TECHNISCHE UNIVERSITÄT
BERGAKADEMIE FREIBERG
The University of Resources. Since 1765.

Fakultät für Mathematik und Informatik
Institut für Informatik

Project thesis

Monocular Depth Estimation on Artificially Scaled Up Camera Images for Robot Applications

Nico Zumpe, Johannes Kohl

Robotik (Diplom)

Matriculation register: 62 351,
64 751

August 14, 2023

Tutor/First Proofreader:
Dr. Sandra Lorenz

In the field of robotics it is still somewhat of a challenge to analyze objects and scenes in camera images that are far away from the robot. This might not be problematic for many stationary and/or slow moving platforms, but it is easy so see what issues may arise when the speed increases or timing in general gets critical.

Combating this problem can be done, for example, by changing the lens of the camera, which in turn worsens the field of view (FOV).
Another approach would be to employ a higher resolution camera. The problem here is that more bandwidth, as well as higher computing power would be needed to transport and process those images. So every advance into more depth clarity comes with its own tradeoffs.

Therefore a more cost effective solution is needed. For this goal, the "higher resolution" approach may still be a viable option by using AI upscaling. The idea is to use a supersampling model to enhance certain regions of interest in the source image and analyze them further. An option for further evaluation would be monocular depth estimation to get a grasp on what a distant area roughly looks like.

The results of this project show that even today the quality of the depth maps produced this way is decent enough to be useful. Although the processing times indicate that an application in real time robotics scenarios is not feasible yet.

1 Introduction

A person operating a motor vehicle always needs to be aware of their surroundings and the environment. They know that the car they saw coming in the background is not gone just because the turning car in front of them is blocking it. This phenomena is called object permanency and is something the majority of humans learn growing up. That same concept is not hard for a computer to grasp either but what might be is the preamble of "detecting the car in the background". Robots rely heavily on images and depth information to orient themselves in their environments but this has some challenges, such as lighting conditions, camera resolution, framerate, etc. There are some options to combat this, but they either come with their own trade offs or may not be applicable in all situations.

Changing to a camera with a thicker lens will improve the zoom, but since the image size stays the same the overall field of view (FOV) is decreased. This may affect close by detection and analysis. As a fix for the above method or a standalone approach it might be useful to change recording resolution to something greater. But also this approach has drawbacks as it requires way more bandwidth and computing power to process those larger images. Another option to combat the lens problem might be varifocal lenses but they add complexity, as well as cost and might not be easily applicable in stereo camera setups since both lenses would need to be well synchronized.

A comparatively low cost solution arises in image superresolution implemented as an AI supersampling model. Pairing this up with monocular depth estimation, a powerful methodology may be born. With this concept no extra camera hardware would be needed. This will be tested in this project in the context of robotics applications in particular.

2 Background

Since we merge and use two aspects of image processing in this project, image super-resolution and depth estimation, we also highlight both aspects separately. The merging process will be explained later in the implementation chapter.

2.1 Super-resolution

Super-resolution is an active research topic, which focuses on recovering a high-resolution image from a low-resolution image. In the original paper [1], the main goal is to achieve better overall perceptual quality for super-resolution. In the paper used in this algorithm the basic architecture of SRResNet is used.

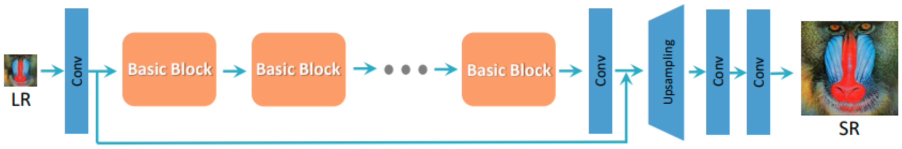

Fig. 1: ESRGAN architecture. [1]

To improve the quality of the recovered image by the super-resolution, there are two modifications made to the structure of the generator. Firstly all Batch Normalization layers are removed. Secondly the replacement of the original basic blocks with the Residual-in-Residual Dense Blocks (RRDB). Those combine multi-level residual networks and dense connections.

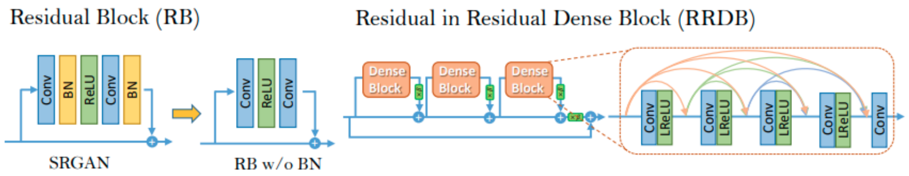

Fig. 2: Improvements to the architecture. [1]

The discriminator based on the Relativistic GAN is also enhanced. The relativistic discriminator tries to predict the probability that a real image is relatively more realistic than a fake one. Specifically in [1] they replaced the standard discriminator with the relativistic average Discriminator RaD.

Also a more effective perceptual loss is developed. Perceptual loss is defined on the activation layers of a pre-trained deep network. In that Network the distance between two activated features is minimized. It is changed so the features are used before the activation layers. A more suitable perceptual loss is developed based on a fine-tuned VGG network for material recognition. It focuses more on textures rather than objects, because the researchers in [1] believe that perceptual loss focused on texture ist critical for super-resolution.

Because there is undesired noise in GAN-based methods, a flexible and effective strategy is proposed which is called network interpolation. First a network oriented on peak signal to noise ratio is trained and from that a GAN-based network is obtained by fine-tuning. All corresponding parameters of the two networks are interpolated to get a new model. This interpolated model has two advantages. First it is able to produce meaningful results for any usable interpolation parameter without introducing artifacts. Second it is possible to continuously balance perceptual quality without the need of re-training the model.

2.2 Monocular Depth Estimation

Depth sensing is traditionally done utilizing multiple lenses with a fixed and known distance and orientation. This concept is based on observations of nature and is also the way human eyes work. Apart from this, there are not many methods of spatial vision. One of those other methods is monocular depth estimation where a depth image gets guessed from a single perspective by a neural network. Initial works in this area of research are for example [2] [3] [4] [5]. Attempts were made to utilize Markov random fields and simpler versions of current concepts like DNN feature encoding.

Through time many implementations and methods for optimization were developed. The relatively recently [6] proposed Global-Local Path Networks consists of an encoder and decoder following a global path with skip connections running into “Selective Feature Fusion” (SFF) modules for local features. The general structure can be seen in figure 3.

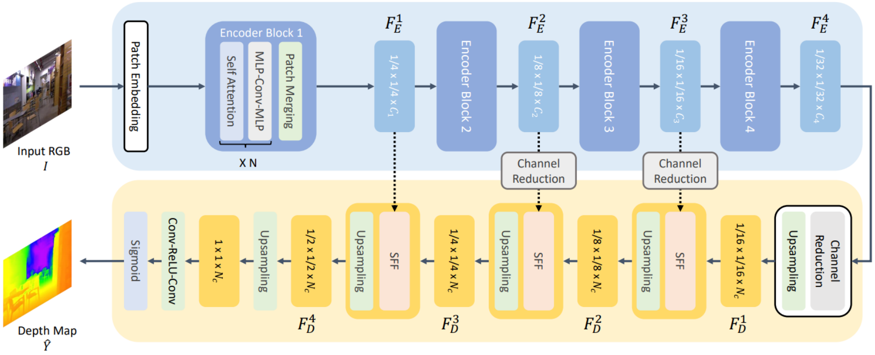

Fig. 3: GLPDepth architecture [6]

The Encoder consists of 4 (encoding) feature stages with an encoding block each. Between stages the dimensions get reduced by scales $\frac{1}{4}$, $\frac{1}{8}$, $\frac{1}{16}$, $\frac{1}{32}$. This is done to reduce computational overhead. The first 3 stages also host skip connections into the decoder which will be talked about later.

The Decoder itself follows a similar but reversed structure as the encoder. It scales the dimensions back up to $\frac{1}{16}$, $\frac{1}{8}$, $\frac{1}{4}$ and $\frac{1}{2}$ the original dimension size. In Between those upsampling steps are so-called SFF modules which merge global with local features from the skip connections. Their specific construction is as depicted in figure 4.

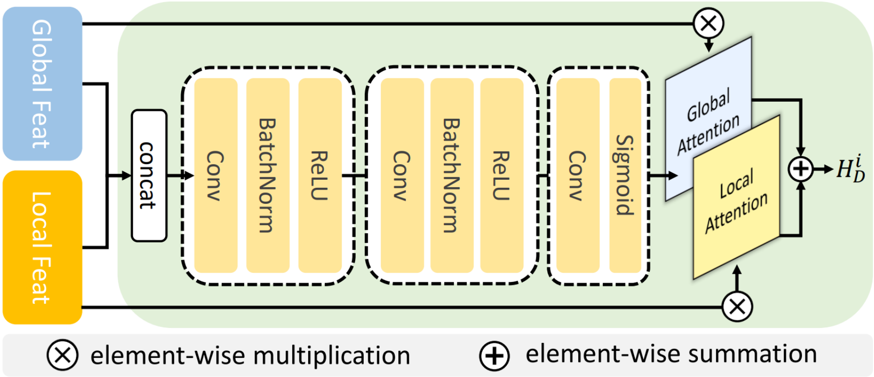

Fig. 4: SFF architecture [6]

According to the researchers, a big performance boost for GLPDepth was also a data augmentation method they called Vertical CutDepth. It is an offspring of the CutDepth method [7] which cuts a region of interest (ROI) from the source image to create a “new” datapoint. The difference with Vertical CutDepth is that the ROI only takes a random horizontal position and width but always remains at $y = 0$ and spans the full height of the input image. This is due to findings [8] in previous depth estimation networks which suggest that they focus more on vertical then horizontal features.

3 Implementation

3.1 Peak Signal Noise Ratio

The signal to noise ratio is a common tool to get a measurement of how much a signal is influenced by random noise. It is calculated by weighing the power of the signal itself against the power of the noise.

$$SNR = \frac{P_{\text{Signal}}}{P_{\text{Noise}}} \tag{1}$$

As images can be seen as discrete signals in two dimensions this concept can also be applied. To avoid evaluating every pixel individually the peak signal to noise ratio (PSNR) can be used which represents the maximum over the signal to noise ratio. In order to calculate it the mean squared error (MSE) over the difference between the image in question and the original image is needed. Typically the PSNR is calculated logarithmically to accommodate a wide range of MSEs.

$$PSNR = 10 * \log_{10}(\frac{R^2}{MSE})[\text{dB}] \tag{2}$$

$$\begin{aligned} \text{with: } & R = 255 \ldots \text{maximum fluctuation in the input image data type} \\ & MSE \ldots \text{mean squared error} \end{aligned}$$

As a general rule of thumb a higher PSNR means a better result and a better quality image. A note for this implementation: When the MSE equates to zero -1 is returned as the PSNR would be ∞ which would make further calculations difficult.

3.2 Monocular Depth Estimation

To lay a baseline for the accuracy of the GLPDepth model we compared it to a ground truth and calculated the MSE, PSNR in Dezibel and computation time. To do this we utilized svo files generated by a Stereolabs ZED2i stereo camera. Those files contain the video streams of both lenses, as well as of all other sensors. In our case the videostream had a resolution of 1920x1080 and was recorded with a framerate of 15fps. The accompanying ZED SDK was used to calculate the depth images from that stereo camera setup as the ground truth. As will be discussed later, this is not necessarily optimal but sufficient. The logged data gets temporarily stored in a numpy array and later written to an output csv file along with the frame index and time. The median and mean values over the aforementioned critical values get calculated in post, just before saving.

3.3 super-resolution

The implementation is kept simple. First the given raw-images get scaled down by 4 using the resize method of OpenCV.

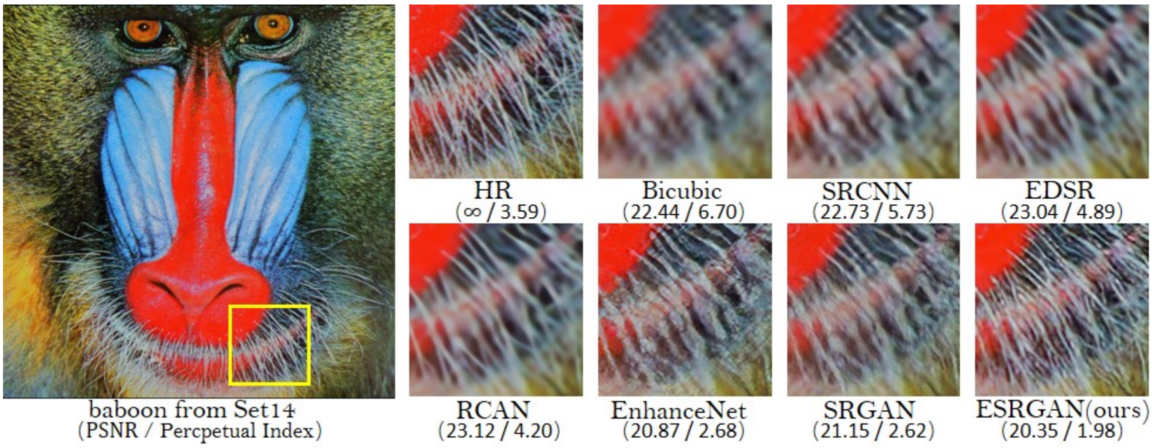

Fig. 5: Comparison of the used super-resolution image to other models. [9]

Then the pretrained model gets loaded which can be obtained in the README of the ESRGAN repository and yields results as shown in figure 5. As can be seen it compares quite well against other super-resolution models. In the repository are two models ready to use, the RRDB_ESRGAN_x4.pth and the RRDB_PSNR_x4.pth. We tested them both and stayed with the ESRGAN model. After loading the model with the parameters, the low resolution images get applied.

Lastly the PSNR is calculated over the raw images and the results after the process.

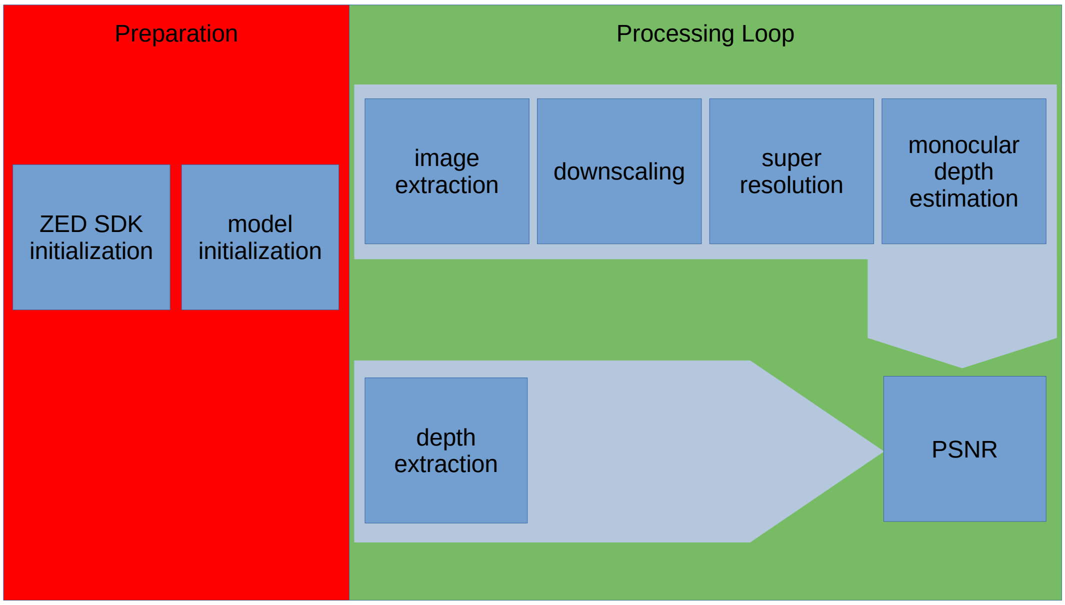

Fig. 6: Flowchart of the final implementation.

3.4 Combination

The actual implementation is very similar to the baseline implementation (Chapter 3.2) in that there is a ground truth extracted from the given stereo information. But since the fine-grained information which would be the goal of the extraction with the super-resolution is not available via this method the effect is approximated by first downscaling the left camera image by a factor of 4 using OpenCV and subsequent upscaling via the super-resolution model. This upscaled image is then used as an input for the monocular depth estimation model to produce a depth image which in turn is compared against the ground truth. Again MSE and PSNR are used to determine the quality of the generated prediction. The time it took to process the images in the neural networks is also logged. Those evaluations are then stored and later saved to a csv file along with the mean and median values of the processing time, MSE and PSNR for evaluation.

The max depth is taken from the parameters used for the svo recording, which in this case is a range from 0.3m to 20.0m. The initialization values for the GLPDepth model are taken from the original implementations repository to accommodate the respective pretrained weights. The latter can also be downloaded as stated in the repositories README. Only nyudepthv2_swin_large.ckpt and kitti_swin_large.ckpt are needed.

The two pretrained weights available to choose from differentiate in the dataset used for training. One was trained on the KITTI Eigen Split while the other one was trained on Nyu Depth V2. These datasets are different from each other in the sense that the former covers primarily the outdoors while the latter contains inside references.

All images are cut or resized to a size of 1216x352 which is rather arbitrary and thus just adopted from the original implementation. This size could be some other size that fulfills the rule of being divisible by 4 without a remainder in both directions. This constraint is necessary since the code does not explicitly check for mismatches between the ground truths size and the resulting depth estimations size resulting from a potential violation of this rule.

4 Experimental Evaluation

4.1 Monocular Depth Estimation

	Nyu Depth V2		KITTI	
	cropped	resized	cropped	resized
execution time [s]	0.524	0.528	0.299	0.306
MSE	10.334	30.742	1.761	0.478
PSNR [dB]	38.001	33.258	45.864	51.761

Tab. 1: Results of GLPDepth alone. Mean and median did not differ much, so only the mean is depicted.

One preparation step was to lay a baseline of how good the depth estimation performs in real world robotics applications. To do this the same pipeline was used as for the final algorithm (See chapter 4.3) except for the downscaling and super-resolution steps in between. By doing this it is possible to get a perspective on a possible negative impact from the super-resolution. It also makes it possible to roughly estimate this impact. Furthermore it enables comparisons against conventional methods and discussions about the practicability and useability of this application. The results are displayed in figure 1.

In the data various trends are visible. Firstly it is apparent that KITTI seems to perform better across the line on our data which checks out with the closeness to the automotive sector that robotics has. Secondly it seems that preserving the whole image by scaling it down instead of cropping it helps the algorithm to make better predictions. This is undoubtedly related to findings in the context of Vertical CutDepth [7].

For reference it may be noted that a PSNR in the range of 30 to 50dB is roughly equivalent to lossy image compression [10].

4.2 super-resolution

Before merging the super-resolution with the monocular depth estimation we wanted to provide some data and check how good the algorithm works with random images. In our notebook we used 10 raw images with different resolutions. After downscaling and upscaling with the super-resolution algorithm, the next step was to use the PSNR as a metric of how well the process worked. The results are displayed in figure 2. The PSNR averages to 29.637dB.

Images	SPQ	Bench	Buffalo	Street Sign	KKB
PSNR [dB]	29.84	28.26	30.42	28.85	29.53
Images	Garden	Frog	Dog	Forecourt	Katze
PSNR [dB]	28.29	29.28	31.26	29.37	31.27

Tab. 2: Results of ESRGAN alone.

4.3 Combination

Figure 6 gives a short recap on the whole process. First an image and depth map get extracted from the original video footage. Next they are scaled to accommodate the requirements of the GLPDepth model. The image is then downsized even further to a sixteenth of its original size and restored via the ESRGAN super-resolution model. On this result the depth estimation generates an additional depth map which in turn gets compared to the ground truth extracted from the video stream. This comparison is evaluated via MSE and PSNR. The whole process from resizing to producing the depth map estimation is also timed for evaluation. At this point it is worth mentioning that the size reduction is also caught in the timing but its influence on the final value should be negligible. The program also provides median and mean values for the PSNR, MSE and execution timing. Table 3 shows the results from the test data with cropping or scaling on KITTI or Nyu Depth V2.

With respect to the results from the evaluation of the GLPDepth and ESRGAN model it would not be unreasonable to suspect that combining both methods results in immense MSE values and a therefore bad image quality reflecting in the PSNR. But

	Nyu Depth V2		KITTI	
	cropped	resized	cropped	resized
execution time [s]	0.786	0.711	0.466	0.468
MSE	9.472	30.884	1.534	0.645
PSNR [dB]	38.449	33.238	46.558	50.395

Tab. 3: Results of GLPDepth and ESRGAN together.

contrary this assumption the actual evaluation results depicted in figure 3 indicate no significant worsening of the depth map quality against the ground truth. The only noticeable difference is to be seen in the execution time which increases over the board by about 0.2s. However this is to be expected since there are extra execution steps involved.

Although all of this seems to be quite positive, there are still things to consider. One major point is that, as can be seen in the ”cropped” columns of the evaluation tables 1 and 3, it makes a grave difference when images are cropped, since this removes context. This plays a large role in regards to the fact that the originally intended use case behaves more like cropping, then resizing. Super sampling a certain region of interest from a camera input to focus on distant objects, is in fact an operation on a cropped image.

Another point to keep in mind is the ground truth, which is not ideal as mentioned before. This can be seen in figure 7 where there are multiple spots with invalid values like NaN or ±∞ depicted in full white. All those points need to be discarded in order to calculate meaningful values.

Overall the combination of both methods appears to be reasonably well suited for at least rough approximations of distant objects' depth maps. Further evaluation is definitely needed to test this in more granularity. One thing that may impede with plans to use this on a live robotic platform is the processing time since execution times of 0.5s only yield a framerate of 2fps which is absolutely not sufficient for high speed or time critical applications. This gets especially obvious when comparing this to the performance of the ground truth calculation which in our case happened 15 times a second and was only constrained by the hardware backing it up which in turn is way less powerful then what was used for this evaluation.

From this the last critical point gets apparent. The tests performed here were executed on a machine equipped with an AMD Ryzen 7 3700x 8-core processor, 16GB of RAM and a Nvidia RTX 2070 graphics card. For comparison the svo files were

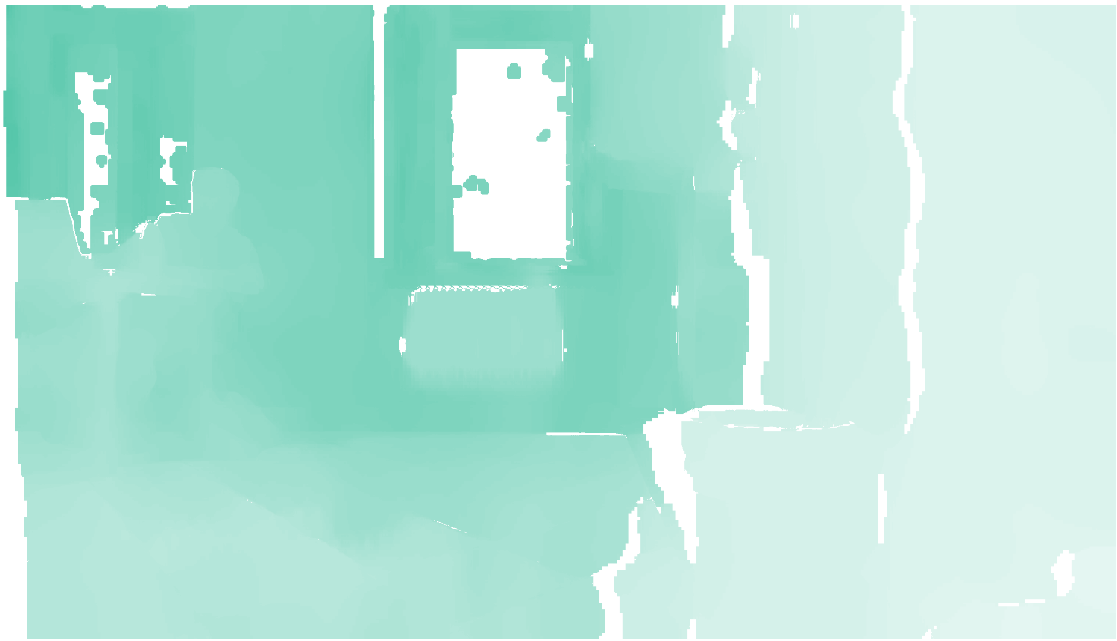

Fig. 7: Problematic ground truth example

recorded on a Nvidia Jetson Xavier NX board with a Nvidia Carmel ARM v8.2 6-core processor, 16GB of RAM and a Nvidia Volta GPU which simultaneously managed two stereo cameras. This Jetson board is not at all as powerful as the system the tests were performed on, so it should be kept in mind that the execution times may vary and stray upwards even more, further decreasing the real time usability of this concept.

5 Conclusion

In conclusion, adding both methods together worked far better than initially anticipated. A trend is definitely visible, that in the future this method of extracting more information from an image can be explored further. For example instead of monocular depth estimation, other algorithms could be used to gather information from supersampled regions of interest, like object detection, segmentation, etc.

With the current state, employing this methodology in real time robotic applications seems unreasonable since the low processing rate and high computational demand are not well suited for critical infrastructure like wayfinding, collision detection and other similar tasks. Still, it may be useful for offline post-processing tasks that do not demand strict time limits and can be built with more powerful hardware.

In the future this concept can be tested again in a similar manner with newer iterations of the used implementations or even extensions in the pipeline to attempt to make it more robust and efficient. For more significant results these tests can also be performed again after eliminating some of the assumptions and problems discussed in the previous chapter.

Erklärung der Urheberschaft

Wir erklären hiermit an Eides statt, dass wir die vorliegende Arbeit ohne Hilfe Dritter und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt haben; die aus fremden Quellen direkt oder indirekt übernommenen Gedanken sind als solche kenntlich gemacht. Die Arbeit wurde bisher in gleicher oder ähnlicher Form in keiner anderen Prüfungsbehörde vorgelegt und auch noch nicht veröffentlicht.

Freiberg, 14.08.2023	Kohl	
Ort, Datum	Unterschrift Johannes Kohl	Unterschrift Nico Zumpe

References

[1] X. Wang, K. Yu, S. Wu, J. Gu, Y. Liu, C. Dong, Y. Qiao, and C. C. Loy, "Esrgan: Enhanced super-resolution generative adversarial networks," in *The European Conference on Computer Vision Workshops (ECCVW)*, September 2018.

[2] A. Saxena, M. Sun, and A. Y. Ng, "Make3d: Learning 3d scene structure from a single still image," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 5, pp. 824–840, 2009.

[3] D. Eigen, C. Puhrsch, and R. Fergus, "Depth map prediction from a single image using a multi-scale deep network," *CoRR*, vol. abs/1406.2283, 2014. [Online]. Available: http://arxiv.org/abs/1406.2283

[4] L. Huynh, P. Nguyen-Ha, J. Matas, E. Rahtu, and J. Heikkilä, "Guiding monocular depth estimation using depth-attention volume," *CoRR*, vol. abs/2004.02760, 2020. [Online]. Available: https://arxiv.org/abs/2004.02760

[5] W. Yin, Y. Liu, C. Shen, and Y. Yan, "Enforcing geometric constraints of virtual normal for depth prediction," *CoRR*, vol. abs/1907.12209, 2019. [Online]. Available: http://arxiv.org/abs/1907.12209

[6] D. Kim, W. Ga, P. Ahn, D. Joo, S. Chun, and J. Kim, "Global-local path networks for monocular depth estimation with vertical cutdepth," *arXiv preprint arXiv:2201.07436*, 2022.

[7] Y. Ishii and T. Yamashita, "Cutdepth: Edge-aware data augmentation in depth estimation," *CoRR*, vol. abs/2107.07684, 2021. [Online]. Available: https://arxiv.org/abs/2107.07684

[8] T. van Dijk and G. C. H. E. de Croon, "How do neural networks see depth in single images?" *CoRR*, vol. abs/1905.07005, 2019. [Online]. Available: http://arxiv.org/abs/1905.07005

[9] X. Wang, "Qualitative comparison 01," https://github.com/xinntao/ESRGAN, 2018.

[10] N. Chervyakov, P. Lyakhov, and N. Nagornov, "Analysis of the quantization noise in discrete wavelet transform filters for 3d medical imaging," *Applied Sciences*, vol. 10, no. 4, 2020. [Online]. Available: https://www.mdpi.com/2076-3417/10/4/1223