Self-Supervised Monocular Depth Estimation

ATHULKUMAR R*, 20250018
K S ASHIN SHANLY*, 20250019
SUDIP DAS*, 20210005

KEYWORDS

Lambertian surfaces, LIDAR, Re-projection error, ResNet18, Self-Supervised Learning

1 PROBLEM STATEMENT

Monocular depth estimation is a computer vision task in which a depth map is estimated from a single 2D image. The input is an RGB image, and the output is a depth image. Depth image includes the information about the distance of the objects in the image from the camera. Estimating depth from images plays a crucial role in many key applications like autonomous driving, automatic 2D-to-3D conversion, AR-composting, rendering of 3D scenes, grasping in robotics, etc. Depth sensors like RGB-D cameras and LIDAR provide good depth estimation but are costly and consume much more power than traditional cameras. Due to the low cost, small size, and wide applications of monocular cameras, estimating the dense depth map from a single image is an interesting field to work in. We face some challenges while extracting depth image from an RGB image: occlusion, non-Lambertian surfaces, etc. There may be many 3D scenes that can give the same picture when projected onto a 2D plane. This makes the estimated depth non-unique. Usage of deep supervised networks is a powerful way to estimate depth from RGB images. However, curating large and varied datasets with accurate ground truth depth images is a formidable task.

Recent works have proposed an alternative approach for this problem. They propose a self-supervised learning approach in which a pair of stereo images or a monocular video is enough to train the depth estimation models. This method is self-supervised in the sense that the ground truth comes from the input signal itself. In this case, the RGB images. There is no need for any external data or signal to teach the network. The depth estimator itself is its teacher. Although this alternative approach helps us bypass the formidable task of curating data sets with accurate ground truth depth images, it has its own drawbacks.

2 RELATED WORKS

2.1 Supervised Depth estimation

Depth Estimation is a classical and inherently ill-posed problem of computer vision. Early works focused on calculating the depth from synchronized stereo images by using 3D geometry-based algorithms [1, 3]. These words relied on point correspondences between two images and, by using triangulation, estimate the depth. The emergence of deep learning has revolutionized computer vision, and many depth estimation Convolutional neural networks have been proposed in recent years. In [6], the authors used convolutional neural networks and train them to predict instance-level segmentation directly. Here the instance ID encodes the depth ordering within image patches. To provide a single coherent image, they developed a Markov random field that uses the CNN predictions. It predicted accurate instance-level segmentation and depth ordering of the input image.

Authors' addresses: AthulKumar R*, athulkr@iitgn.ac.in20250018; K S Ashin Shanly*, ksashin@iitgn.ac.in20250019; Sudip Das*, sudipd@iitgn.ac.in20210005.

2.2 Self-Supervised Depth estimation

Recently, self-supervised learning is getting the attention of researchers. The main weakness of most current deep Convolutional networks is the lack of data. In [2], they used an inverse warp of the target image using the predicted depth and known interview displacement. This is used to reconstruct the source image. They used the photometric error in this reconstruction loss as their loss function. Their network trained on less than half of the KITTI dataset gives comparable performance to state-of-the-art supervised methods for single view depth estimation. With the advent of powerful deep networks like VGG and ResNet, the depth estimation accuracy has been boosted in recent years. In [5], the authors proposed a 2D-to-3D algorithm that takes 2D images or video frames as inputs and outputs 3D stereo image pairs. They used VGG16, which is a large convolutional network trained on Image-Net.

3 APPROACH

Collecting large datasets with accurate ground truth depth labels for supervised learning is difficult. Nevertheless, they give state-of-the-art performance in the field of monocular depth estimation when trained on standard datasets like KITTI [4]. The alternative approach is to use self-supervised learning. There are two approaches to self-supervised learning. The model can be trained using either synchronized stereo images or monocular videos. We trained the self-supervised model using only monocular videos here. There is no need for labeled data as the network teaches itself.

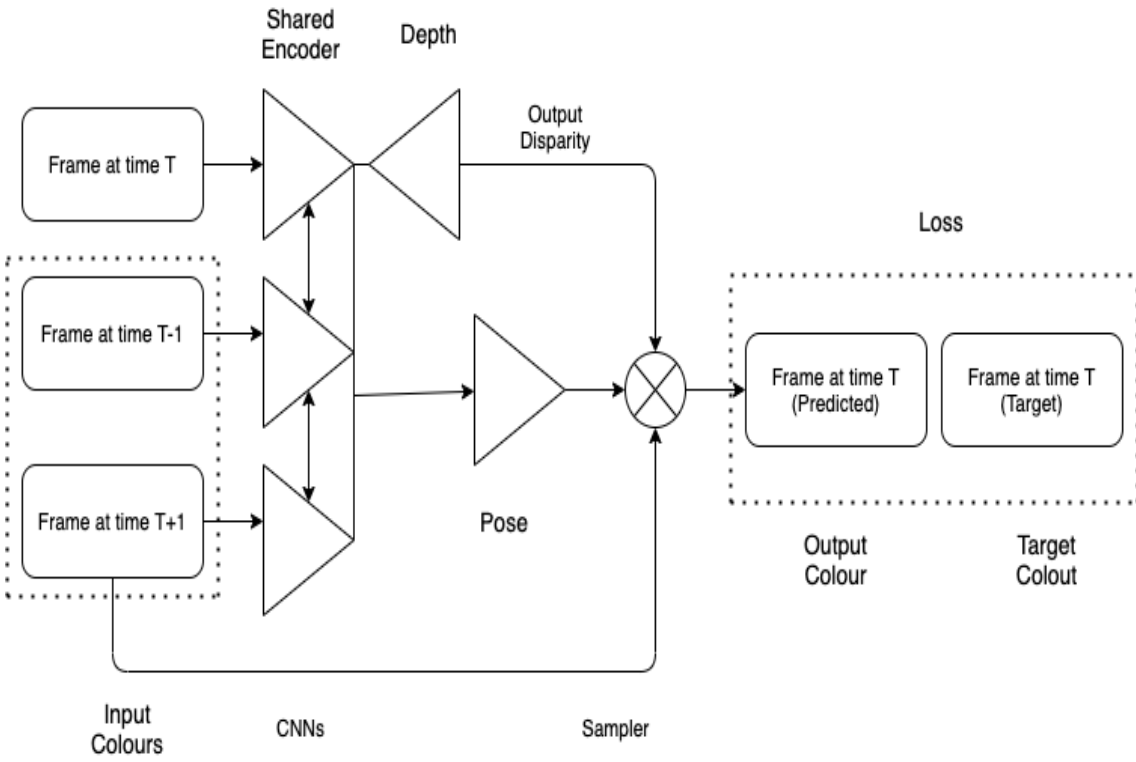

Fig. 1. Model Architecture

In contrast to training on stereo images, using monocular video has its own challenges. The model's objective is to generate an image with a different pose based on some input image. The target image is generated by using 3D scene geometry. The model pipeline is shown in Figure 1. The model needs to estimate depth along with the egomotion between two frames of a video during training. When depth, together with ego-motion, is provided, we can project a new view (target) by applying a projective warping from the source camera point of view. The warping is achieved using a view synthesis module here. Depth is an input to the module and, in our case, is predicted from a neural network. Then, learning is guided by minimizing a proxy photometric loss between the target and the projected target, and gradients are derived and propagated through the bilinear sampler module and the depth network. So, if the target image and the projected target are similar appearance-wise, this implies that depth is implicitly and correctly learning. A depth network is used to understand motion parallax. At the same time, a pose network is used to predict the change in observation angle between frames.

3.1 Training with Monocular video

The model's objective is to generate the target view I_t using images at timestep I_{t-1} and I_{t+1}. We constrain the network to perform an image synthesis task using an intermediary variable, depth or disparity. Now, we can estimate the depth of the image by accurately reconstructing the target image from the sources. The problem is that there can be multiple incorrect depths for each pixel that can give us a correct re-projection of the target image. This ambiguity can be addressed by enforcing smoothness in the depth maps.

3.2 Improvements

State-of-the-art monocular methods provide lower quality depths compared to the best of supervised models. To close this gap, three improvements are suggested. These will significantly increase the quality of the predicted depth without inducing additional model components.

Per-Pixel Minimum Reprojection Loss

Existing self-supervised depth estimation methods take the average across multiple source images while computing the re-projection error. Pixels that are visible in the target image but not visible in one of the source images can create problems here. It can be either occluded pixels or those out-of-view pixels at the image boundaries because of the egomotion.

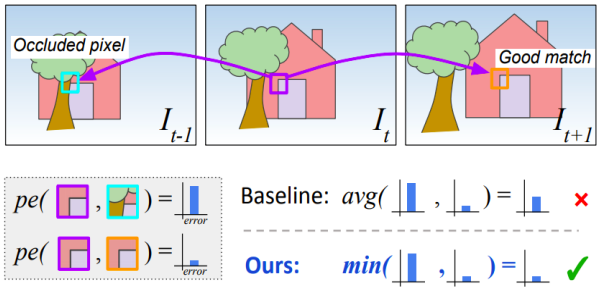

Fig. 2. Per-pixel minimum reprojection loss: Taking the minimum loss among the source images rather than averaging out the losses.

The network may correctly predict such pixels, but it will not match those source images where that pixel is not visible. This will induce a high photometric error penalty.

The proposed improvement that can deal with both kinds of pixels is to use the minimum photometric error among the source images instead of their average (figure 2). Rather than averaging the photometric error in all the source pictures at every pixel, we just use the minimum. The final formula of per-pixel photometric loss is

$$L_p = min_{t'} \, pe(I_t, I_{t'->t})$$

Using this component in the minimum re-projection loss considerably reduces artifacts that often exist at image borders. This improves the sharpness of occluded pixels at image boundaries and leads to an overall better accuracy.

Auto-Masking Stationary Pixels

One of the most important assumptions we take while self-supervised monocular training is of a moving camera and a static scene. But these assumptions can break down frequently. One such instance is when the objects in the scene are moving. Such motion can create 'infinite depth holes' for those moving objects in the predicted depth image.

A simple yet effective solution to this problem is an auto-masking technique that filters out those pixels that do not change appearance from frame to frame. This lets the model ignore those objects that move at the same velocity as the camera or if the camera is stationary, ignore the whole frame. The per-pixel mask, μ, used here is a binary mask, μ 0, 1. Instead of being learned or estimated from object motion, it is computed automatically during the forward pass of the network. μ is set to 1 only for those pixels where the reproduction error of the projected image $I_{t'->t}$ is lower than that of the original source image $I_{t'}$.

$$\mu = [\ min_{t'} \, pe(I_t, I_{t'->t}) < min_{t'} \, pe(I_t, I_{t'})\ ],$$

where [] is the Iversion bracket. In cases where an object and the camera are both moving at a similar velocity, the mask prevents such stationary pixels in the image from contaminating the loss. When the camera is static, the same mask can filter out all pixels in the image.

Multi-scale Estimation

Current models use multi-scale depth prediction and reconstruction of images to prevent the objective function from getting stuck in local minima. Loss at each scale is calculated, and the combination of these individual losses will give us the total loss. This method has a defect that it can create 'holes' in the depth map for low texture regions. To improve the multi-scale formulation, we can decouple resolutions of the color images and depth images. We can upsample the lower resolution depth images to higher resolution (same as that of the input image), thus reducing the photometric error on the ambiguous low-resolution images. This high resolution image will be used to reproject and compute the error. By this, we can constrain the depth maps at each scale to work towards the same objective of reconstructing high-resolution input target image as accurately as possible.

3.3 Experimental Setup

The standard KITTI data set is used. The depth estimation network of the model is based on an encoder-decoder network with skip connections. This allows the model to represent both deep abstract features and local information equally well. ResNet18 is used as the encoder, containing 11M parameters. This is smaller and faster than Resnet50, which is usually used in other similar works. The weights are initialized to weight values pre-trained on ImageNet. Depth decoder has sigmoid activation function at the output and ELU nonlinearities at remaining layers. We convert the sigmoid output σ to depth with $D = 1/(a\sigma + b)$, where a and b are chosen to constrain D between 0.1 and 100 units. In the decoder, reflection padding is used instead of zero paddings. Reflection padding returns the value of the nearest boundary pixels in the source image when samples land outside of the image boundaries. For pose estimation, the model predicts the rotation using an axis-angle representation and scale the rotation and translation outputs by 0.01.

To train the model on monocular videos, the model uses a sequence length of three temporally adjacent frames. ResNet18 is used as the pose network. It is changed in a way such as to allow a pair of color images as input and to output a relative pose with 6 degrees of freedom. The smoothness term λ is set to 0.001. Horizontal flips and the following training augmentations are performed with 50% chance: contrast, saturation, random brightness, and hue jitter with respective ranges of ±0.2, ±0.2, ±0.2, and ±0.1. Color augmentations are done only on images given to the networks. All three images fed to the pose and depth networks are augmented with the same parameters. The model is implemented in PyTorch and trained for 20 epochs. The batch size chosen was 12, and the input/output resolution was 640 × 192. For the first 15 epochs, the learning rate used was 0.0001, and it was reduced to 0.00001 for the remaining five epochs. The learning rates were chosen using cross-validation with a validation set size of 10% of the data.

4 RESULTS

The model was tested with images of different scenes with varied illumination. The resulting depth maps for different input RGB images are shown in figure 3. The model was tested on a video as well where each frame was extracted and the depth was estimated
The model fail to accurately delineate objects where boundaries are ambiguous or shapes are intricate. When the input contained objects that do not satisfy the Lambertian assumptions of appearance loss, the model was observed to not give satisfactory results as expected. The results of such cases can be found in figure 4. The side-view mirror of the car in the image violates Lambertian assumptions, and therefore is predicted to have infinite depth and hence, is seen as a 'hole'.

5 KEY OBSERVATIONS

The model outputs satisfactory results when compared to state-of-the-art models. The model does not work when the input image contains objects that do not obey the Lambertian assumptions of appearance loss. It failed to correctly identify depths of objects which had reflective surfaces and uneven brightness. Some objects that were not stationary in the scene were predicted with infinite depth for video sequence input, even though binary automasking was used. We observed that the variant of the same model trained with stereo pairs gave similar results as well.

We found that using reflection padding instead of zero paddings in the decoder reduced the boundary artifacts to a large extent. This is because it returns the value of the nearest boundary pixels in the source image when samples land outside of the image boundaries.

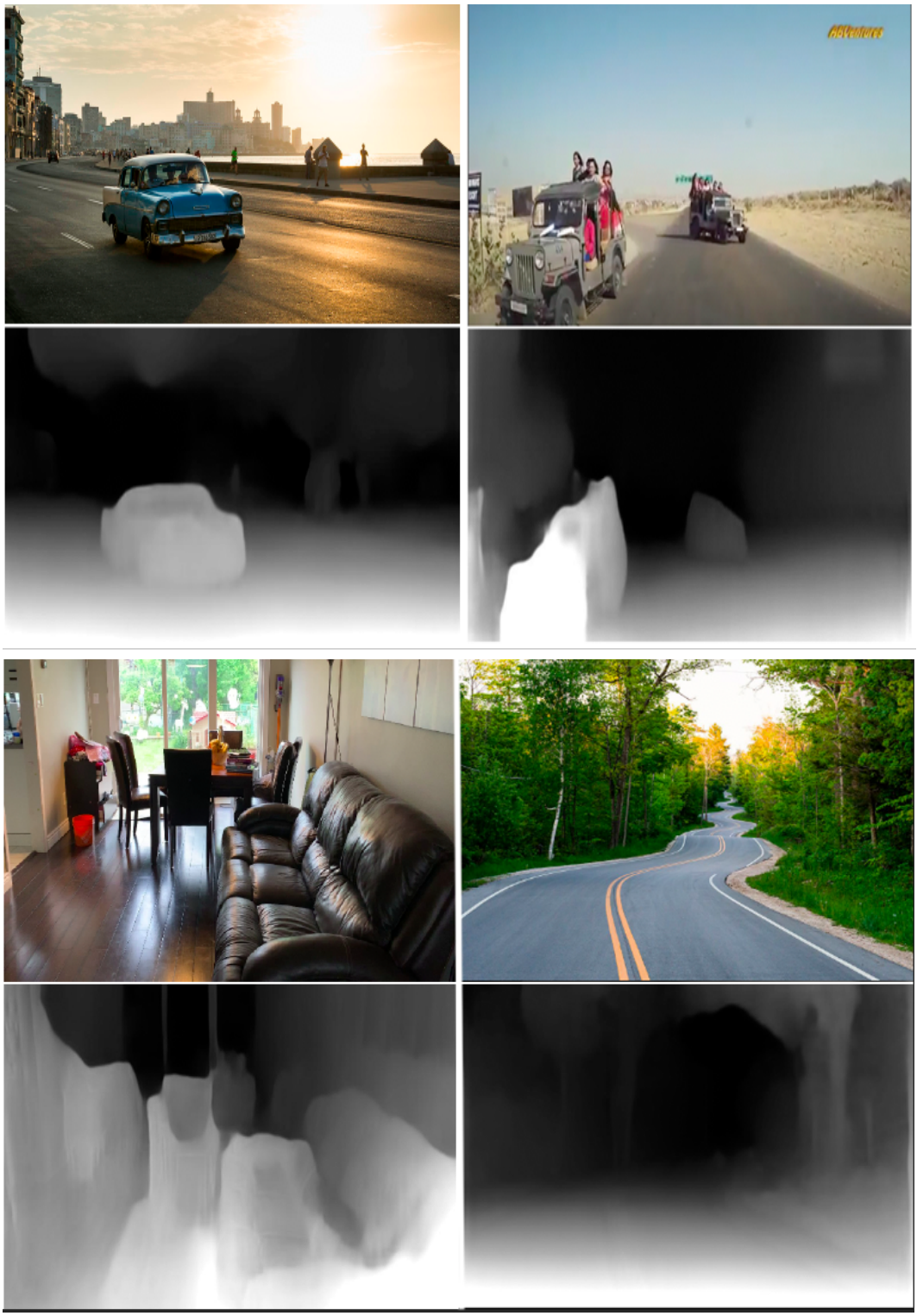

Fig. 3. Images and their corresponding depth images produced from the model

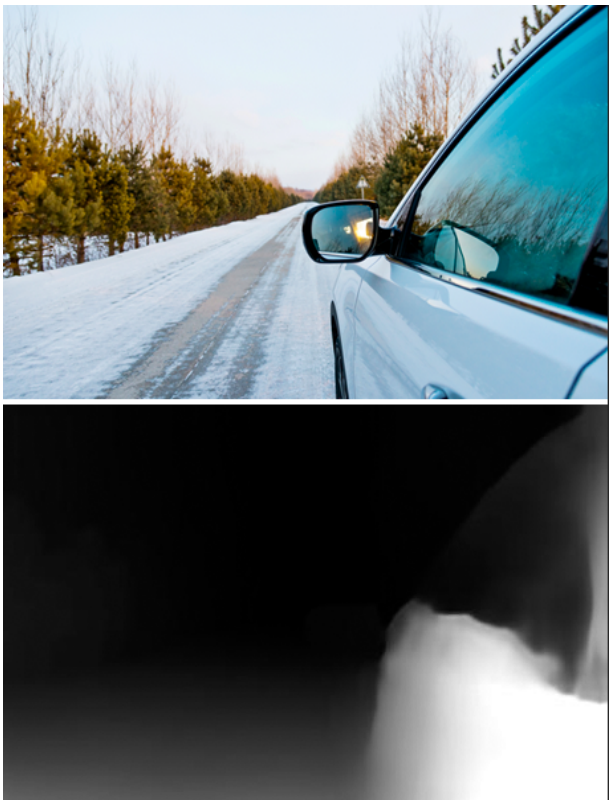

Fig. 4. An example of a failure case where there is a non-Lambertian surface (Side-view mirror of Car)

6 CONTRIBUTIONS

The code that has been developed from scratch is a direct implementation of the existing method. No novel contribution on top of the paper has been made explicitly. We expected to achieve better results after using ResNet50 instead of ResNet18 as the encoder, but the model turned out to be slower and did not produce better results than at hand.

All three of us were equally responsible in implementing the model. However, the portions which we have implemented are as follows: Creating functions for backprojection, smoothing function, compute SSIM loss between pair of images, building ResNet encoder, pose decoder functions, depth decoder and other miscellaneous functions which have aided to us to make it as a complete project on monocular depth estimation.

7 CONCLUSION

In this work, we have implemented a self-supervised model for monocular depth prediction. Instead of using labeled ground truth data which is a challenging task to get, we exploit the relationship of different temporal frames to get the depth parameter. Self-supervised depth estimation frames this learning problem as predicting the appearance of a target image from the viewpoint of another image. By constraining the network using an intermediate variable, it is possible to extract this value from the model in the case of depth or disparity. This problem is inherently ill-posed. Many3D

scenes can project to the same depth per pixel. This problem was addressed by enforcing smoothing in the output depth map. Due to this reason, the outline of the depth of the objects was not very well defined. This model fails to estimate the depth of objects that are either reflective or transparent, as shown in figure 4. Nevertheless, with the addition of i) Per-Pixel Minimum Reprojection Loss ii) Auto-Masking Stationary Pixels iii) Multi-scale Estimation, the model gives good performance if the object is within the range and doesn't contain reflective or transparent surfaces, i.e. do not violate Lambertian surface properties.

REFERENCES

[1] John Flynn, Ivan Neulander, James Philbin, and Noah Snavely. 2016. Deepstereo: Learning to predict new views from the world's imagery. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 5515–5524.

[2] Ravi Garg, Vijay Kumar Bg, Gustavo Carneiro, and Ian Reid. 2016. Unsupervised cnn for single view depth estimation: Geometry to the rescue. In *European conference on computer vision*. Springer, 740–756.

[3] Daniel Scharstein and Richard Szeliski. 2002. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International journal of computer vision* 47, 1 (2002), 7–42.

[4] Jonas Uhrig, Nick Schneider, Lukas Schneider, Uwe Franke, Thomas Brox, and Andreas Geiger. 2017. Sparsity Invariant CNNs. In *International Conference on 3D Vision (3DV)*.

[5] Junyuan Xie, Ross Girshick, and Ali Farhadi. 2016. Deep3d: Fully automatic 2d-to-3d video conversion with deep convolutional neural networks. In *European Conference on Computer Vision*. Springer, 842–857.

[6] Ziyu Zhang, Alexander G Schwing, and Raquel Fidler. 2015. Monocular object instance segmentation and depth ordering with cnns. In *Proceedings of the IEEE International Conference on Computer Vision*. 2614–2622.