Monocular-Depth-Estimation
The influence of Image Augmentation on state-of-the-art depth estimators

Leonardo Pohl
l.b.pohl@umail.leidenuniv.nl
Universiteit Leiden, Leiden Institute of Advanced Computer Science
Leiden, Netherlands

Abstract

Monocular-Depth-Estimation is a highly researched field in the area of computer vision. The goal of Monocular-Depth-Estimation is to extract the depth information of a single RGB image. In this report we investigate two different methods which achieve just that, Global-Local Path Networks for Monocular Depth Estimation with Vertical CutDepth (GLPN) and Neural Window Fully-connected CRFs for Monocular Depth Estimation. Both algorithms are in the top five of state-of-the-art methods on the two most commonly used datasets, KITTI Eigen Split and NYU-Depth V2 Dataset. The KITTI-Dataset consist of frames of a video captured by cameras mounted on a car. The ground truth is determined using a 360° Velodyne laser scanner. The NYU-Depth V2 Dataset consists of frames of videos captured by a Kinect camera indoors. To evaluate the performance of any given model we calculate the most commonly use metrics in the field. Additionally to comparing the models on their own we also investigate how different image augmentation methods influence the depth estimation. In our experiments we found that it is important to recheck that the depth of the predicted image matches the depth of the ground truth.

Keywords: Monocular Depth Estimation, Computer Vision, Comparison of algorithms, Image Processing, GLPN, NeWCRFs

1 Introduction

As a human it is sometimes hard to understand that some feats that seam trivial for us are very hard for computers to do. For example, even without binocular vision, which humans somehow magically stitch together to one coherent image while ignoring the nose, which is in the field of vision of both, it is possible to easily and quite reliably estimate the distance of what we see. This task is essential for human survival, or the survival of any animal, which might be eaten by a lion running towards them. Therefore, evolution has improved this quality of the human brain, or presumably any animals' brain, to perfection in the last couple million years.

However, the human nowadays is quite impatient and cannot wait a couple million years so that computers might evolves to achieve such a feat by themselves. Therefore, one heavily researched area of computer vision focuses on Depth Estimation of images, to be precise with Monocular Depth estimation.

In this report, we will focus on achieving this feat without using millions of years of evolution and instead using different state-of-the-art methods which are contrary to the years of evolution usable by computers. The goal of this project is to compare two very recent methods which are in the top five on both the NYU-Depth V2 dataset and the KITTI dataset, both of which will be described in detail later in this report.

In addition to the comparison, we will introduce image augmentation to test both if the models are robust enough to handle different image conditions such as desaturation or oversaturation or if possibly these augmentation steps might improve the estimation quality.

First, we will introduce the two different estimators and explain why these were selected. Afterwards, we will describe the datasets on which we have tested both estimators and thereafter, the metrics with which, we have evaluated the experiments. Next, we will summarise the different augmentation algorithms we have investigated in our experiments. Finally, we will share the results of our experiments using the augmentation methods, metrics and datasets which have been introduced prior.

2 Estimators

In this section we will introduce the different predictors that have been examined in the scope of this project. All selected predictors are state of the art methods which have only recently been developed. Both selected methods are part of the top five for both the KITTI dataset as well as the NYU dataset, the most commonly used datasets for monocular depth estimation, we will focus on the datasets and the comparison metrics more in a later section.

Additionally to the selected algorithms others were regarded, but later discarded due to difficulties in the implementation. These also seemed to be quite promising but alas this was not within the scope of this project.

The plan was to utilise the very promising Monocular-Depth-Estimation Toolbox which was introduced by Zhenyu et.al. in [Li et al. 2022a]. A multitude of algorithms are available for comparison, including Depthformer [Li et al. 2022b], which introduced the toolbox and BinsFormer [Li et al. 2022c] which was also introduced by Zhenyu et.al, both of which are included in the top three based on the results

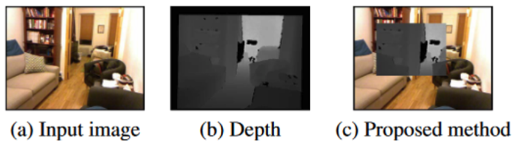


Figure 1. An example for Cutdepth applied on training data

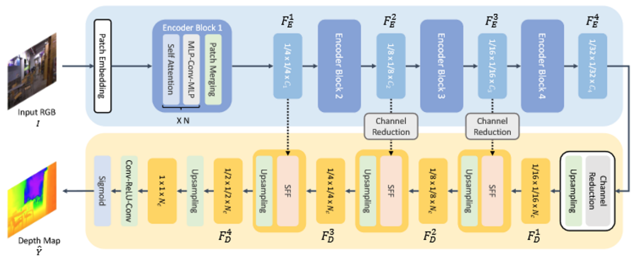


Figure 2. The general architecture of a GLPN.

of on the KITTI dataset and the NYU dataset as can be seen on [Paperswithcode 2022].

2.1 GLPN: Global-Local Path Networks for Monocular Depth Estimation with Vertical CutDepth

The first estimator we focus on is Global-Local Path Networks (GLPN) [Kim et al. 2022] which was introduced by Kin et.al. in 2022. It extracts global and local features and creates paths which are passed into various layers of CNNs.

To increase the size of the training dataset a data augmentation for depth images is applied, CutDepth [Ishii and Yamashita 2021]. In CutDepth part of the depth image is cut into the training image, as can be seen in Figure 1.

Like a lot of depth estimators, this one is also heavily influenced by Big to Small: Multi-Scale Local Planar Guidance for Monocular Depth Estimation [Lee et al. 2019].

The architecture of the network can be seen in Figure 2. First the global and local features are extracted and paths are formed, then various levels of CNNs are applied until finally a sigmoid funtion is applied to extract the Depth Map.

2.2 NeWCRFs: Neural Window Fully-connected CRFs for Monocular Depth Estimation

Secondly we chose Neural Window Fully-connected CRFs (NeWCRFs) [Yuan et al. 2022] which was introduced by Yuan et.al in 2022 in. It uses Conditional Random Fields(CRF) to regress the depth map. CRF were introduced in [Zheng et al. 2015], however, plain CRF are not very performant and do not scale well to the size of images. Therefore, the image is split into windows and fully connected conditional random fields (FC-CRFs) are applied, these were introduced in [Zhang 2018].

The general structure of NeWCRFs can be seen in Figure 3. In the encoder the features of the image are extracted and

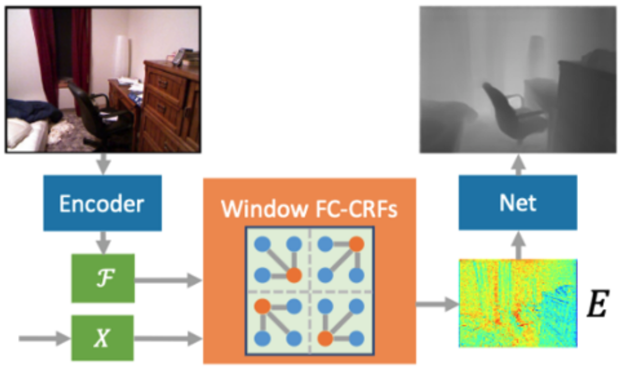


Figure 3. General structure of NeWCRFs.

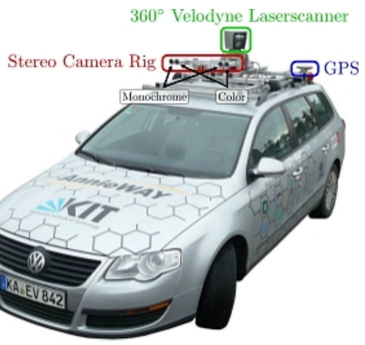


Figure 4. Sensor setup on a Passat which was used to collect the data for the KITTI dataset

the image is split into windows, then FC-CRFs are applied and finally the depth image is extracted.

3 Datasets

For our experiment we picked the two most used datasets for Monocular Depth Estimation; the KITTI Eigen split dataset and the NYU-Depth V2 dataset.

3.1 KITTI Eigen split Dataset

The KITTI Eigen split was introduced by the Karlsruhe Institute of Technology and Toyota Technological Institute (KITTI).[Geiger et al. 2012] The dataset consists of 200 gigabytes worth of images captured by two cameras mounted on a car driving on different days on a distance of around 39 kilometres.

The ground truth for this dataset was captured by a 360° Velodyne Laserscanner, mounted next to the camera. An image of the setup can be seen in Figure 4. An example of the ground truth can be seen in Figure 5. Due to the nature of the Velodyne camera, the depth image does not completely cover every area.

In some evaluation algorithms these points were set to a specific value, mostly 0. However, we find that this influences

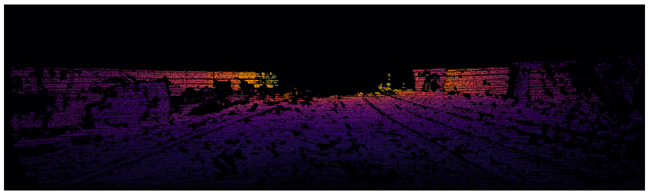


Figure 5. A ground truth sample of the KITTI dataset captured using a 360° Velodyne Laserscanner

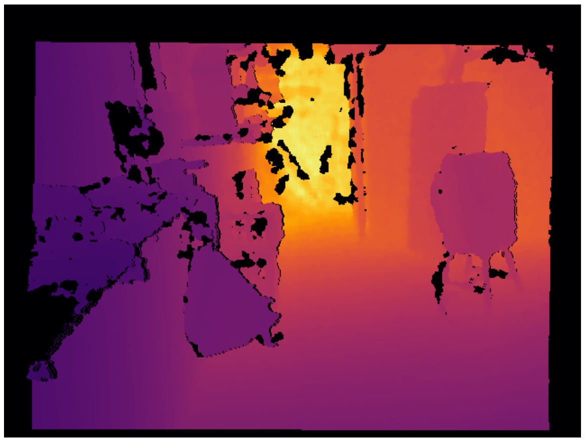


Figure 6. Sample ground truth of a scene in the NYU-Depth V2 dataset recorded with a Kinect camera

the results and therefore, we chose to only regard the points of the predicted image, where a specific depth is determined.

3.2 NYU-Depth V2 dataset

The second dataset consists of frames of videos taken in various indoor environments.[Nathan Silberman and Fergus 2012] The data was recorded using both the depth and colour sensor of a Microsoft Kinect camera. An example of the depth ground truth can be seen in Figure 6.

As opposed to the Velodyne Laserscanner used for the KITTI dataset, which only captures stripes of the scene, the Kinect covers a more coherent area of depth pixels. Nonetheless, some areas, especially reflective surfaces, translucent objects and edges of object may create areas which are not captured by the camera. These areas are visible in figure 6 as black spots. Identically to the KITTI dataset evaluation we ignore these areas when comparing the estimated depth image with the ground truth.

4 Metrics

To evaluate the performance of the depth estimation different metrics are picked. Not every metric gives a fair overview of how well one predictor performs. A trivial approach would be to compare the ground truth pixel by pixel and calculate the error, but for some cases this is not the most reliable metric.

In this section we will discuss the metrics that have been used to evaluate the performance of each predictor. As mentioned before, since the ground truth is missing some values, those will not be considered in our evaluation.

In this section we will elaborate the different metrics that were used to compare both algorithms. These metrics are widely used in this field of research and therefore offer a good comparison to other state of the art methods.

4.1 Root mean squared error

As mentioned above the most straight forward metric is a straight comparison. This is done by subtracting both images (the Ground Truth (gt) and the Estimated image ($pred$)) calculating the mean of all subtracted pixels and calculating the square root. This can be seen in Equation 1.

$$RMSE = \sqrt{mean((gt - pred)^2)} \tag{1}$$

4.2 Relative squared and absolute error

The relative squared error and the absolute squared error are calculated by dividing the absolute and squared difference of the ground truth respectively by the ground truth. This shows the certainty of the values. The calculation of the relative squared error can be seen in Equation 2 and the relative absolute error can be seen in Equation 3.

$$sq\_rel = mean(\frac{(gt - pred)^2}{gt}) \tag{2}$$

$$abs\_rel = mean(\frac{abs(gt - pred)}{gt}) \tag{3}$$

4.3 δ_1, δ_2 and δ_3

The different magnitudes of 1.25 are regarded depending on the subscript of δ. The calculation of the different δ_n can be seen in Equations 4 - 6 with $threshold = max(\frac{gt}{pred}, \frac{pred}{gt})$.

$$\delta_1 = mean(threshold < 1.25) \tag{4}$$

$$\delta_2 = mean(threshold < 1.25^2) \tag{5}$$

$$\delta_3 = mean(threshold < 1.25^3) \tag{6}$$

4.4 Scale-Invariant Error

The Scale-Invariant error has been introduced by Eigen et.al. in [Eigen et al. 2014]. It represents . The equation to calculate the Scale-Invariant Error can be seen in Equation 7.

$$silog = \sqrt{mean((log(gt) - log(pred))^2) - mean(log(gt) - log(pred))^2} \tag{7}$$

Table 1. 3x3 Laplacian Kernel used to calculate edge detection for all images.

-1	-1	-1
-1	8	-1
-1	-1	-1

5 Image Augmentation methods

The goal of this project is to analyse the influence of different augmentation methods on monocular depth estimation. In this section we will focus on which methods have been picked and why they might yield different results.

The different augmentation methods are shown in Figure **??** where the sample image can be seen on the left and the different augmentation methods can be seen on the right.

5.1 Grey-scale

The image was turned into a grey-scale version of itself within the Rec. 609 colour space, that is to say, that the grey value was calculated by multiplying the red value with 0.299, the blue value with 0.587 and the green value with 0.114.

Since the models were trained to accept 3 channel RGB images the image had to be transferred to RGB after calculation of the grey value.

5.2 Saturation

The saturation was both increased and decreased by a factor of two, i.e. the saturation for some experiments was 0.5 and of others was 2. This was calculated using the build in image enhancer in the python pillow library which was used throughout the project.

5.3 Edge Detection

The edge detection is done using a 3x3 Laplacian kernel who's structure can be seen in Table 1. This kernel was applied on a grey-scale image to derive with an image only containing the edges that have been detected.

5.4 Edge Detection in combination with the source image

Finally we overlayed the finished edge detection image over the coloured and grey-scale image. We created two different instances of the image, one where any value of the edge detection was regarded if the value value is larger than 5 and one where the value of the pixel is larger than 50, i.e. grey value $> 2\%$ and $> 20\%$ respectively.

6 Experiments

All experiments were run on a single computer. The specifications can be seen in Table 2. Retrospectively, it would have been easier to have run the experiments on the provided servers of the university of Leiden. This would, additionally to allowing the usage of said computer while running the experiments for multiple days, allow for the usage of Cuda cores, which are present on the server but not on the machine the experiments were run on. The presence of these Cuda cores would have also simplified the use of the existing methods, since there would not have been a need to rewrite the parts, which required them.

Table 2. Technical Specifications of the computer on which all experiments were run.

	Component
CPU	AMD®Ryzen 5 2600x six-core processor
GPU	GeForce RTX 2070 SUPER
Memory	16 GB DDR4 2400Hz

In the following section we will summarise the experiments that have been conducted to compare the two estimators mentioned above and what effect augmentation of the images have on the depth estimation.

6.1 GLPN vs. NeWCFs

Firstly, we execute depth estimation of both algorithms without any preprocessed images on both datasets. The results of this can be seen in Table 8a.

Sadly the calculation of the errors was incorrect, since the depth of the ground truth and the predicted depth do not align. This was noticed too late and therefore there was no time left to redo the experiments.

The results of the NYU-Depth V2 dataset can be seen in Table **??**.

Due to the calculation errors it is hard to get information about the performance of either estimator. However, the δ values are correctly calculated. We observe that NeWCFs outperforms GLPN on the KITTI dataset for every δ. However on the NYU-Depth dataset GLPN outperforms NeWCFs.

6.2 Comparison of the different image augmentation methods

As mentioned before, the calculation of the metrics was incorrect due to differences in the ground truth data and the predicted data. However, when comparing the method with itself using different augmentation methods we do not have the issue of needing an exact match since the relative error of each calculated error is still correct.

6.3 Time consumption

In the results of this report and also on a lot of the papers introducing the methods the time consumption was not taken into account. All experiments were executed over the span of multiple weeks, due to the large size of the dataset and the limited resources of that computer that was used. However, it is important to mention that NeW CFRs took a lot longer than GLPN. While NeW CFRs had an average of around

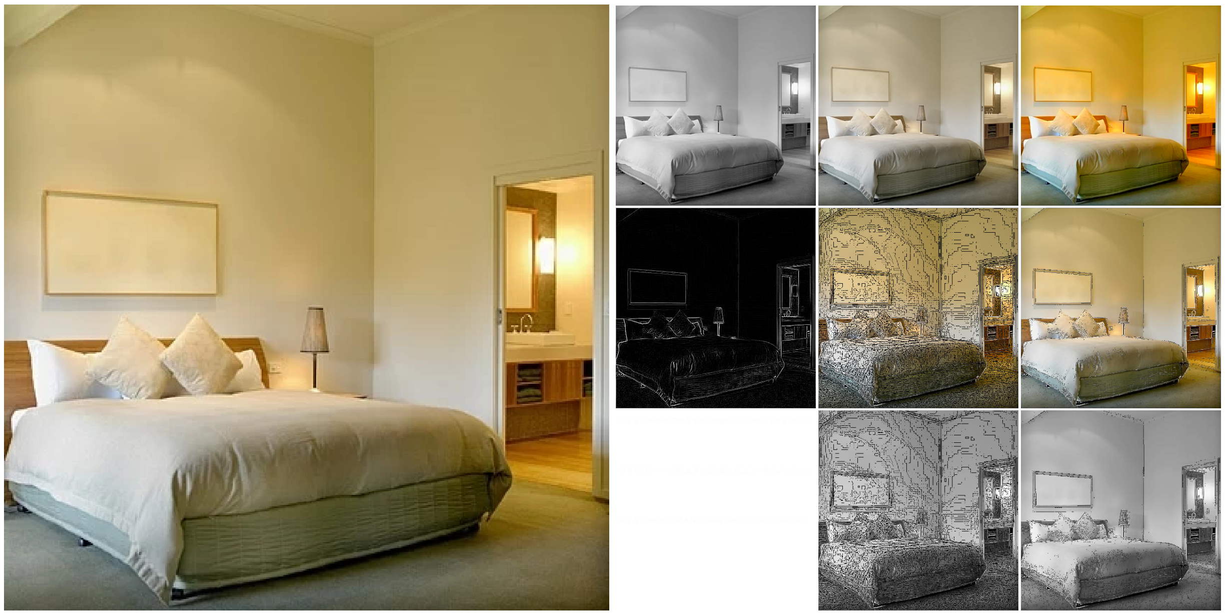
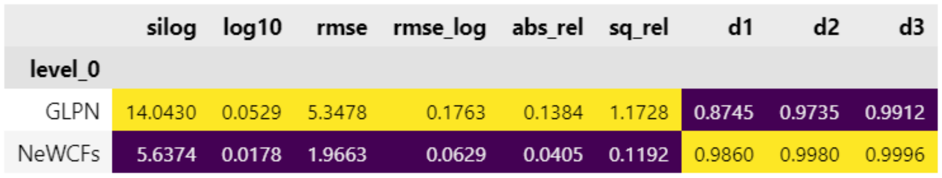
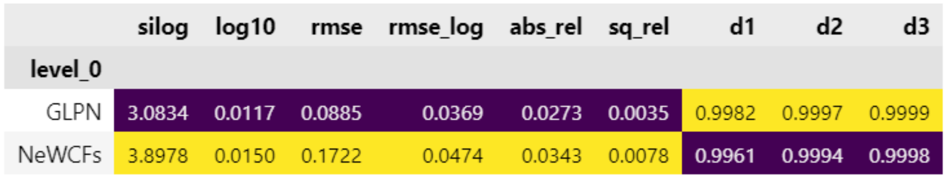


Figure 7. Different Data augmentation methods applied on the sample image which can be seen on the left. The images were augmented from left to write int the following ways: (a) Greyscaled, (b) Desaturated, (c) Oversaturated, (d) Edge Detection, (e) Colour Image overlayed with edges with $> 2\%$ white value, (f) Colour Image overlayed with edges with $> 20\%$ white value, (g) Greyscaled Image overlayed with edges with $> 2\%$ white value and (h) Greyscaled Image overlayed with edges with $> 20\%$ white value



	silog	log10	rmse	rmse_log	abs_rel	sq_rel	d1	d2	d3
level_0									
GLPN	14.0430	0.0529	5.3478	0.1763	0.1384	1.1728	0.8745	0.9735	0.9912
NeWCFs	5.6374	0.0178	1.9663	0.0629	0.0405	0.1192	0.9860	0.9980	0.9996

(a) Experiment results of GLPN vs. NeWCFs on the KITTI Eigen Dataset.



	silog	log10	rmse	rmse_log	abs_rel	sq_rel	d1	d2	d3
level_0									
GLPN	3.0834	0.0117	0.0885	0.0369	0.0273	0.0035	0.9982	0.9997	0.9999
NeWCFs	3.8978	0.0150	0.1722	0.0474	0.0343	0.0078	0.9961	0.9994	0.9998

(b) Experiment results of GLPN vs. NeWCFs on the NYU-Depth V2 Dataset.

Figure 8. The results of the experiments without image augmentation

$1.5\frac{frames}{seconds}$, GLPN averaged at around $5\frac{frames}{seconds}$, which is a large speed up considering the number of images that were processed.

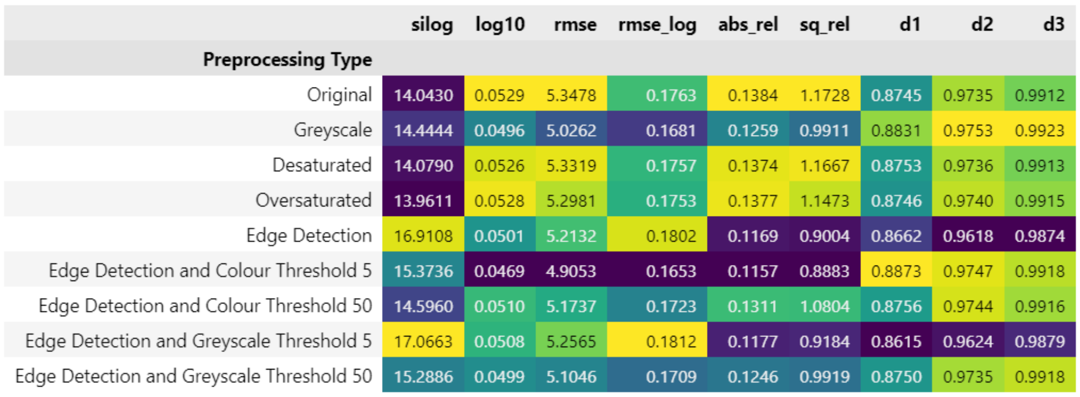
7 Discussion

In this section we will discuss the results of the experiments. Furthermore we will note some issues with the current datasets.

7.1 GLPN vs. NeWCFs

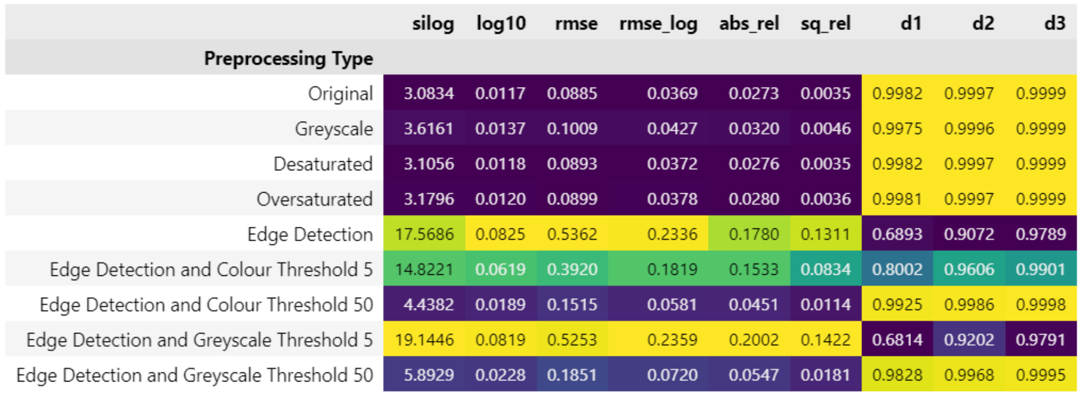
Due to some issues in the final stage of the evaluation our results cannot be concisely compared. Though it seems that we achieved similar results as [Paperswithcode 2022], i.e., NeWCFs outperforms GLPN.

However, a metric that is not mentioned on [Paperswithcode 2022] is the time consumption. NeWCFs takes a lot longer calculate than GLPN. This might be due to GLPN being intendet to be run on a CPU, whereas, the default implementation of NeWCFs as it is found in the Git repository



Preprocessing Type	silog	log10	rmse	rmse_log	abs_rel	sq_rel	d1	d2	d3
Original	14.0430	0.0529	5.3478	0.1763	0.1384	1.1728	0.8745	0.9735	0.9912
Greyscale	14.4444	0.0496	5.0262	0.1681	0.1259	0.9911	0.8831	0.9753	0.9923
Desaturated	14.0790	0.0526	5.3319	0.1757	0.1374	1.1667	0.8753	0.9736	0.9913
Oversaturated	13.9611	0.0528	5.2981	0.1753	0.1377	1.1473	0.8746	0.9740	0.9915
Edge Detection	16.9108	0.0501	5.2132	0.1802	0.1169	0.9004	0.8662	0.9618	0.9874
Edge Detection and Colour Threshold 5	15.3736	0.0469	4.9053	0.1653	0.1157	0.8883	0.8873	0.9747	0.9918
Edge Detection and Colour Threshold 50	14.5960	0.0510	5.1737	0.1723	0.1311	1.0804	0.8756	0.9744	0.9916
Edge Detection and Greyscale Threshold 5	17.0663	0.0508	5.2565	0.1812	0.1177	0.9184	0.8615	0.9624	0.9879
Edge Detection and Greyscale Threshold 50	15.2886	0.0499	5.1046	0.1709	0.1246	0.9919	0.8750	0.9735	0.9918

(a) Results of the experiments run on the KITTI Eigen Dataset



Preprocessing Type	silog	log10	rmse	rmse_log	abs_rel	sq_rel	d1	d2	d3
Original	3.0834	0.0117	0.0885	0.0369	0.0273	0.0035	0.9982	0.9997	0.9999
Greyscale	3.6161	0.0137	0.1009	0.0427	0.0320	0.0046	0.9975	0.9996	0.9999
Desaturated	3.1056	0.0118	0.0893	0.0372	0.0276	0.0035	0.9982	0.9997	0.9999
Oversaturated	3.1796	0.0120	0.0899	0.0378	0.0280	0.0036	0.9981	0.9997	0.9999
Edge Detection	17.5686	0.0825	0.5362	0.2336	0.1780	0.1311	0.6893	0.9072	0.9789
Edge Detection and Colour Threshold 5	14.8221	0.0619	0.3920	0.1819	0.1533	0.0834	0.8002	0.9606	0.9901
Edge Detection and Colour Threshold 50	4.4382	0.0189	0.1515	0.0581	0.0451	0.0114	0.9925	0.9986	0.9998
Edge Detection and Greyscale Threshold 5	19.1446	0.0819	0.5253	0.2359	0.2002	0.1422	0.6814	0.9202	0.9791
Edge Detection and Greyscale Threshold 50	5.8929	0.0228	0.1851	0.0720	0.0547	0.0181	0.9828	0.9968	0.9995

(b) Results of the experiments run on the NYU-Depth V2 Dataset

Figure 9. The results of the experiments with image augmentation GLPN

accompanying [Yuan et al. 2022] the only implementation uses a GPU.

7.2 Experiments with Preprocessing

The results that were achieved here also have to be taken with a grain of salt since the exact values are incorrect. However, since the relative values still keep meaning we are able to get arrive at some conclusions which we will discuss in the following.

7.2.1 Greyscale and Saturation. Saturation had little effect on the results of GLPN when estimating the depth of the KITTI dataset. It is worth noting that the depth estimation with an oversaturated image outperforms the original image across the board. This difference is however quite marginal. It seems that for this algorithm some colour preparation was done before the feature extraction.

For all cases, using a greyscale image decreased the accuracy, but only by a negligable amount. If this proves to be replicable then the storage of these datasets might be cut down to a factor of 3 by only storing the grey value while still achieving a fairly accurate image.

7.2.2 Edge Detection. It comes as no surprise that just passing edge detection as an image does not result in very good depth detection, it is however surprising that is performs relatively well given the fact that it is a harder task even as humans to grasp the depth of images just including the edges of objects.

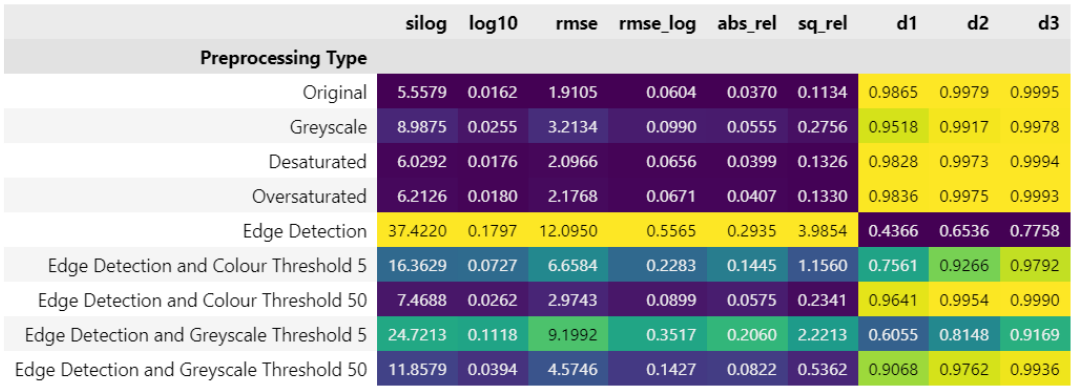
We believe that the reason for this "soft" failure might be that edge detection is part in a lot of feature extraction process and therefore does not have as much influence on the final image.

While including the image and the greyscale version of the image show an improvement over using soley the edge detection, it does not improve the message and no specific

7.3 Ground Truth Datasets

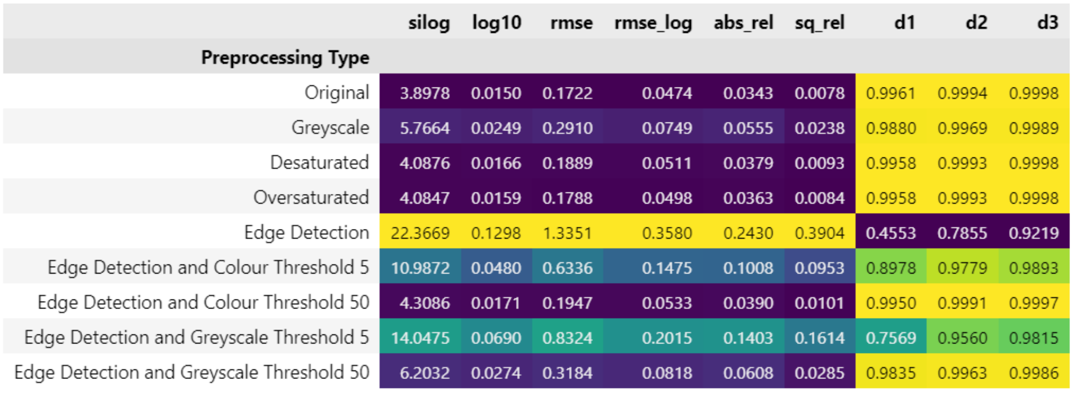
In this report we have seen the use of two datasets. Both of which are flawed in two major factors. The extend of these flaws are impossible to repair and also pose questions which need to be answered.

Firstly, the position of the ground truth detector, for example the Velodyne laser or the Kinect IR sensor, are not mounted at the exact position where the RGB Camera is. As



Preprocessing Type	silog	log10	rmse	rmse_log	abs_rel	sq_rel	d1	d2	d3
Original	5.5579	0.0162	1.9105	0.0604	0.0370	0.1134	0.9865	0.9979	0.9995
Greyscale	8.9875	0.0255	3.2134	0.0990	0.0555	0.2756	0.9518	0.9917	0.9978
Desaturated	6.0292	0.0176	2.0966	0.0656	0.0399	0.1326	0.9828	0.9973	0.9994
Oversaturated	6.2126	0.0180	2.1768	0.0671	0.0407	0.1330	0.9836	0.9975	0.9993
Edge Detection	37.4220	0.1797	12.0950	0.5565	0.2935	3.9854	0.4366	0.6536	0.7758
Edge Detection and Colour Threshold 5	16.3629	0.0727	6.6584	0.2283	0.1445	1.1560	0.7561	0.9266	0.9792
Edge Detection and Colour Threshold 50	7.4688	0.0262	2.9743	0.0899	0.0575	0.2341	0.9641	0.9954	0.9990
Edge Detection and Greyscale Threshold 5	24.7213	0.1118	9.1992	0.3517	0.2060	2.2213	0.6055	0.8148	0.9169
Edge Detection and Greyscale Threshold 50	11.8579	0.0394	4.5746	0.1427	0.0822	0.5362	0.9068	0.9762	0.9936

(a) Results of the experiments run on the KITTI Eigen Dataset



Preprocessing Type	silog	log10	rmse	rmse_log	abs_rel	sq_rel	d1	d2	d3
Original	3.8978	0.0150	0.1722	0.0474	0.0343	0.0078	0.9961	0.9994	0.9998
Greyscale	5.7664	0.0249	0.2910	0.0749	0.0555	0.0238	0.9880	0.9969	0.9989
Desaturated	4.0876	0.0166	0.1889	0.0511	0.0379	0.0093	0.9958	0.9993	0.9998
Oversaturated	4.0847	0.0159	0.1788	0.0498	0.0363	0.0084	0.9958	0.9993	0.9998
Edge Detection	22.3669	0.1298	1.3351	0.3580	0.2430	0.3904	0.4553	0.7855	0.9219
Edge Detection and Colour Threshold 5	10.9872	0.0480	0.6336	0.1475	0.1008	0.0953	0.8978	0.9779	0.9893
Edge Detection and Colour Threshold 50	4.3086	0.0171	0.1947	0.0533	0.0390	0.0101	0.9950	0.9991	0.9997
Edge Detection and Greyscale Threshold 5	14.0475	0.0690	0.8324	0.2015	0.1403	0.1614	0.7569	0.9560	0.9815
Edge Detection and Greyscale Threshold 50	6.2032	0.0274	0.3184	0.0818	0.0608	0.0285	0.9835	0.9963	0.9986

(b) Results of the experiments run on the NYU-Depth V2 Dataset

Figure 10. The results of the experiments with image augmentation NeWCFs

mentioned before this is impossible to repair. There is currently no way of simultaneously capture depth images and RGB images at the same time at the same location. Unless it is captured at different times and even then the camera settings, such as the aperture size or the field of view or in case of certain RGB cameras the depth of field.

The extend of all the mentioned discrepancies is hard to measure. The error might be marginal however it might also be the case that models trained on such a dataset are slightly skewed when applied on a different setup. One possible artefact of this non matching of ground truth and RGB data is actually visible in the top part of Figure 11. In the KITTI dataset, for example, the upper part is not covered by the Velodyne Laser. However, whether this is caused by the ground truth or not is hard to tell.

The second issue of datasets becomes visible in Figure 12. In this sample of the NYU-Depth V2 Dataset a bathroom scene is visible, which contains a mirror. In real life this is considered as a flat surface, so the correct depth would be a flat surface, but even humans are sometimes incapable to detect mirrors as flat surfaces and not as extended spaces. Furthermore, it is also hard for depth sensors to detect mirrors or transparent surfaces since light might be reflected off

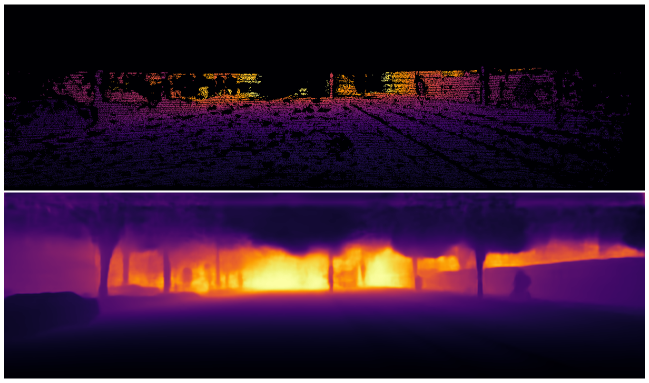


Figure 11. Artefact in Monocular Depth Estimation which might be caused by missing ground truth.

of them and in the case of mirrors capture the surface that is visible through the mirror and might not be captured at all since the light has been scattered immeasurable due to the reflection on the glass or the scattering after the light exits the glass. However, capturing this is not possible with current depth cameras since they rely on light. Unless a physical

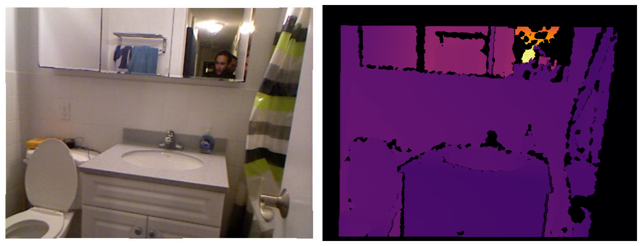


Figure 12. A question that arises with depth estimation and depth capturing are mirrors or transparent surfaces to be considered as flat surfaces or as an extended space or nothing respectively.

probe is used which is highly impractical and also seemingly dangerous.

The question now arises what is desirable, should Monocular Depth Estimation simulate the real world, or what is capturable by depth cameras.

7.4 Applicability of Monocular-Depth-Estimation

While the results are promising it is still important to note that monocular depth estimation is not yet deployable for real time depth estimation due to the very processing time given a fairly decent computer in the experiments. For a lot of the experiments the computer ran at 100% CPU utilisation with a high power consumption and is therefore not applicable for example in robotics where usually microprocessors with less ram are employed and running on battery power.

For those scenarios it is hard to see the use of Monocular-Depth-Estimation when things such as time of flight cameras exist, which also reliably estimate the depth. However, in scenarios where the depth needs to be extracted after an image was taken and there are no time limits then the results presented in this report are quite promising.

8 Conclusion

In this project we compared two state-of-the-art methods for Monocular-Depth-Estimation, Global-Local Path Networks for Monocular Depth Estimation with Vertical CutDepth and Neural Window Fully-connected CRFs for Monocular Depth Estimation. At the time of writing, both of these methods are in the top five for both introduce datasets according to [Paperswithcode 2022].

We have miscalculated the errors which makes coming to conclusions hard. it seems that our results were in line with the results shown on paperswithcode, with NeWCFs outperforming GLPN.

Additionally to standard experiments we have applied different filters and processing methods to the input images. We changed the saturation of the image and also turned it into a greyscale image. Furthermore, we applied edge detection and used that as an input image to be detected and also overlayed the RGB image and the grey-scale image with the derived edges at different thresholds.

We found that GLPN handles data augmentation better than NeWCFs. Moreover, using pure Edge detection does not limit the ability to perform monocular depth detection as badly as was expected, but it surely did not help. Including an underlying image with the edge detection proved to improve the results but did not yield any overall improvements.

Finally we discussed potential flaws in the current datasets.

References

David Eigen, Christian Puhrsch, and Rob Fergus. 2014. Depth Map Prediction from a Single Image using a Multi-Scale Deep Network. https://doi.org/10.48550/ARXIV.1406.2283

Andreas Geiger, Philip Lenz, and Raquel Urtasun. 2012. Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*.

Yasunori Ishii and Takayoshi Yamashita. 2021. CutDepth:Edge-aware Data Augmentation in Depth Estimation. https://doi.org/10.48550/ARXIV.2107.07684

Doyeon Kim, Woonghyun Ga, Pyungwhan Ahn, Donggyu Joo, Sehwan Chun, and Junmo Kim. 2022. Global-Local Path Networks for Monocular Depth Estimation with Vertical CutDepth. https://doi.org/10.48550/ARXIV.2201.07436

Jin Han Lee, Myung-Kyu Han, Dong Wook Ko, and Il Hong Suh. 2019. From Big to Small: Multi-Scale Local Planar Guidance for Monocular Depth Estimation. (2019). https://doi.org/10.48550/ARXIV.1907.10326

Zhenyu Li, Zehui Chen, Xianming Liu, and Junjun Jiang. 2022a. DepthFormer: Exploiting Long-Range Correlation and Local Information for Accurate Monocular Depth Estimation. *arXiv preprint arXiv:2203.14211* (2022).

Zhenyu Li, Zehui Chen, Xianming Liu, and Junjun Jiang. 2022b. DepthFormer: Exploiting Long-Range Correlation and Local Information for Accurate Monocular Depth Estimation. https://doi.org/10.48550/ARXIV.2203.14211

Zhenyu Li, Xuyang Wang, Xianming Liu, and Junjun Jiang. 2022c. BinsFormer: Revisiting Adaptive Bins for Monocular Depth Estimation. https://doi.org/10.48550/ARXIV.2204.00987

Pushmeet Kohli Nathan Silberman, Derek Hoiem and Rob Fergus. 2012. Indoor Segmentation and Support Inference from RGBD Images. In *ECCV*.

Paperswithcode. 2022. Monocular-Depth-Estimation. https://paperswithcode.com/task/monocular-depth-estimation

Weihao Yuan, Xiaodong Gu, Zuozhuo Dai, Siyu Zhu, and Ping Tan. 2022. NeW CRFs: Neural Window Fully-connected CRFs for Monocular Depth Estimation. https://doi.org/10.48550/ARXIV.2203.01502

Bin Zhang. 2018. Fully Connected Conditional Random Fields for High-Resolution Remote Sensing Land Use/Land Cover Classification with Convolutional Neural Networks. *Remote Sensing* 10 (11 2018). https://doi.org/10.3390/rs10121889

Shuai Zheng, Sadeep Jayasumana, Bernardino Romera-Paredes, Vibhav Vineet, Zhizhong Su, Dalong Du, Chang Huang, and Philip Torr. 2015. Conditional Random Fields as Recurrent Neural Networks. (02 2015).