Alma Mater Studiorum · University of Bologna

ML4CV Project work

Giovanni Minelli (giovanni.minelli2@studio.unibo.it)

June 8, 2022

1 Introduction

This project concerns the exploitation of self-supervised depth information from a single image as an aid in solving the Novel View Synthesis (NVS) task. Tackling two tasks at a time may, in some cases, be easier if the improvement of one can be beneficial to the other.

Single image depth estimation (SIDE) is a crucial step towards inferring scene geometry from 2D images. The goal is to predict the depth value of each pixel or infer depth information relative to the camera, given only a single RGB image as input. While traditional methods use multi-view geometry to find the relationship between pairs of images, newer methods can directly estimate depth by minimizing regression loss, learning from clues, or in general gaining an higher level understanding of the scene [1].

Novel view synthesis (NVS) is the task of generating new images of a scene with an arbitrary target camera pose given single or multiple source images with corresponding camera poses. NVS has several applications, e.g., it can be used in virtual reality applications where capturing all possible views of real-world scenes is impractical or in general for augmented content generation such as interactive 3D models of products in online shopping websites. With NVS techniques, only a few images need to be captured to provide a complete and seamless experience [2].

In this approach, a single encoder is involved to estimate a latent code from which both depth information and newly generated views of the object can be extracted. A novel training objective enables to learn to perform depth estimation of a single image along with the synthesis of views at arbitrary locations, and then reciprocally transfer improvements from one task to another. Because of the self-supervised approach, the algorithm requires only images with relative transformation poses between views during training, leveraging us from the burden of also obtaining depth ground truth values, which in real-world settings could be challenging: this is considered a major plus point of the overall design.

2 Method

Unlike previous work that approached the same combination of tasks [4, 5, 2], this project aims to get good results on both ends, indeed it does not have a side task that generates additional information for the main one, but uses a pipeline that can generate results to aid and mutually supervise both.

The architecture has a common encoder-decoder structure, with a shared encoder and two decoders assigned to the two different tasks. The encoder is inspired by [5], which uses a transforming auto-encoder (TAE) previously proposed by [3] to learn a meaningful latent code in 3D metric space. The source view is first encoded by obtaining the corresponding latent code z along with intermediate features z^F extracted at multiple levels of the encoding process, which are then used as skip-connections in the decoders. From a starting dimension of 256x256, the input image is convoluted into a one-dimensional tensor and then transformed by applying a geometric transformation. Given a transformation matrix $T_{s\longrightarrow t} = [R|t]_{s\longrightarrow t}$, the latent representation of the source is multiplied in dot product with it to obtain the latent representation of the target view $z_T = T_{s\longrightarrow t} \cdot z_S$.

This transformation works by assuming an equivariant representation learned through the TAE, and, intuitively, training in this way will encourage the latent code to encode 3D position information for low detailed features. The transformed latent code z_T given as input to a view decoder generates images that may suffer from blurriness, lack of texture detail, or identity inconsistency. This problem was also encountered in [2], which solved the problem by aiding the decoder with skip-connections representing encoded mid-features warped to the target position, where the required depth is obtained from a depth decoder also fed with z_T.
Improvements in this approach are aimed at obtaining better depth estimates and ultimately better novel views by introducing a skip-guided depth decoder and refining the overall NVS pipeline. In this work, the depth scales, we will call them $d_T^{F_{unskip}}$, obtained as output of the depth decoder with z_T as input, are used only as an initial estimate with which the dense mid-features z_S^F are transformed through an inverse warp operation.

$$d_T^{F_{unskip}} = DepDec_{unskip}(z_T), \quad z_T^{F_{unskip}} = inv\_warp(z_S^F, d_T^{F_{unskip}})$$

The dense features warped $z_T^{F_{unskip}}$ are now sent to the same depth decoder enhanced by the ability to use additional information at different scales to obtain more detailed depth maps $d_T^{F_{skip}}$. At this point the finer depth scales $d_T^{F_{skip}}$ are again involved in an inverse warp process to obtain the target image features,

$$d_T^{F_{skip}} = DepDec_{skip}(z_T, z_T^{F_{unskip}}), \quad z_T^{F_{skip}} = inv\_warp(z_S^F, d_T^{F_{skip}})$$

and these together with the dense transformed code z_T constitute the input of the NVS decoder.

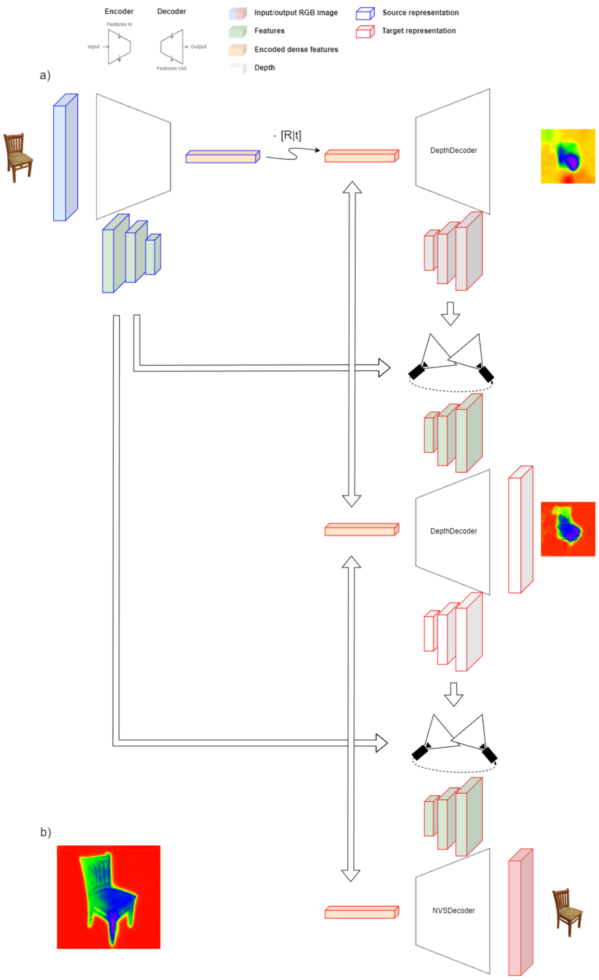

Figure 1: Overview of the training pipeline. (a) shows the data flow with the source image as input, with intermediate depth maps $d^{F_{unskip}}$ and final predictions related to the two tasks; (b) shows the result of the depth decoder fed directly with the target image latent representation.

2.1 View transformation

Having at our disposal the depth relative to the target position d, the intrinsics of the camera K and the relative pose $T_{s \to t}$ (extrinsics), we can find the correspondences into the source view by backprojecting the 3D points of the target view through the homogenous space:

$$[X, Y, Z]^T = d(x_t, y_t) K^{-1} [x_t, y_t, 1]^T$$

$$[x_s, y_s, 1] \sim K T_{s \to t} [X, Y, Z, 1]^T$$

where each target pixel (x_t, y_t) encodes the corresponding position in the source view (x_s, y_s). Since the corresponding pixel coordinate is a continuos coordinate, bilinear sampling is used to obtain the value. This operation propagates texture and local details and also is amenable to backpropagation of gradients derived from erroneous correspondences.

2.2 Architecture details

As encoder I used a pretrained ResNet-18 decomposing the layers to extract the middle features as in [7], but differently from the standard architecture I avoided the use of the last MaxPool layer in favor of a fully

connected network to obtain the final dense representation which resulted in a faster convergence for depth estimations. In detail, a final dimension of 1x1x600 is used.
For the decoders, are involved CNNs with upsampling layers interleaved by convolution blocks inspired by UNet architecture [8], but with some modifications: 3x3 convolutions with reflection padding [7] and LRelu non-linear activation function [2].
The DepthDecoder has the peculiarity of being able to work both with and without the additional 3D features passed as skip connections: as entry point a fully connected layer expands the latent input and that output is reshaped into 512 channels with a spatial resolution of 8x8, then 5 stages follow to upsample the features while halving the depth channels. In the case of skipped connections, additional convolutional blocks of appropriate size are used to include the additional features as well, thus allowing most of the architecture to be reused. It returns depth maps at different scales and these are used to compute the warping of source-encoded mid-features to use as skip-connections in the NVSDecoder. The choice to use these less detailed intermediate depths instead of downscaled versions of the last depth map, was observed to allow the multiscale loss to be propagated more accurately.
The NVSDecoder has a very similar structure to the DepthDecoder, with the difference being the format of the output data.

2.3 The loss functions

The whole model is trained in an end-to-end fashion requiring only a single source image, a target image, and their relative transformations from which a rotation matrix is computed. Since the predicted depth maps influence the view synthesis as well, we can use the target image as only ground truth data and in this way provide a supervision to the encoder and both decoders. Here follows the loss functions used to supervise the different parts involved:

Multiscale reconstruction loss+Photometric reprojection loss: to correct the value of itensities from low texture regions or estimates far from the correct value, a multiscale loss is introduced as L1 loss between pixel values for both decoders. For the NVS decoder, the loss values are obtained by considering the upsampled output layers with ground truth, each weighted according to the importance of its resolution. The total loss is the sum of individual losses at each decoder scale.
For the depth decoder, instead of resizing the depth scales, the images (source and target) are downsampled to the resolution of depth scales in output: these are used to warp the source image at the target location to calculate the loss with the ground truth target image.

VGG perceptual loss: as already demonstrated by [2], the adoption of a VGG perceptual loss enables sharper synthesis results. A pre-trained VGG16 network is used to extract features from generated results and truth images, and the perceptual loss is the sum of feature distances (L1 distance) computed from a number of layers.

Edge-aware smoothness loss: as in [6, 7], to train the depth decoder in an unsupervised manner, I used the edge-aware smoothness loss, which, by calculating the gradient of the prediction and ground truth (both normalized), allows to highlight the edges not well predicted, this makes the predictions be locally smooth over time, thus avoiding holes or discontinuities.

Skip connections loss: to supervise the effects of the skip-connections in the depth decoder, an L1 loss is also introduced between: $DepDec_{unskip}(z_{a2b})$ and $DepDec_{skip}(z_b, z_b^F)$, respectively the initial target depth obtained with the source image transformed latent as input and the depth obtained with the target latent+features, since the initial depth even without additional features should be the best possible; $DepDec_{skip}(z_{a2b}, z_{a2b}^{F_{unskip}})$ and $DepDec_{skip}(z_b, z_b^F)$, respectively the target depth obtained with the source image transformed latent+warped features as input and again the depth obtained with the target image latent+features.

In conclusion, the final loss function is the following

$$L_{tot} = \alpha L_{recon} + \beta L_{VGG} + \gamma L_{smooth} + \delta L_{skip}$$

where $\alpha, \beta, \gamma, \delta$, are weighting hyper parameters properly chosen (see section 3).

2.4 Training process

To train the model, given the many steps and the limits of computational resources at my disposal, I tried some tactics to speed up the process: training the depth decoder first and then, at a later time, unlocking the NVS decoder as well with relative losses saves time by avoiding training the NVS decoder when the provided depth is still very unreliable; alternatively, I tried to decouple the back-propagation of the error derived from the two desired outcomes by learning in parallel to obtain a depth map for the target from the source and to obtain a sharp view with the NVS decoder using ground truth mid-features. In this way, as long as L_{skip} does not decrease below a handpicked threshold, the NVS decoder will not be affected by the performance of the depth decoder.
Using the first mentioned tactic, I performed several Bayes searches to adjust the hyper parameters and select the best performing implementation details. In about 25k+25k steps I was able to obtain a good level of convergence with a marked distinction between the contributions of each variable. As for the second tactic, it is able to provide the best results in terms of depth as early as 8k-10k steps crossing the threshold and regressing to standard training.
To obtain the final evaluation results, I trained the entire model in a standard way for a total of 50 epochs for both datasets.

3 Experiments

To test the proposed pipeline, I ran numerous executions controlled through a Bayes search procedure to determine the best hyper parameters. Using this approach, the best method for image padding was determined in the use of edge pixel reflection as filler values; bilinear sampling as the preferred interpolation method; Adam was used as optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.999$ and a starting learning rate of $1e^{-5}$. In addition to these implementation choices, extensive research led to the choice of weight parameters for losses. All experiments were performed on a cluster of Nvidia RTX 2080Ti, kindly made available to the students by UniBo, and on a Nvidia RTX 3060. According to the results obtained, training executions were limited to 15-20 epochs and a batch size of 8 was used to comply with the memory limits of the GPU. The purpose of the experiments was to determine the effectiveness of the proposed approach, in pair with the set of losses used, especially in comparison to [2] which is, to the best of my knowledge, the best approach in the literature combining SIDE and NVS tasks.

3.1 Dataset

Training and evaluation are limited to a set of synthetic objects, as they are notoriously easier to handle during experiments, nevertheless further extension could consider possible application to real world scenes. In detail, an extended version of ShapeNet[10] was used [11]. Such a dataset provides a large-scale, richly annotated archive of shapes represented by 3D CAD models of objects in different poses determined by camera azimuth and elevation. Depending on the desired difficulty, views of objects may or may not be centered, and for each variation are provided metadata regarding camera poses, albedo, depth, and normal map.
In this works only view-centered objects belonging to the category of cars and chairs were used. The poses were used to calculate the rotation matrix between views and the depth maps as ground truths for calculating metrics. The images used were scaled to 256x256 to match the input dimensions of the model, and the

reference pairs were sampled to have an azimuth difference within the range [-40°,40°]. The same dataset split of [2] was used, with 20088 training samples for chairs and 100832 for cars.

3.2 Metrics

To quantitatively evaluate the expected results obtained in the NVS task, I reported the L1-norm measuring the difference error per pixel (to be minimized) and the structural similarity index (SSIM) [13] indicating the perceptual quality of the image (to be maximized).
To evaluate the DepthDecoder performance I firstly computed the metrics of ground truth depth respect the target image encoded-decoded, but also I monitored constantly the performance of its output with the source-transformed input: $L1(y'^{(T)}, y'^{(S)})$ where $y'^{(T)}$ and $y'^{(S)}$ are the two predictions respectively with target and source view as input.
Here I used the classical metrics of [12], where SILog is used as the main metric. Given the ground truth depth image y with N valid pixels and the predicted depth image y', the metrics are defined as follows:

- square root of the scale invariant logarithmic error (SILog) $(\sqrt{\frac{1}{N}\Sigma_{i=1}^{N} z_i^2 - \frac{1}{N^2}(\Sigma_{i=1}^{N} z_i)^2}) * 100$ where $z_i = log(y'_i) - log(y_i)$ (min)
- absolute relative error % (Abs. Rel.): $\frac{1}{N}\Sigma_{i=1}^{N} \frac{|y_i y'_i|}{y_i}$ (min)
- squared relative error % (Sq. Rel.): $\sqrt{\frac{1}{N}\Sigma_{i=1}^{N} \frac{|y_i - y'_i|^2}{y_i}}$ (min)
- root mean squared error (RMSE): $\sqrt{\frac{1}{N}\Sigma_{i=1}^{N} (y_i - y'_i)^2}$ (min)
- log root mean squared error (Log RMSE): $\sqrt{\frac{1}{N}\Sigma_{i=1}^{N} |log y_i - log y'_i|^2}$ (min)
- threshold accuracy (σ_j): fraction of y_i such that $max(\frac{y_i}{y_i}, \frac{y'_i}{y_i}) = \sigma \leq 1.25^j \quad$ for $j \in \{1, 2, 3\}$ (max)

Methods	SSIM$^{(\uparrow)}$	L1$^{(\downarrow)}$	SILog$^{(\downarrow)}$	Abs Rel$^{(\downarrow)}$	Sq Rel$^{(\downarrow)}$	RMSE$^{(\downarrow)}$	Log RMSE$^{(\downarrow)}$	$\sigma \leq 1.25^{(\uparrow)}$	$\sigma \leq 1.25^{2(\uparrow)}$	$\sigma \leq 1.25^{3(\uparrow)}$
					CHAIR					
Hou [2]	**0.906**	0.136	21.075	0.110	0.035	0.197	0.218	0.818	0.940	0.973
Ours	0.888	**0.075**	**10,100**	**0,067**	**0,011**	**0,114**	**0,101**	**0,956**	**0,992**	**0,999**
					CAR					
Hou [2]	**0.930**	0.109	36.074	0.268	0.100	0.327	0.399	0.496	0.704	0.942
Ours	0,905	**0,059**	**9,629**	**0.037**	**0.010**	**0,110**	**0,097**	**0,952**	**0,988**	**0,997**

Table 1: Evaluation of the method proposed in comparison to [2] with described metrics, both for depth and novel view results.

3.3 Test

The results obtained for method comparison can be found in Tab.1. On both synthetic datasets the model performs quite well in terms of depth, while on the NVS task it has more obvious difficulties when dealing with chair objects that have elongated parts and sharp edges. In addition, in general, the predicted image has a consistent shape but often appears washed out and when complex textures are involved the prediction become more blurred. A limitation of this approach is also the ability to predict correctly the albedo of the object since light effects are carried from one view to another without adaptation to the new pose.
Training the model on images with an higher rotation value makes the NVS prediction be more accurate nevertheless the depths metrics show a decrease of performance. During training the sampled pairs were

formed by considering images with a random degree of rotation in the range [-40°,40°], while for testing only pairs with this maximum value were considered.

The effectiveness of skip connections has been demonstrated in many previous applications and also here the two computed depth maps $d^{F_{unskip}}$ and $d^{F_{skip}}$ are extremely different in terms of the utility they can bring (See Fig.2): while the former remains blurry for many epochs before starting to slightly improve (thanks to the presence of L_{skip}) the latter becomes visually more detailed and relatable to the shape of the source image already after a thousand steps.

In addition, since it is also important to estimate the effectiveness of the other components along with the depth decoder, the similarity between the depth obtained with the original target features and source-transformed ones is tracked to monitor its improvements over time.

About the different values contribuiting to L_{tot}, an ablative study was conducted by training different models end-to-end on the chair dataset (ShapeNet), in which one at a time the losses were dropped. The evaluation results of these models are shown in Tab.2.

The loss which supervise the effects of skip-connections makes the depth results be more consistent when considering the same input object but with different orientations: because depth profoundly affects the creation of the new view, it is important that the predictions of the DepthDecoder remain consistent with each other. With $Loss^{-skip}$ there is a slight increase in the performance of the NVS decoder, while the depth metrics decrease consistenly. Indeed evaluating the results qualitatively, the depth maps appear fuzzier, probably because the NVS-Decoder shapes the maps at his own desire to improve its prediction, in particular $d^{F_{unskip}}$ assume a more cloudy shape. Also effects of VGG loss can be spotted qualitatively: although there is just a slight decrease in the recorded NVS metrics for $Loss^{-vgg}$ model, the generated images are evidently blurrier and faded. Lastly $Loss^{-recon}$ and $Loss^{-smooth}$ show lower performance in the metrics of both tasks, as their supervisory role normally applies to multiple pipeline stages.

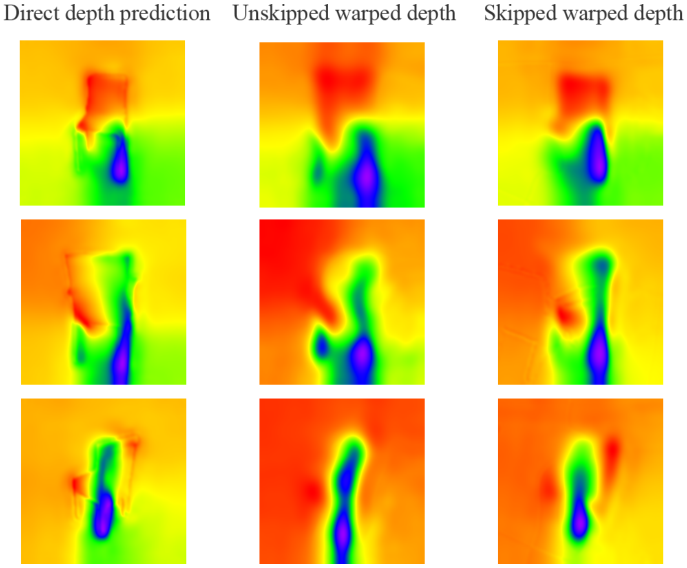

Figure 2: Qualitative comparison of depth predictions: from left to right the results relative to a single sample with the target image latent as input, and both intermediate and final depth maps ($d^{F_{unskip}}$ and $d^{F_{skip}}$) with the source image latent warped to target position as input.

	SSIM$^{(\uparrow)}$	L1$^{(\downarrow)}$	SILog$^{(\downarrow)}$	L1$_{direct}$$^{(\downarrow)}$	Abs Rel$^{(\downarrow)}$	Sq Rel$^{(\downarrow)}$	RMSE$^{(\downarrow)}$	Log RMSE$^{(\downarrow)}$	$\sigma \leq 1.25^{(\uparrow)}$	$\sigma \leq 1.25^{2(\uparrow)}$	$\sigma \leq 1.25^{3(\uparrow)}$
$Loss$	**0.888**	**0.075**	10,100	0,060	0,067	**0,011**	0,114	**0,101**	0,956	**0,992**	0,999
$Loss^{-reco}$	0,854	0,135	9.437	**0,000**	**0,042**	0,015	**0,113**	0,167	**0,974**	0,992	**1,000**
$Loss^{-vgg}$	0,859	0,109	**9.235**	0,054	0,048	0,015	0,183	0,165	0,921	0,976	0,995
$Loss^{-smooth}$	0,848	0,120	9.912	0,092	0,060	0,016	0,179	0,171	0,583	0,969	1,000
$Loss^{-skip}$	0,863	0,103	9.654	0,126	0,053	0,015	0,185	0,171	0,721	0,972	0,993

Table 2: Comparison of results obtained from models with different losses in action. Each model has been trained end-to-end and evaluated on ShapeNet chairs dataset.

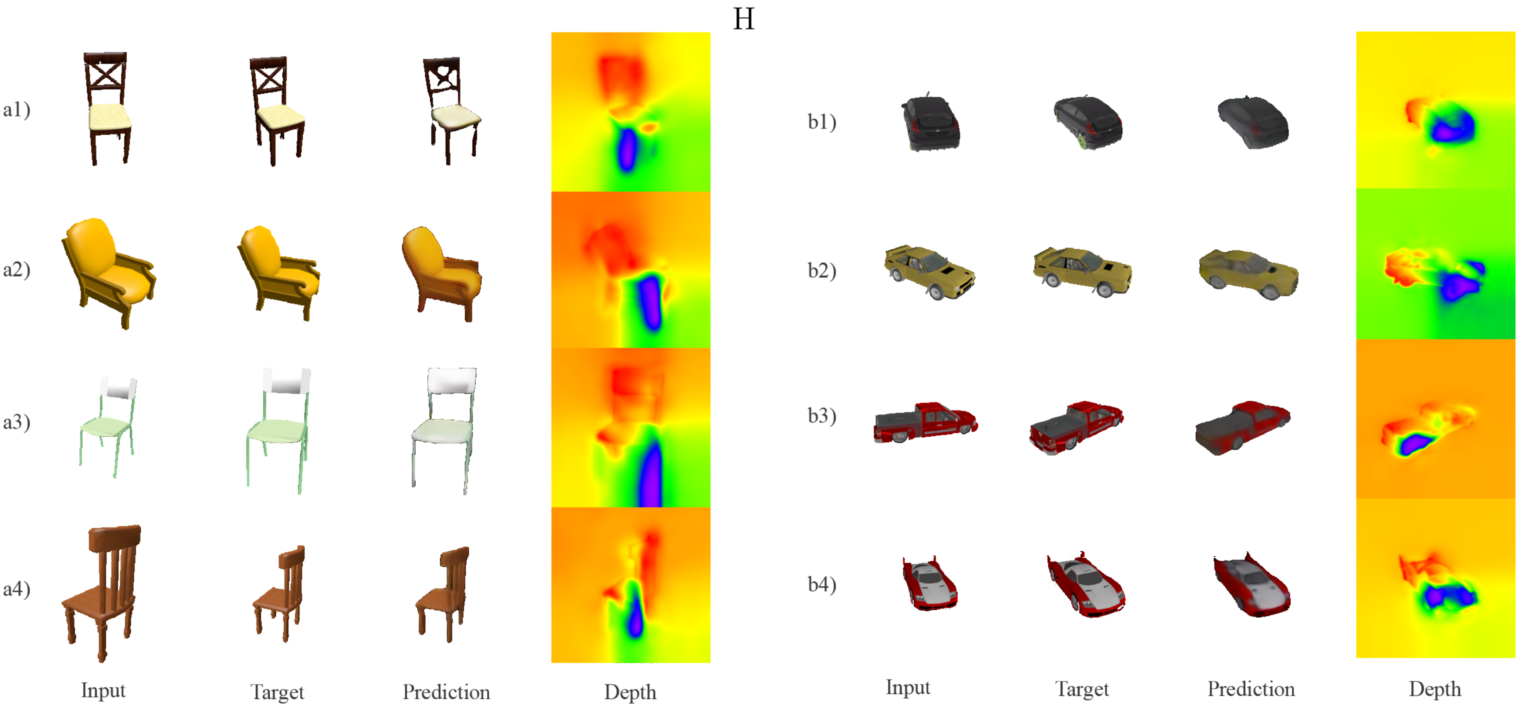

Figure 3: Results on ShapeNet objects. From a qualitative analysis it can be seen that most of the predictions of the NVSDecoder have an almost perfect shape, but color tones and details are more difficult to generate. While the DepthDecoder does an effective job in finding edges and depth changes in objects.

4 Conclusions

In this project I propose an alternative method to solve two widely known problem in the machine vision search field: inference of depth values and novel view generation, having at disposal only a single view of the scene. The key supervisory signal here is the mutual influence of the two tasks to obtain increasingly better predictions, and the final evaluation shows how no ground truth depth is truly necessary to obtain promising results.

The encoder-decoder structure seems to be a winning choice, as it is able to compress most of the information relevant to different tasks into a space of fixed size, and the enhanced decoder with skip-connections was critical for the emergence of details in the predictions and thus for achieving accurate results.

This is a promising approach, even for other combinations of tasks, however there is still room for improvement. As further work I think the loss could be rethought to be compressed into fewer obvious signals for both tasks, and certainly a benchmark on real-world data would be needed in order to validate the result already obtained.

References

[1] Vasiljevic, Igor Kolkin, Nick Zhang, Shanyi Luo, Ruotian Wang, Haochen Dai, Falcon Daniele, Andrea Mostajabi, Mohammadreza Basart, Steven Walter, Matthew Shakhnarovich, Gregory. (2019). DIODE: A Dense Indoor and Outdoor DEpth Dataset.

[2] Hou, Yuxin and Solin, Arno and Kannala, Juho. (2021). Novel View Synthesis via Depth-Guided Skip Connections: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), January, 3119-3128.

[3] Geoffrey E Hinton, Alex Krizhevsky, and Sida D Wang. Transforming auto-encoders. In International conference on artificial neural networks, pages 44–51. Springer, 2011.

[4] Tulsiani, Shubham Tucker, Richard Snavely, Noah. (2018). Layer-Structured 3D Scene Inference via View Synthesis: 15th European Conference, Munich, Germany, September 8–14, 2018, Proceedings, Part VII. 10.1007/978-3-030-01234-2_19.

[5] Xu Chen, Jie Song, Otmar Hilliges; Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 2019, pp. 4090-4100.

[6] Clement Godard, Oisin Mac Aodha, and Gabriel J Bros´tow. Unsupervised monocular depth estimation with leftright consistency. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 270–279, (2017).

[7] Godard, Clément Aodha, Oisin Firman, Michael Brostow, Gabriel. (2019). Digging Into Self-Supervised Monocular Depth Estimation. 10.1109/ICCV.2019.00393.

[8] Olaf Ronneberger, Philipp Fischer, Thomas Brox. U-Net: Convolutional Networks for Biomedical Image Segmentation. In Medical Image Computing and Computer-Assisted Intervention (MICCAI), Springer, LNCS, Vol.9351: 234-241, (2015).

[9] Zhou, Tinghui Brown, Matthew Snavely, Noah Lowe, David. (2017). Unsupervised Learning of Depth and Ego-Motion from Video. 6612-6619. 10.1109/CVPR.2017.700.

[10] Chang, Angel Funkhouser, Thomas Guibas, Leonidas Hanrahan, Pat Huang, Qixing Li, Zimo Savarese, Silvio Savva, Manolis Song, Shuran Su, Hao Xiao, Jianxiong Yi, Li Yu, Fisher. (2015). ShapeNet: An Information-Rich 3D Model Repository.

[11] Xu, Qiangeng and Wang, Weiyue and Ceylan, Duygu and Mech, Radomir and Neumann, Ulrich (2019). DISN: Deep Implicit Surface Network for High-quality Single-view 3D Reconstruction. NeurIPS.

[12] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. In Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2, NIPS'14, pages 2366–2374, Cambridge, MA, USA, 2014. MIT Press.

[13] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility tostructural similarity. IEEE transactions on image processing, 13(4):600–612, 2004.