SmallDepthMask: Large DataSet Generation for Monocular Depth Estimation and Foreground Segmentation from few Internet Images

Anonymous CVPR 2021 submission

Paper ID ****

Abstract

Segmentation of the desired object along with depth estimation is useful in various applications like robotics and autonomous navigation. Any deep learning workflow to estimate monocular depth and segment the desired foreground object in a scene require significant training data. The data generation process usually involve expensive hardware like RGB-D sensors, Laser Scanners or significant manual involvement. Moreover, for every specific foregroud object, the data collection process need to repeat. This paper presents a novel way to utilize only a small number of readily available png images with transparency for the foreground object, and representative background images from the internet and combine them to generate a large dataset for deep learning, utilizing current state of the art monocular depth estimation techniques. To illustrate the effectiveness of the data generation approach, this paper presents a baseline model for depth and foreground mask estimation for detecting cattle on road using the generated data from proposed approach. The baseline models exhibit strong generalization to real scenarios. The generated dataset is available for public use.

1. Introduction

Depth estimation and segmentation of desired objects in the scene are often used together in many vision tasks [**?**] like autonomous navitation of agents, augmented reality, self driving cars and other robotics applications. In all these applications, identification of desired objects precisely in the scene and its depth estimate from the camera are crucial for safe and effective navigation. Modern RGB-D sensors like OAK-D OpenCV AI Kit: OAK-D (https://opencv.org/introducing-oak-spatial-ai-powered-by-opencv/)[0]re capable of simultaneously running advanced neural networks while providing depth from two stereo cameras and color information from a single 4K camera in the center. Deep learning based techniues using convolution neural nets have effective solution in both depth estimation and semantic segmentation. In general for high accuracy outcomes, a deep learning network is dependent on large training dataset availability. To gather such data itself incur high cost and time. For specific applications requiring several forground objects against variety of backgrounds become even more challenging in terms of simulating those scenarioes. Synthetic datasets using Virtual Reality have been proposed to that end [**?**].

Recent research indicates effective use of readily available images on the internet to curate training data [**?**]. This paper introduces SmallDepthMask, a way to curate custom dataset containing hundereds of thousands to millions of images by multiplexing desired foreground objects over representative background scenes, while also generating corresponding depth and foreground mask images. This significantly reduces the cost and time overheads. The authors also experiment by creating baseline models for sev-

[0]a

eral application contexts and show that the generated data successfully generalizes to detect relevant objects in real scenes. Multiplexing, combined with random cropping, scaling and translation, make the datageneration fast and effective. With only 100 pair of background and foregroung images, the authors generate 0.4 million image triplets and effectively leverage existing SOTA models for depth estimation.

The main contributions of this paper are the following:

1. A novel effort to mix and match foreground and background images reducing the need for complex scene generation for data curation.

2. Curate large dataset to effectively train models for custom applications of detecting depth and mask for specific foreground objects over any target background, from a limited input of ready available internet images.

3. Combine image, depthmap and forground mask in a single dataset using current SOTA models for depth estimation.

4. To release curated dataset and the trained models making them publicly available. Researchers can use this single dataset to do segmentation, train models to predict depth, or to predict both depth and mask.

Title figure represents an example from the generated dataset to help detect cattle on road, which is very common on Indian roads, leading to several accidents involving loss of life and property. The generated datasets and the trained models are publicly available[1].

2. Related Work

A depth image is an image channel in which each pixel relates to a distance between the image plane and the corresponding object in the RGB image.Monocular depth gives information about depth and distance and the Monocular Depth Estimation is the task of estimating scene depth using a single image[**?**]. Image Segmentation is the process of partitioning an image into multiple segements and it can be used for locating objects, boundaries [**?**]. RGBD image is a combination of a RGB image and its corresponding depth image[**?**].

Depth information is integral to many problems in robotics including mapping, localization and obstacle avoidance for terrestrial and aerial vehicles, autonomous navigation, and in computer vision, including augmented and virtual reality[**?**]. RGBD datasets usually collected using depth sensors, monocular cameras and LiDAR scanners are expensive and data collection is a time consuming job. The well known datasets for monocular 3D object detection are Context-Aware MixEd ReAlity (CAMERA), Objectron, Kitty3D, Cityscape3D, Synthia, etc. [] and these datasets have limitations like indoor only images, small number of training examples and sparse sampling.

To address the issues usage of expensive devices and small number of training examples, this paper proposes a technique to come up with a custom dataset by using existing accurate depth predictor models, like High Quality Monocular Depth Estimation via Transfer Learning(nyu.h5) [**?**].

A variety of RGBD datasets in which images are paired with corresponding depth maps(D) have been proposed through the years.Some of the frequenlty used RGBD datasets are the Kitti dataset [**?**], the Synthia dataset [**?**], Make3D dataset [**?**], NYU dataset [**?**]

The dataset Kiiti [**?**] is the well known RGB-D dataset collected using a vehicle equipped with a sparse Velodyne VLP-64 LiDAR scanner and RGB cameras, and features street scenes in and around the German city of Karlsruhe. The Primary application of this dataset involves perception tasks in the context of self-driving. Synthia [**?**] is a street scene dataset with depth maps of synthetic data, requiring domain adaptiation to apply to real world settings. Cityscapes [**?**] provides a dataset of street scenes, albeit with more diversity than KITTI. Sintel [**?**] is another synthetic dataset which mainly comprises of outdoors scenes.

Megadepth [**?**] is a large-scale dataset of outdoor images collected from internet, with depth maps reconstructed using structure-from-motion techniques, but this dataset lacks in ground truth depth and scale. The RedWeb [**?**] dataset provide depth maps generated from stereo images which are freely available in large-scale data platforms such as Flicker. The datasets MegaDepth and RedWeb can be easily computed with the existing MVS methods.

Make3D [**?**] provides RGB and depth information for outdoor scenes. The NYUv2 dataset [**?**] is widely used for monocular depth estimation in indoor environments. The data was collected with a Kinect RGBD camera, which provides sparse and noisy depth returns. These returns are generally in-painted and smoothed before they are used for monocular depth estimation tasks. As a result, while the dataset includes sufficient samples to train modern machine learning pipelines, the "ground-truth" depth does not necessarily correspond to true scene depth.

Most of the existing datasets consists of indoor images, or outdoor images of city streets. For every specific application, like detecting animals roaming on roads for self driving or assisted driving cars, or people inside a room for autonomus room cleaners etc. Researchers need to curate specific dataset to train relevant deep learning models. The present work makes the task of curating dtaset extremely simple and cost effective.

[1]Curated Dataset for Cattle on Road: https://www.kaggle.com/bsridevi/modes-dataset-of-stray-animals

3. Method

The curated dataset must have following objectives:

1. dataset which dedicatedly includes foreground object.
2. dataset should drive deep learning models and generalize.
3. dataset should provide accurate dense depth maps.
4. dataset should provide foreground mask
5. dataset can be stored offline or generated online during training phase dynamically

3.1. Data Acquisition

The first step to curate data is to determine a target application scenario and thus determine the foreground object(s) and the representative background context. At the same time the dataset must have sufficient variability to include majority of the types and views that the trained deep network may see when deployed.

We propose to download or take RGB image of n foreground object(s) and m background images (we used $n = m = 100$) balancing the types and views. For example, for cattle on roads dataset, we shose several cow, bull and calf types, individual or in group, sitting, standing or walking, and from various angles. Similarly for background, we chose backgrounds of streets, storefronts, main roads, highways, markets, railway tracks, landscapes, garbage piles etc. PNG images with transparency are readily available on the internet for almost any desired foreground object. Such images will easily allow to generate foreground mask from non-transparent pixels. If not, tools like GIMP [**?**], combined with deep learning forground extractors[2] can help generate the required PNG foreground images.

Fig. **??** shows few of the sample scene and foreground images used for the creation of this dataset.

3.2. Multiplexing and Depth Generation

This step is to place each foreground object on to several background images generating a fg-bg image and the corresponding mask corresponding to the foreground placement and scale. Depth is computed from the fg-bg image via the model proposed by Ibraheem Alhashim et al. in their paper titled "High Quality Monocular Depth Estimation via Transfer Learning" [**?**][3]. This model takes 448×448 size images as input, hence we resize all backfround images to this size while maintaining their aspect ratio.

The data generation process is completely online and produces one batch of images for training a deep model. By repeating one forground object k times for each background image and repeating another k times with horizontally flipped version of the same foreground, one can generate $2kmn$ fg-bg images. For $k = 20$ and $n = m = 100$ this becomes $400,000$ fg-bg images. Algorithm **??** describes the data generation process

[2]https://www.remove.bg/ uses a combination of Image based techniques and DNN to separate foreground from background

[3]source code for depth estimation model: https://github.com/ialhashim/DenseDepth/blob/master/DenseDepth.ipynb

4. Experimental Analysis

In this section, we provide a baseline for monocular depth estimation and foreground segmentation on the *SmallDepthMask* dataset. Convolutional Neural Networks(CNN) are progressive in exploring structural features and spatial image formation. To come up with baseline, we started training simple CNN, Resnet, and Unet++. The state-of-the-art models for image segmentation are variants of U-Net and fully convolutional networks (FCN)[**?**]. Long skip connections are used to skip features from the contracting path to the expanding path in order to recover spatial information lost during downsampling [**?**]. Short skip connections can be used to build deep FCNs.

4.1. Model

By using both long and short skip connections we proposed a light weight model following U-Net architecture with two decoder networks meant for segmentation mask prediction and depth prediction. The architecture of the model is shown in Fig **??**. The total number of parameters of this model are $5,525,568$ including both the decoders.

The encoder part of network is comprised of four DownSampling units. Every downsampling unit compresses the input scene image with the help of a series of convolutional operations. In our implementation, the source image of size $128X128$, changed into $64X64 \rightarrow 32X32 \rightarrow 16X16 rightarrow 8X8$. This model has DepthDecoder and MaskDecoder and each of them is comprised of four upsampling units. The compressed source image is expanded with the help of Atrous and Transposed convolution operations. The encoder outcome $8X8$ is expanded into $16X16 \rightarrow 32X32 \rightarrow 64X64 \rightarrow 128X128$. As shown in model architecture Fig.**??** the outcome of encoder downsampling units were added to outcomes of decoder upsampling units.

We have trained this model on the entire *SmallDepthMask* dataset from scratch with train-test-split of $70-30\%$. During training the network is trained with the batch size of 64 for 10 epochs using SGD optimizer[**?**]. Every epoch took one hour of time on GPU because of the huge training data. We have used OneCycleLR scheduler [**?**] with a maximum Learning rate of 0.1. This made the initial learning rate as 0.0099. The Deep Convolutional Neural Networks encoder is fed with a image $(128X128)$ and the first decoder outputs a mask image and and second decoder outputs a depth image. To reduce overfitting[**?**], and achieve generalization

Figure 1. Scene and foreground object images

Algorithm 1: Generate Dataset(*[bgimages]*, *[fgimages]*, k, b)

input : m Background Image paths, $2n$ Foreground Image paths, multiplexing factor k, batch size b *must be multiple of* k
output: Yield $2kmn$ fg-bg, mask and depth images in batches of size b

for offline use it creates 3 folders with fg_bg, mask and depth each having $2kmn$ *images*;
for $bg \leftarrow 1$ **to** m **do**
  for $fg \leftarrow 1$ **to** $2n$ **do**
    for $i \leftarrow 1$ **to** k **do**
      $croppedbg \leftarrow$ take maximal random crop of 448×448 from bg without affecting the aspect ratio;
      randomly pick a center point (x, y) in range $[0, 447]$;
      randomly pick a scale in range $[0.3, 0.6]$ (ratio of area fg covers bg);
      create $fg - bg$ image by resizing the fg to scale and place it on top of $croppedbg$ centered at x, y calculated;
      calculate binary mask from current placement of fg by thresholding the transparency channel. save $fg - bg$ image and mask add $fg - bg$ image to a batch;
    if b *new fg-bg images generated* **then**
      run depth model on batch and save corresponding depth images. Rescale saved images to 224×224

this work employed the augmentation techniques, Random Rotation, Random Grayscale, Color Jitter, random horizontal flips and random channel swaps.

4.2. Loss function

Deciding a universal loss function is not possible for complex objectives like Image segmentation and depth prediction. Based on the survey done by Shruti Jadon [**?**] we have picked L1 loss and SSIM(Structural Similarity Index) loss [**?**]. Their work also suggested to use penalty term which helps the network to focus towards hard-to-segment boundary regions. The Loss is calculated with the help of L1 and SSIM at both the decoders and employed regularization for weight penality.

For training our network with two decoders, we defined the same loss function L for depth and mask prediction, between y and $\hat{y}$ as the weighted sum of two loss function values.

$$L(y, \hat{y}) = \lambda L_{term1}(y, \hat{y}) + (1 - \lambda) L_{term2}(y, \hat{y}) \quad (1)$$

The first loss term $L_{term1}(y, \hat{y})$ is the point-wise L1 loss defined on the predictions of Mask Decoder and Depth Decoder units of the network.

$$L_{term1}(y, \hat{y}) = \frac{1}{n} \sum_{x=1}^{n} \mid y_i - \hat{y_i} \mid \quad (2)$$

The second loss term $L_{term2}(y, \hat{y})$ uses a commonly used metric for image reconstruction task i.e., SSIM. Many recent toody depth prediction CNNs employed this metric. The loss term is redefined as shown in equation as SSIM has an upper bound of one.

$$L_{term1}(y, \hat{y}) = \frac{1 - SSIM(y, \hat{y})}{2} \quad (3)$$

Different weight parameters λ were tried and we have ended with a value $\lambda = 0.84$. The final loss function is as follows.

$$L(y, \hat{y}) = 0.84 * L_{term1}(y, \hat{y}) + 0.16 * L_{term2}(y, \hat{y}) \quad (4)$$

Figure 2. Three images resulted from the algorithm used. (top) A scene image on which a foreground object is positioned at random location with random scale, (middle) respective mask for the scene image, and (bottom) calculated depth by using a model.

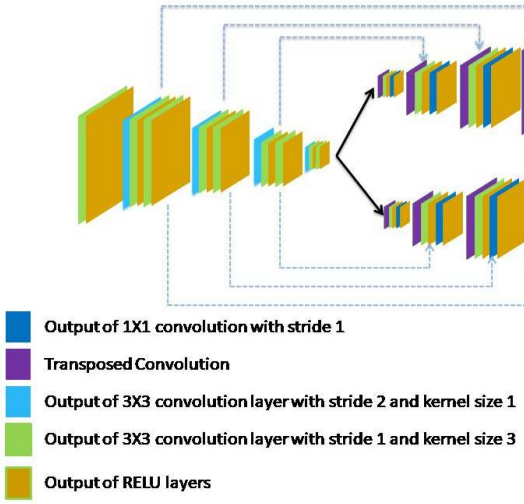

Figure 3. Network Architecture

4.3. Optimizer and Learning Rate

MENTION DETAILS HERE

4.4. Results

The model is trained on the entire dataset and obtained significant accuracy and minimal loss. The outcome of the model on validation dataset is shown in Fig **??**. and on the unseen data is shown in Fig **??**. The unseen data fed to the model is a real picture and not one curated as in *SamllDepthMask* where forgerround is placed on background. From the obtained segmentation masks and depths, It is very clear that the model generalized well. Few exceptions like the spots on cow and two calves not having very good detection indicate the non-presence of such examples in the training set. However, it should be straighforward to introduse few more examples to make the application more robust.

5. Conclusion and Future Work

WRITE MORE HERE HOW WE FULFILLED THE CLAIMS Robust dataset for Image Segmentation and depth generation is proposed whose implementation cost is low. A lightweight baseline model to infer both mask and depth is proposed.

Due to random scaling and placement the fg-bg images are often not so realistic. Figure **??** middle show sample images with wrong placement and scale. Regardless of this, the trained model is generalizing well. However, we experimented with detection of ground and sky regions from semantic segmentation. [ONE LINE WITH REFERECE TO THE WORK WE USED FOR THIS] That combined with the depth information of the background gives necessary cues as to where to place the foreground image and at what scale. Figure **??** shows the outcomes. We further experimented with adding occlusions from semantic segmentation information where non ground/sky pixels that belong to regions starting below the top margin of the foreground location, are placed on top of the foreground. Figure **??** illustrates this. The effect of such data on training is yet to be explored. At the same time, the generated images by such informed scaling and placement are also not free from artifacts. The orientation and unknown size of foregroung object pose a challenge, leading to use of several experimental constants in the transformation process. We would like to formalize that and analyze its outcome on training as future work.

6. Acknowledgement

Authors are grateful to Vishnu Institute of Technology for providing necessary infrastructure, and to Rohan Sravan of The School of AI for initiating the idea and providing necessary support to carry out the research.

Figure 4. Segmentation mask and depth inference on validation data

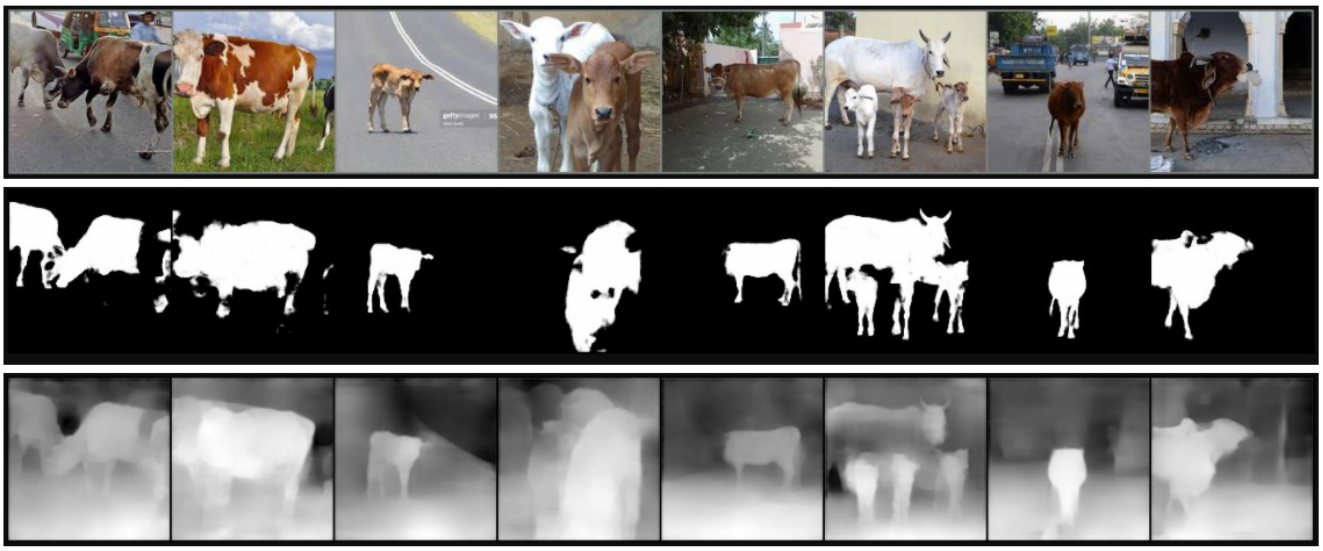

Figure 5. Segmentation mask and depth inference on unseen data.

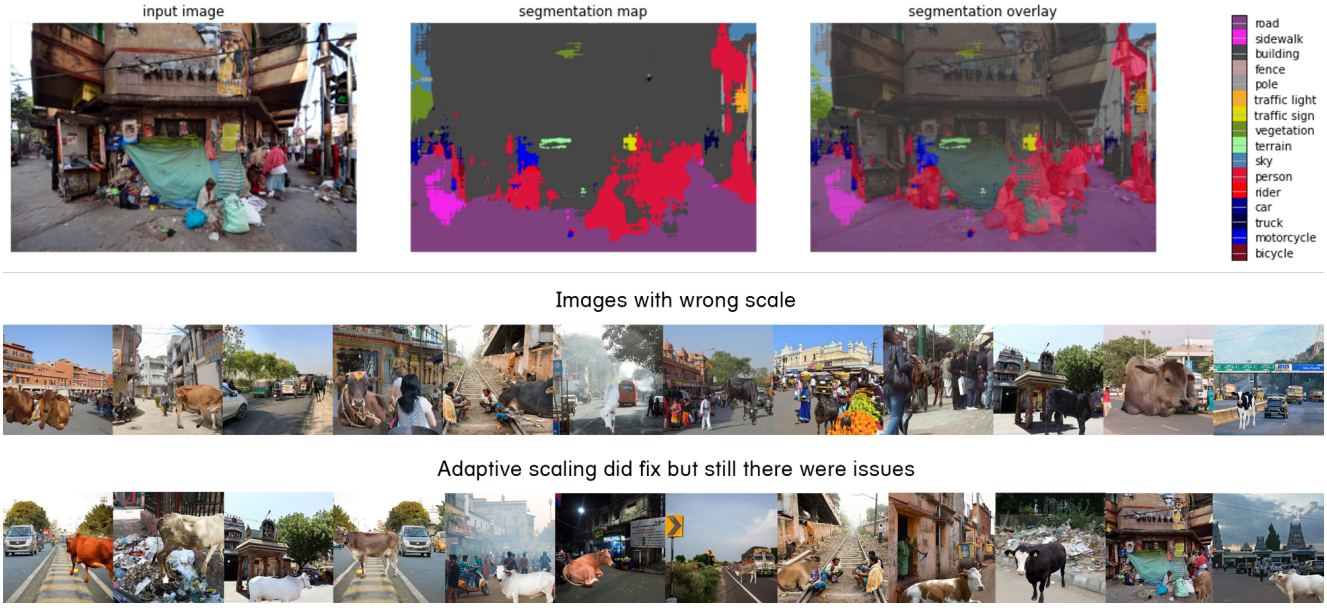

Figure 6. Using Semantic Segmentation for adaptive placement and scaling. (Top) semantic segmentation. (Middle) Example of poor scaling and placement in proposed approach. (Bottom) Attempted correct placement and scaling

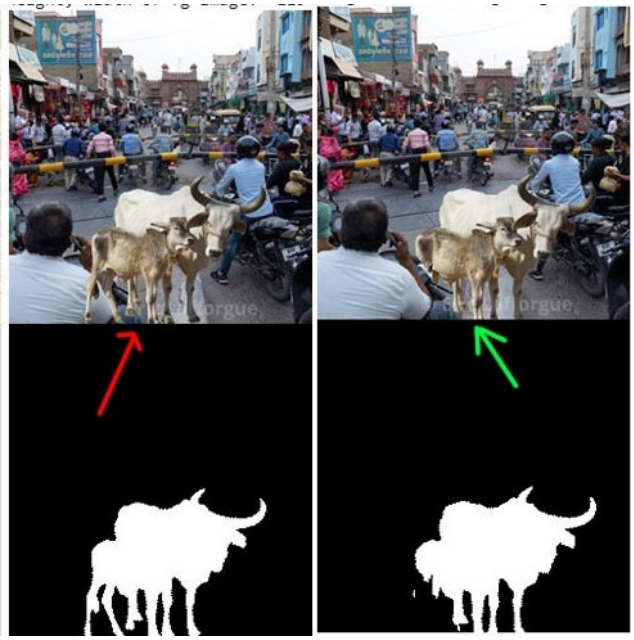

Figure 7. Restoring occlusion from semantic segmentation