depth estimation on embedded systems," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, 2019, pp. 6101–6108.

[19] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017.

[20] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam, "Encoder-decoder with atrous separable convolution for semantic image segmentation," in *Proc. Eur. Conf. Computer Vision (ECCV)*, 2018, pp. 801–818.

[21] (2020) TensorRT Open Source Software. [Online]. Available: https://github.com/NVIDIA/TensorRT

[22] G. Yang, P. Hu, and D. Ramanan, "Inferring distributions over depth from a single image," *arXiv preprint arXiv:1912.06268*, 2019.

[23] O. H. Jafari, O. Groth, A. Kirillov, M. Y. Yang, and C. Rother, "Analyzing modular CNN architectures for joint depth prediction and semantic segmentation," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, 2017, pp. 4620–4627.

[24] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus, "Indoor segmentation and support inference from rgbd images," in *Proc. Euro. Conf. Computuer Vision (ECCV)*, 2012, pp. 746–760.

[25] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *Int. J. Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.

[26] (2009) Cornell activity datasets: CAD-60 & CAD-120. [Online]. Available: http://pr.cs.cornell.edu/humanactivities/data.php

[27] T. Bagautdinov, F. Fleuret, and P. Fua, "Probability occupancy maps for occluded depth images," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. 2829–2837.

[28] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," in *Proc. IEEE Int. Conf. Computer Vision (ICCV)*, 2017, pp. 2961–2969.

[29] T. Chen, T. Moreau, Z. Jiang, L. Zheng, E. Yan, H. Shen, M. Cowan, L. Wang, Y. Hu, L. Ceze *et al.*, "TVM: An automated end-to-end optimizing compiler for deep learning," in *Proc. 13th Symp. Operating Systems Design and Implementation (OSDI)*, 2018, pp. 578–594.

[30] T.-J. Yang, A. Howard, B. Chen, X. Zhang, A. Go, M. Sandler, V. Sze, and H. Adam, "NetAdapt: Platform-aware neural network adaptation for mobile applications," in *Proc. Eur. Conf. Computer Vision (ECCV)*, 2018, pp. 285–300.

[31] W. Shi, J. Caballero, F. Huszár, J. Totz, A. P. Aitken, R. Bishop, D. Rueckert, and Z. Wang, "Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 1874–1883.

[32] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE Conf. Comp. Vision and Pattern Recognition (CVPR)*, 2015, pp. 3431–3440.

[33] S. Bai and C. Wang, "Information aggregation and fusion in deep neural networks for object interaction exploration for semantic segmentation," *Knowledge-Based Systems*, vol. 218, p. 106843, 2021.

[34] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *Proc. Int. Conf. Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, 2015, pp. 234–241.

[35] V. Badrinarayanan, A. Kendall, and R. Cipolla, "SegNet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 39, no. 12, pp. 2481–2495, 2017.

[36] I. Ulku, P. Barmpoutis, T. Stathaki, and E. Akagunduz, "Comparison of single channel indices for u-net based segmentation of vegetation in satellite images," in *Proc. 20th Int. Conf. Machine Vision (ICMV)*, 2020.

[37] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 40, no. 4, pp. 834–848, 2017.

[38] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," *arXiv preprint arXiv:1706.05587*, 2017.

[39] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Semantic image segmentation with deep convolutional nets and fully connected crfs," *arXiv preprint arXiv:1412.7062*, 2014.

[40] H. Li, P. Xiong, H. Fan, and J. Sun, "DFANet: Deep feature aggregation for real-time semantic segmentation," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 9522–9531.

[41] Y. Wang, Q. Zhou, J. Liu, J. Xiong, G. Gao, X. Wu, and L. J. Latecki, "LEDNet: A lightweight encoder-decoder network for real-time semantic segmentation," in *Proc. IEEE Int. Conf. Image Processing (ICIP)*, 2019, pp. 1860–1864.

[42] T. Emara, H. E. Abd El Munim, and H. M. Abbas, "LiteSeg: A novel lightweight convnet for semantic segmentation," in *Proc. Digital Image Computing: Techniques and Applications*, 2019, pp. 1–7.

[43] S. Mehta, M. Rastegari, A. Caspi, L. Shapiro, and H. Hajishirzi, "ESPNet: Efficient spatial pyramid of dilated convolutions for semantic segmentation," in *Proc. Eur. Conf. Computer Vision (ECCV)*, 2018, pp. 552–568.

[44] S. Mehta, M. Rastegari, L. Shapiro, and H. Hajishirzi, "ESPNetv2: A light-weight, power efficient, and general purpose convolutional neural network," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2019, pp. 9190–9200.

[45] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014.

[46] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2016, pp. 770–778.

[47] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," *arXiv preprint arXiv:1511.07122*, 2015.

[48] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common objects in context," in *Proc. Eur. Conf. Computer Vision (ECCV)*, 2014, pp. 740–755.

[49] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in neural information processing systems*, 2019, pp. 8026–8037.

[50] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, "MobileNetV2: Inverted residuals and linear bottlenecks," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2018, pp. 4510–4520.