POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Executive Summary of the Thesis

Depth Map Super-Resolution Fusing Color Information

Laurea Magistrale in Computer Science Engineering - Ingegneria Informatica

Author: Davide Palesano

Advisor: Prof. Marco Marcon

Co-advisor: Marco Paracchini

Academic year: 2022-2023

1. Introduction

In the last decades the importance of depth information has grown in many different applications of computer vision. Self driving cars need depth information to generate 3D models of their environment so they can avoid obstacles and support the recognition of specific objects and the ability of tracking them. There are to ways digital devices deal with depth information: depth map and 3D point cloud. The former is a two dimensional matrix with only one channel of information related to the distance of an interest object or a scene from a viewpoint. Point cloud models a collection of individual points plotted in 3D space, each one containing several measurements: coordinates along the X, Y, and Z-axes, color value stored in RGB format and luminance, which determines how bright the point is. The oldest method of 3D depth sensing is the passive stereo cameras which operate by calculating the disparity of pixels from two sensors working in sync. However, passive technology had the disadvantage that these cameras cannot be used in the dark. To overcome this issue, active stereo vision technology is used. This technology uses an IR pattern projector to illuminate the scene, which improves the performance in low light and works well on scenes with objects of less complex texture. However, stereo cameras usually are not able to offer large depth ranges (in the range of 10 meters) and do not perform well if object texture is uneven. These problems could be overcame using different technologies, such as LIDAR (Light Detection and Ranging). LiDAR uses the light detection technique to calculate depth. It measures the time it takes for each laser pulse to bounce back from an obstacle. This pulsed laser measurement is used to create 3D models (also known as a point cloud) and maps of objects and environments. Obstacle detection requires low latency to react fast and, in many cases, a wider FoV. For vehicles moving at high speed, a Lidar sensor is commonly used. Since LiDAR generates millions of data points in real-time, it easily creates a precise map of its changing surroundings for safe navigation of autonomous vehicles. Also, the distance accuracy of LiDAR allows the vehicle's system to identify objects in a wide variety of weather and lighting conditions. Many high end depth estimation systems,though, could be very expensive and depending of the application their usage could not be available. The main contribution of this thesis is to develop a combination of cheaper low resolution depth sensors, high resolution traditional cameras and Convolution Neural Network(CNN). This configuration goes

under the name of Super Resolution. We supply high resolution RGB image of the same scene of depthmap to the network in order to deal with the high frequency details of the scene and recognize more easily different objects. We are going to show how we built and trained a Multiple Input Super Resolution (MISR) CNN fusing color information.

2. State of the Art

Human beings use information from both eyes to create a single visual image. This ability to converge information from both eyes is called binocular vision. Each eye sees slightly different spatial information and transmits these differences to the brain. The field of view, which is the area that you can see when you close one eye, overlaps significantly between each eye also. The brain then uses the discrepancies between the two eyes to judge distance and depth. Depth perception is technically called stereopsis or stereoscopic vision. There are variety of depthmap acquisition digital system, passive stereo vision and active stereo vision such as structured light(SL) and time of light(ToF) technologies, each with its pros and cons. Stereo vision is a machine vision technique that can provide full field of view 3D measurements using two or more machine vision cameras. The foundation of stereo vision is similar to 3D perception in human vision and is based on triangulation of rays from multiple viewpoints. It achieves real-time depth perception by using two sensors a set distance apart to triangulate similar pixels from both 2D planes, Figure 1

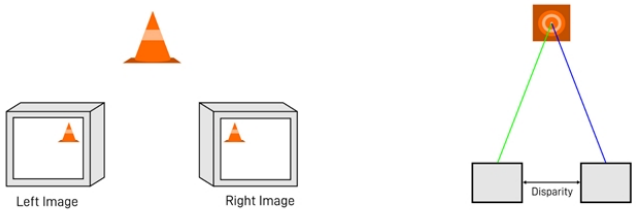


Figure 1: 3D Rays Intersection

Stereo vision in machine vision is a passive technology, as it does not require any artificial illumination to work. A stereo camera can simply be plugged in, calibrated, and deployed. Some stereo vision applications will, however, benefit from artificial illumination or a structured light source to aid visibility; This is known as active stereo and has its pros and cons just as passive stereo does. Stereo vision can be CPU intensive when not hardware accelerated with FPGAs or GPUs for example. Passive stereo camera systems can be deployed without the need for lasers or LEDs and can generally perform effectively in most ambient lighting conditions but if the system is operating in low light, or scanning non-textured scenes or objects with textureless surfaces, then stereo vision tends to underperform as a 3D technology. With no lasers or expensive lighting required, passive stereo vision can be much more affordable compared with 3D machine vision technologies. Stereo vision can cope well with long distances and moving objects, something that other 3D imaging technologies tend to fall short on. Once calibrated, a stereo vision camera system can go on to detect depth in real time. Considering all the difficulties mentioned above related to the use of the high resolution depth sensor, new ways has been taken to deal with depthmap acquisition that exploit the convolution neural networks. In 2014, Eigen et al. [1] introduce CNNs to the SIDE problem and achieve relatively good performance, when compared to earlier methods. CNN based solutions were already achieving quite satisfactory results in different vision problems at that time. Eigen et al. utilize the experience gained so far on CNNs and combine it with problem-specific knowledge to tackle the SIDE problem. They formulate the problem as a supervised regression problem and solve it with their framework. An evolution to this and what we base our model on, is Multiple Image Super-Resolution (MISR), where we feed a CNN for Super-Resolution of depth images with its corresponding RGB high resolution image; The detailed structures in the RGB image can help the net dealing with the high frequency components and recognizing easily different objects in the scene;

3. Methods

3.1. Architecture

Both 4x and 8x networks present the following components:

- convolution block: we have a 2D convolution operation and the Leaky Rectified

Linear Unit (Leaky ReLU) which is an improved version of ReLU function as it has a it has a small slope for negative values instead of a flat slope in the negative area. The slope coefficient is determined before training and not learnt during training. This functions introduce non linear real-world properties to artificial neural networks.

- subpixel convolution block: it takes as input the scaling factor we want to perform. The core of the subpixel convolution is the depth to space function which consists of a rearrangement of the pixel.

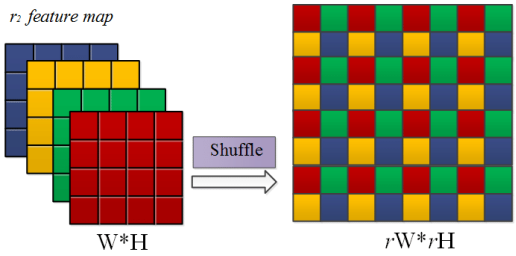


Figure 2: example of a subpixel convolution function: depth to space shuffle

Each pixel from each one of the r^2 feature map of dimension W*H is mapped into one r^2 square area in the output image. This means that the final image will have a total dimension of r^2 x WxH so that the total number of pixels is consistent with the HR image to be obtained.

- residual block: we have two branches: the main branch and the skip branch. The main branch is fed into a convolution layer and subsequently into an activation layer. The skip branch is basically the same tensor used as input for the convolution layer. Both are then summed together into the output tensor.
- upsample block: the first operation is a convolution layer followed by a Leaky ReLU activation, then we perform two subsequently residual block operation; based on the scaling factor we perform the actual upsample activity and lastly we repeat the convolution block.
- downsample block: this is only applied to color branch. we perform two subsequent convolution layers with their related Leaky ReLU activation layers, then we have the 2D max pooling layer which downsamples the input along its spatial dimensions (height and width) by taking the maximum value over an input window for each channel of the input. The window is shifted by strides along each dimension.
- composed block: is made up of one residual block, one upsample block with a two scaling factor, which performs the depth to scale function we discussed before, and one last residual block.

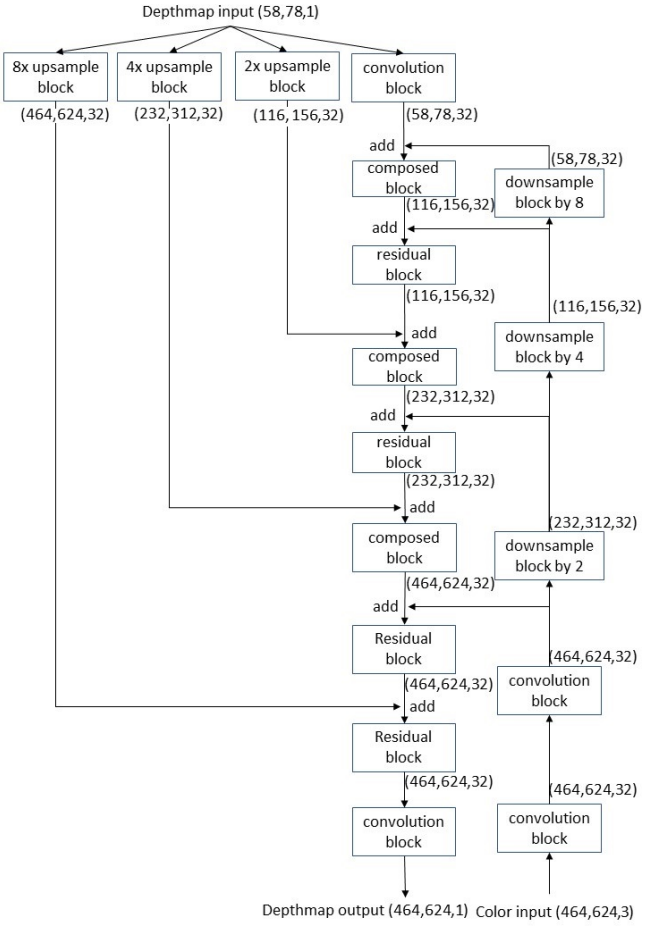


Figure 3: architecture of the 8x upsample network trained with NYU dataset

As you can see from Figure 3 we have two branches, one primary, the depthmap branch, dealing with depthmap manipulation and the other one, the secondary, which deals with RGB images manipulation. Seven times information belonging to the color branch is fused with depth information together into the primary depthmap branch.

3.2. Dataset

For training process we used three different datasets. MVS-Synth Dataset [3] which is a synthetic photo-realistic dataset that consists of scenes with frames of urban outdoor settings captured in a video game. Compared to other synthetic datasets, MVS-Synth Dataset is more realistic in terms of context and shading, and compared to real-world datasets, it provides complete ground truth disparities which cover regions such as the sky, reflective surfaces, and thin structures, whose ground truths are usually missing in real-world datasets. Since we use a sigmoid as last activation layer we performed a normalization preprocess operation before feeding the network. Since we are dealing with outdoor scenes, we decided to compute a clamping operation: all the object that has an indefinite long distance, i.e. the sky in the background, has a value we fixed at 0 which is translated into a black color. This also helps creating a strong contrast to make the silhouette of long distance object emerge with respect to the background.
NYU dataset [10]. It consists of 1449 RGBD images, gathered from a wide range of commercial and residential buildings in three different US cities, comprising 464 different indoor scenes across scene classes such as bathrooms, kitchens, libraries, living rooms. The original high resolution RGB images and the high resolution depthmaps show a white frame. This could be source of a problem for the network in the learning process and when computing the indexes for quantitative evaluation. When we do upsample operation, some of the new pixel values could wrongly be influenced by them and even when computing measure indexes, their values are badly effected by these white pixel which does not reflect the real scene. To deal with this we decided to crop all those images from 640x480 pixel to 624x464 pixel and then we could perform the downsample operation, using bicubic algorithm, to obtain the 58x78 pixel low resolution depthmaps fed to the network as input. The low resolution depthmap and high resolution depthmap are saved as .npy files. For testing phase we also used Middlebury dataset; we took a subset of 30 RGBD pairs from the 2001-2006 datasets provided by Lu et al. [8]. They are an enhancement of the original Middlebury dataset [9], since the original depthmaps presented noise and multiple black colored area which affected the depthmap values.

3.3. Training

For compiling process we used the Adam optimizer and a custom loss function, which has the combined action of two loss functions: L1 and DSSIM. The L1 loss, also known as Absolute Error Loss, is the absolute difference between a prediction and the actual value, calculated for each example in a dataset. The aggregation of all these loss values is called the cost function, where the cost function for L1 is commonly Mean Absolute Error (MAE). The Data Structural Similarity Index Measure (DSSIM) can be applied directly to the floating point data; we did not apply this loss on the whole picture but first we extract some patches from it and then we compute the ssim. This is done at batch level and once we compute the ssim on all the patches we compute the mean between them and obtain a final loss value for that batch. Using two loss functions allows us considering distinctive aspects related to the discrepancy between the groundtruth and the predicted value from the network.
For training we implemented fit generator function, so we can keep high the number of learnable parameters without the drawback of memory video saturation. Furthermore we perform some data augmentation. In our case we have two random bit generators to perform either a random vertical and horizontal flip.

3.4. Parameters

After several trials, a filter size of 32 is selected, creating a wide network together with the branches, although it increases the training time after 70 epochs it achieves better results than the 16 filter model. From the training results, we can observe that the error surface has lots of error peaks in high parameter numbers. This can be the result of the network's residual nature. The network residual part focuses on the high frequency details; this might entail the network to produce tons of high frequency details that do not exist in the image resulting in high error. This means that the network needs to be trained using a lower learning rate as the network gets deeper and wider. We found 2.5×10^{-4} to be a good value as starting learning rate value.

Kernel size is another parameter deserved to be noticed. One of the reason to prefer small kernel sizes over fully connected network is that it reduces computational costs and weight sharing that ultimately leads to lesser weights for back-propagation. 3x3 sized kernel has became a popular choice. 2x2 and 4x4 are generally not preferred because odd-sized filters symmetrically divide the previous layer pixels around the output pixel and if this symmetry is not present, there will be distortions across the layers.
Epoch value is important to have a good balance between the time of training and the quality of the output. Depending also on the learning rate, training for too long could lead to missing the loss, bouncing back and forth the minimum we are looking for but never reaching it, or even exploding the value, meaning that we will keep getting away from the minimum, increasing the loss even more and never finding the minimum. Based on all the test we run, we came up with a sweet spot of 200 epochs for a regular training and additional 100 epochs when it comes to finetuning.
It has been observed that smaller batch sizes not only has faster training dynamics but also better generalization on test dataset versus larger batch sizes. Using small batch size means the model makes updates that are all about the same size. Because neural network systems are prone overfitting, the idea is that seeing many small batch size, each batch being a noisy representation of the entire dataset, will prevent the neural network from overfitting on the training set and performing badly on the test set. Therefore, smaller batch sizes means the model can find the faraway better optima whereas large batch size means the model cannot. With the other parameter fixed, we decided to maintain the batch size equal to 1 in excahnge of a longer time of training, since it gives us the best performance.

4. Results

Since we obtained similar result as far as concern the 8x and the 4x upsample network , for simplicity we show results related to the only 8x network. Table 1 shows quantitative results of our model trained with the syntethic dataset. As you can see the model outperforms the standars upsample algorithm. We had an improvement of the 65% , 12% and 40% respectively on the RMSE, SSIM and PSNR metrics if compared to the bicubic method.

MVS dataset	RMSE	SSIM	PSNR
Nearest	0.0960	0.8452	20.5823
Bilinear	0.0870	0.8540	21.4665
Bicubic	0.0826	0.8669	21.9263
Proposed Method	**0.0297**	**0.9733**	**31.0944**

Table 1: Table comparing our 8x network with interpolation algorithm

We highlight the progress our model can perform if a finetuning operation is carried out. Starting from 200 epochs of training and Mean Square Error(MSE) as loss function. Then we perform a first finetune introducing the combined action of MAE and DSSIM; we trained the network 100 times more. We gained a 5.6% on the RMSE, meaning that similiraty loss takes into account aspect that the MSE does not. We take a step further and add 100 supplementary epochs of training reducing the learning rate. The little improvement shows hat the network has reached its limit and it does not make sense to investigate further with this configuration.

MVS dataset	RMSE	SSIM	PSNR
Single Loss	0.0316	0.9693	30.4971
Double Loss	0.0298	0.9719	31.0221
Halved Learning Rate	**0.0297**	**0.9733**	**31.0944**

Table 2: Table comparing finetuning steps for 8x architecture

Now we take into exam the model trained with the realistic indoor dataset NYU2. In the table 3 below, you can see the metrics related to the CNN using 8 as scaling factor. Also in this case our netowrk outperforms the standards upsample algorithms. We had an improvement of the 60% , 10% and 33% respectively on the RMSE, SSIM and PSNR metrics of our network compared to the bicubic method.

NYU dataset	RMSE	SSIM	PSNR
Nearest	0.0932	0.7422	20.7804
Bilinear	0.0740	0.8746	22.7875
Bicubic	0.0676	0.8775	23.5404
Proposed Method	**0.0284**	**0.9637**	**31.4272**

Table 3: Table comparing our 8x network with interpolation algorithm

We also compared our method to other previous

state of the art works which used [10] dataset as well in their testing phase, including recent learning based methods such as DJFR [6] and DKN [5]. Table 4 shows the detailed results on the average RMSE which is measured in centimeters.

RMSE on NYU dataset	x4 upsample	x8 upsample
DSMG[4]	3.02	5.38
DG[2]	3.68	5.78
DJF[6]	2.80	5.33
DJFR[7]	2.38	4.94
PAC[11]	1.89	3.33
DKN[5]	1.62	3.26
Proposed Method	**1.52**	**2.84**

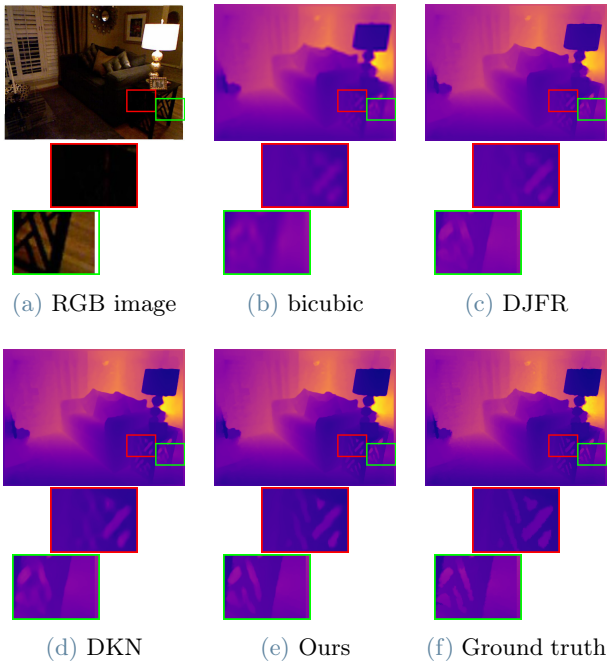
Table 4: Quantitative comparison with the state of the art on depthmap upsampling in terms of average RMSE

Here in table 5 we show the results obtained in the testing phase, using the Middlebury dataset [8]. In the scenario the average RMSE is measured in the original scale of the provided disparity. We compared both our 4x and 8x architecture to other previous state of the art works as done before. Table 5 shows that our method outperforms the existing one.

RMSE on Middlebury dataset	4x upsample	8x upsample
Bicubic	2.32	3.99
DSMG[4]	1.88	3.45
DG[2]	1.97	4.16
DJF[6]	1.68	3.24
DJFR[7]	1.32	3.19
PAC[11]	1.32	2.62
DKN[5]	1.23	2.12
Proposed Method	**1.19**	**1.97**

Table 5: Table comparing the RMSE index of our network trained with Middlebury dataset to some of the state of the art methods,both for the x8 and 4x upsample model

In Figure 4 we show a qualitative comparison between our model trained with NYU dataset and other previous works. We used a plasma colormap instead of the usual grayscale to better emphasize the contrasts in the depthmaps, since the low range of measurement in the indoor scenes. We call the attention to the details within the red and green rectangles. These are related to the sides of a night table near the sofa, which present empty gaps and so a sharp change in the depth information. Going from the bicubic interpolation to our method, a huge improvement takes place; the blurry effect reduces



(a) RGB image (b) bicubic (c) DJFR

(d) DKN (e) Ours (f) Ground truth

Figure 4: Qualitative comparisons of $\times 8$ guided depth map super-resolution on the NYU v2 dataset

and the edges become sharper. This happens because in the other works a mean between the depth values belonging to the night table and the ones belonging to the floor behind is carried out. The color information allows our network to assign each pixel a more precise value.

5. Conclusions

In this work we propose two different deep learning based methods for depth map upsampling. Both of these are proposed in order to exploit a high resolution color image of the same scene simultaneously acquired. In particular, one is suggested for training and testing with photorealistic synthetic dataset in order to minimize possible acquisition errors and working with highly accurate groundtruth data. On the other hand, we have a network that deals with a real dataset in order to see if our model behaves work well also working with low quality groundtruth and color images, taking into account possible acquisition error from the camera. We tested both model on 8x and 4x upsample scaling factor. Both qualitative and quantitative results, highlight the massive impact of considering high resolution color images while performing depth map upsampling. In particular by using high

resolution RGB images as additional input data, deep learning based 8x upsampling can outperform 4x upsampling (also based on deep learning but not using the auxiliary RGB image). This is due to the high frequency details present in the color images that could be efficiently transferred on the upsampled depth map by the proposed network. Although further investigation must be conducted on higher quality not synthetic data, these results show that a system composed by a cheap low resolution depth estimation device coupled with a high resolution color camera could outperform more expensive higher resolution 3D ranging devices.

6. Acknowledgements

I wish to thank Professor Marco Marcon and Marco Paracchini for their guidance, collaboration and help during the realization of this work.

References

[1] David Eigen, Christian Puhrsch, and Rob Fergus. Depth map prediction from a single image using a multi-scale deep network. In Zoubin Ghahramani, Max Welling, Corinna Cortes, Neil D. Lawrence, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 2366–2374, 2014.

[2] Shuhang Gu, Wangmeng Zuo, Shi Guo, Yunjin Chen, Chongyu Chen, and Lei Zhang. Learning dynamic guidance for depth image enhancement. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 712–721. IEEE Computer Society, 2017.

[3] Po-Han Huang, Kevin Matzen, Johannes Kopf, Narendra Ahuja, and Jia-Bin Huang. Deepmvs: Learning multi-view stereopsis. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[4] Tak-Wai Hui, Chen Change Loy, and Xiaoou Tang. Depth map super-resolution by deep multi-scale guidance. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part III*, volume 9907 of *Lecture Notes in Computer Science*, pages 353–369. Springer, 2016.

[5] Beomjun Kim, Jean Ponce, and Bumsub Ham. Deformable kernel networks for joint image filtering. *Int. J. Comput. Vis.*, 129(2):579–600, 2021.

[6] Yijun Li, Jia-Bin Huang, Narendra Ahuja, and Ming-Hsuan Yang. Deep joint image filtering. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part IV*, volume 9908 of *Lecture Notes in Computer Science*, pages 154–169. Springer, 2016.

[7] Yijun Li, Jia-Bin Huang, Narendra Ahuja, and Ming-Hsuan Yang. Joint image filtering with deep convolutional networks. *IEEE Trans. Pattern Anal. Mach. Intell.*, 41(8):1909–1923, 2019.

[8] Si Lu, Xiaofeng Ren, and Feng Liu. Depth enhancement via low-rank matrix completion. In *2014 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2014, Columbus, OH, USA, June 23-28, 2014*, pages 3390–3397. IEEE Computer Society, 2014.

[9] Daniel Scharstein and Chris Pal. Learning conditional random fields for stereo. In *2007 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2007), 18-23 June 2007, Minneapolis, Minnesota, USA*. IEEE Computer Society, 2007.

[10] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from RGBD images. In Andrew W. Fitzgibbon, Svetlana Lazebnik, Pietro Perona, Yoichi Sato, and Cordelia Schmid, editors, *Computer Vision - ECCV 2012 - 12th European Conference on Computer Vision, Florence, Italy, October 7-13, 2012, Proceedings, Part*

V, volume 7576 of *Lecture Notes in Computer Science*, pages 746–760. Springer, 2012.

[11] Hang Su, Varun Jampani, Deqing Sun, Orazio Gallo, Erik G. Learned-Miller, and Jan Kautz. Pixel-adaptive convolutional neural networks. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019*, pages 11166–11175. Computer Vision Foundation / IEEE, 2019.