https://doi.org/10.2352/EI.2022.34.16.AVM-126

Efficient high-dynamic-range depth map processing with reduced precision neural net accelerator

Peter van Beek, Chyuan-Tyng (Roger) Wu, and Avi Kalderon
Imaging and Camera Technology Group, Intel Corp., Santa Clara, CA, USA

Abstract

Depth sensing technology has become important in a number of consumer, robotics, and automated driving applications. However, the depth maps generated by such technologies today still suffer from limited resolution, sparse measurements, and noise, and require significant post-processing. Deep convolutional neural nets can be used to perform denoising, interpolation and completion of depth maps. Depth map data often has higher dynamic range than common 8-bit image data and may be represented as 16-bit values; however, in practical applications there is a need to enable efficient low-power deep net inference with 8-bit precision. In this paper, we explore methods to process high-dynamic-range depth data using neural net inference engines with 8-bit precision. We propose a simple technique that attempts to retain signal-to-noise ratio in the post-processed data and can be applied in combination with most convolutional network models. Our initial results using depth data from a consumer camera device show promise, achieving inference results with 8-bit precision that have similar quality to floating-point processing.

Introduction

Depth sensing technologies, capturing the 3D environment, enable a variety of applications in automated/autonomous vehicles, robotics, AR/VR, user interfaces, etc. Depth sensing technologies include stereo cameras, time-of-flight sensors, structured-light systems, as well as lidar. RGB-D cameras combine depth sensing with color (RGB) imaging.

In many cases, depth data captured by 3D sensing technology is noisy, low-resolution and/or sparse, and benefits significantly from post-processing. Figure 1 shows a few example depth images and their corresponding graylevel images captured with a commercially available RGB-D camera. Various image processing methods can be used to denoise, deblur, and interpolate depth maps, in a similar manner to processing regular images. Depth map denoising and depth map completion methods based on deep convolutional neural nets (CNNs) have been proposed recently, for example [1][2][3]. Methods using deep learning and CNNs are capable of a deeper understanding of the entire scene and outperform traditional methods, as in other domains. Datasets for training and evaluating learning-based depth map processing methods include the KITTI depth completion dataset [3] and the NYU-Depth-V2 dataset [4].

The depth completion problem is defined more concretely in Figure 2 (top). In general, the output of the camera includes the depth map itself, a confidence map indicating a measure of confidence in each depth sample, as well as a regular image. The confidence map may represent a continuous value between 0.0 and 1.0 or may represent a binary occupancy map. We assume the availability of a regular color or gray-level image that is spatially registered with the depth map. Figure 2 (bottom) illustrates a specific example of data captured using a stereo camera, where pixels/samples with unavailable or 'zero-confidence' depth values are visualized as black in the color-coded image, and the registered left stereo image is available to the post-processing algorithm. In this example, the only output of the post-processing step is an improved depth map (shown with the same color coding).

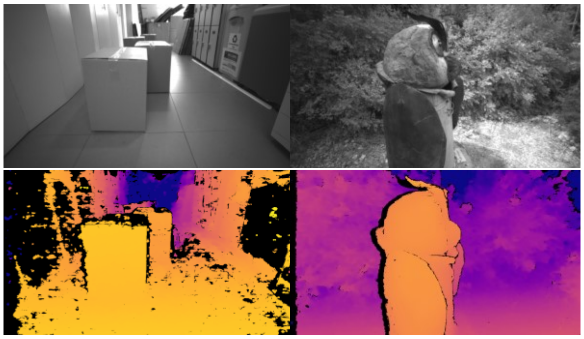

Figure 1: Examples of depth maps and corresponding gray-level images. Depth maps are visualized using a color coding where blue corresponds to large depth/distance samples and yellow corresponds to small depth/distance samples. Black corresponds to samples where a depth/distance value could not be obtained.

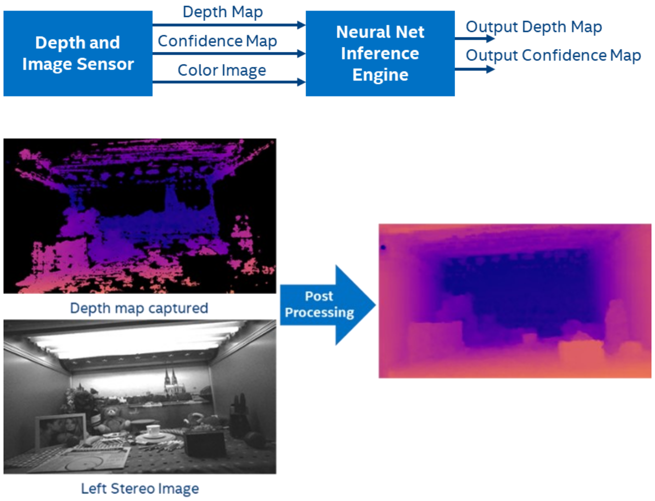

Figure 2: Depth map completion problem setup (top). Example depth map completion inputs and output (bottom).

Often, the data captured by depth sensing technologies needs to be represented with 10-, 12-, or 16-bits per sample, due to its

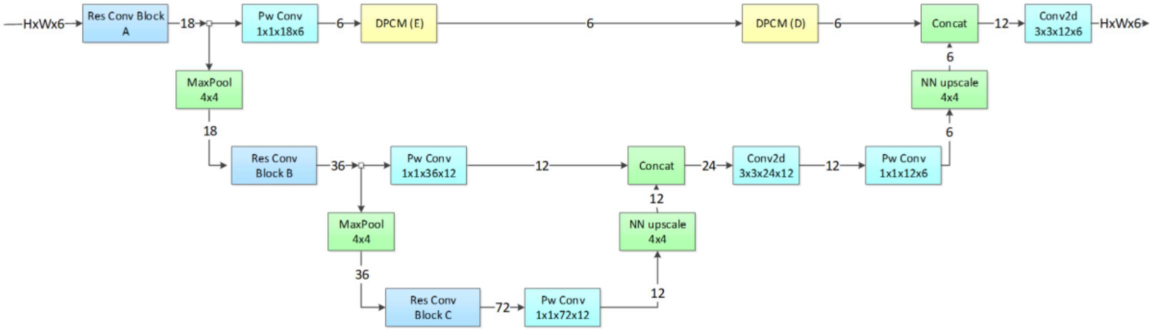

Figure 3: Encoder-decoder CNN used for depth map post-processing, reproduced from [6].

high dynamic range and precision requirements. For example, short-range depth sensor data may be represented in mm units and have a range in the order of several tens of meters, while long-range depth sensor data may be represented in cm units and have a range in the order of several hundred meters.

However, to reduce the computational cost and increase processing speed, software and hardware neural net inference engines and accelerators often process the data with 8-bit precision in practical applications. Especially for inference engines for so-called edge applications, reducing precision is critical to reduce power, bandwidth and memory requirements. Hence, processing depth data with higher bit-precision presents a problem of interest.

In this paper, we explore methods to process high-dynamic-range 16-bit depth data using neural net inference engines with 8-bit precision. The goal of the inference engine is to run post-processing tasks such as depth completion and denoising, to improve the quality of the depth map, while retaining its precision.

Efficient depth map post-processing

Although not the focus of this paper, we first describe the convolutional neural network used in our work for depth map completion. We use a very lightweight fully convolutional encoder-decoder network sketched in Figure 3. This network, introduced in [6], was designed for highly efficient image processing in dedicated hardware. The network is similar in architecture to U-Net [5] and contains an encoder that extracts features at the original resolution, 4x subsampled resolution, and 16x subsampled resolution, a decoder that successively rebuilds a depth map at the original resolution, and skip-connections at each level of resolution that connect encoder and decoder. The network supports up to 6 channels input and 6 channels output, each with 8-bit precision. The number of channels in the network bottleneck (at 16x scale) is 72. Convolutional blocks use highly efficient separable convolution layers, and use an IIR filter instead of the traditional FIR convolution filter in the bottleneck. The total number of parameters is only about 50,000 (compared to ~8 million for the standard U-Net).

We used a TensorFlow implementation of this lightweight CNN model to obtain simulation results. The network was trained and evaluated using a relatively small depth data set collected by our team, for the purpose of experiments. In this data set, we used a RealSense depth camera with and without IR illumination. The depth map captured without IR illumination serves as the input depth map in our experiments, while the depth map captured with IR illumination serves as the ground-truth. Example inputs, ground-truth, and outputs, using floating point precision calculations in TensorFlow, are shown in Figure 4.

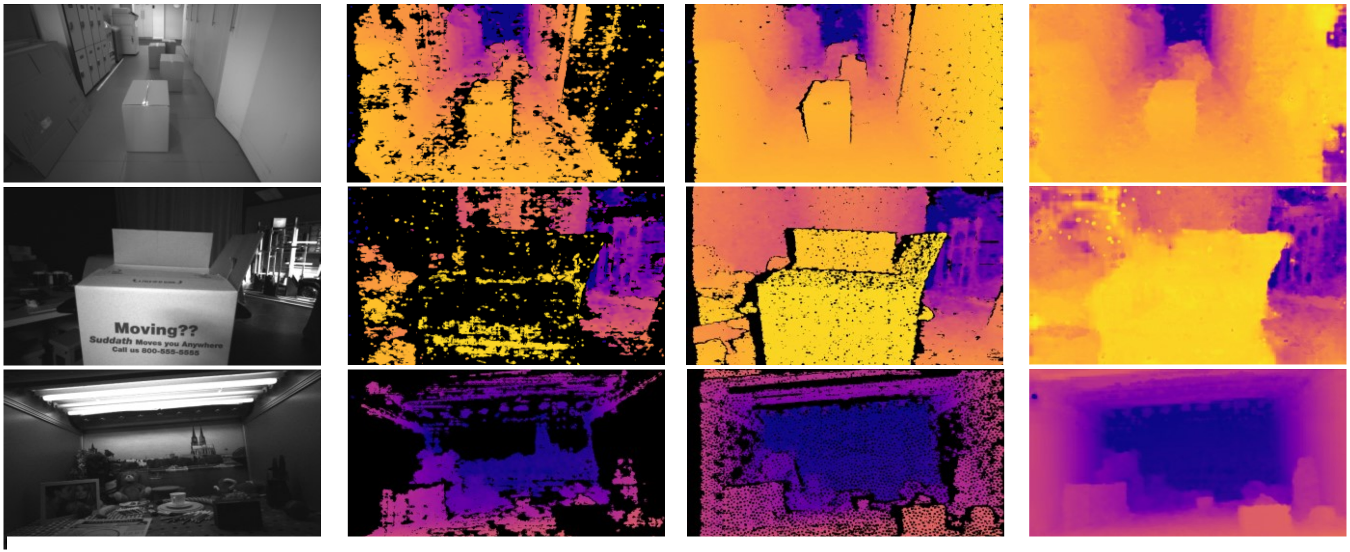

Figure 4: Depth map completion examples. Columns from left to right: input gray-level image; input depth map (captured without IR); ground-truth (captured with IR); processed output depth map. Note that our ground-truth data may still have “invalid” samples.

Depth map completion at limited precision

In this section, we describe methods to process high-dynamic range depth data on an 8-bit neural net inference engine. As mentioned, depth data is often represented as 10-, 12-, or 16-bit data. The distribution of the depth values over the data set collected by our team is shown in Figure 5. The majority of depth values are below ~5 m. However, the dynamic range varies strongly from image to image, depending on the scene. In our experiments, we will assume each depth map is represented as a 16-bit precision image, even if the dynamic range of the depth samples might require only 10- or 12-bit precision.

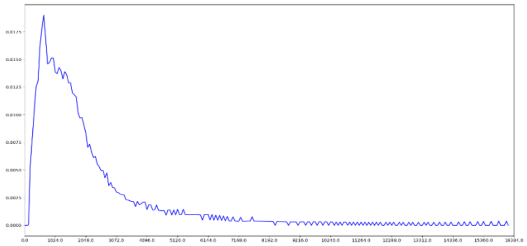

Figure 5: Distribution of depth values over our data set. The range of the horizontal axis in the plot is [0, 16384] and the unit of depth values is mm (hence up to 16.4 m).

We have explored several methods for post-processing of 16-bit data using an inference engine that is constrained to 8-bit inputs and outputs. In all our experiments, we include a pre-normalization of the input data in each depth map due to the strongly varying range in different scenes. Namely, the depth values are normalized based on the 98th percentile of the histogram of depth values in an individual image.

The **baseline** method is to simply uniformly quantize the 16-bit depth maps (after normalization) to 8-bit values, or in other words, use the MSB of each 2-byte value.

The first method that attempts to retain more than 8-bit precision is to simply break each 16-bit input depth value into its MSB and LSB parts and feed each to a separate channel of the network. We refer to this method as **direct channel split**. This approach might be counter-intuitive since the LSB depth map shows 'wrap-around' effects at level-set boundaries between magnitude ranges of 256 (8 bits). However, the network appears to be able to represent and preserve such images, while interpolating/extrapolating to areas in the map without valid data.

The second method is to utilize a **tone mapping** operator on the input depth image and subsequently quantize the depth values prior to feeding the values to the inference network. Subsequently, an inverse tone mapping operator is applied to the network output depth values. This is equivalent to non-uniform quantization. In addition, we propose to utilize multiple channels, where multiple tone mapping operators are tuned to different ranges of the depth values, and the network output channels are recombined to generate the final post-processed result. We refer to this method as **multi-channel tone mapping**, and the approach is similar to well-known techniques applied to high dynamic range gray-level/color image data for capture, representation, and communication purposes (e.g. multi-exposure HDR image capture) [7].

The multi-channel tone mapping approach is illustrated in Figure 6. The CNN has up to 6 input channels available, due to its implementation as hardware accelerator. In this example, 4 input channels are used to represent multiple ranges of the input dynamic range, each with 8 bits. Correspondingly, 4 output channels represent processed depth maps in multiple ranges. A channel combination step is applied after inverse tone-mapping. The combination operator, in our case, consists of simple addition.

We use simple gamma functions for tone mapping the depth values, as shown in Figure 7. The goal is to be able to represent large depth values with sufficient accuracy while sacrificing precision, but at the same time represent small depth values more precisely. The gamma function is a convenient and simple way to achieve this. By using multiple gamma functions, some channels can be used to represent the short-range depth values more precisely, while other channels can be used to represent mid or high-range depth values more precisely.

Finally, we make the gamma values for each channel **trainable** in conjunction with training the post-processing CNN itself. The gamma tone mapping is implemented as a 1-parameter trainable layer in TensorFlow. These layers are attached to the CNN model during end-to-end training, so that the optimal parameters are determined automatically by minimizing the loss function. In our experiments, we find that such end-to-end training of the gamma function parameters indeed converges to parameter values that would be expected, i.e. compressive tone curves prior to quantization and CNN processing, as illustrated in Figure 7.

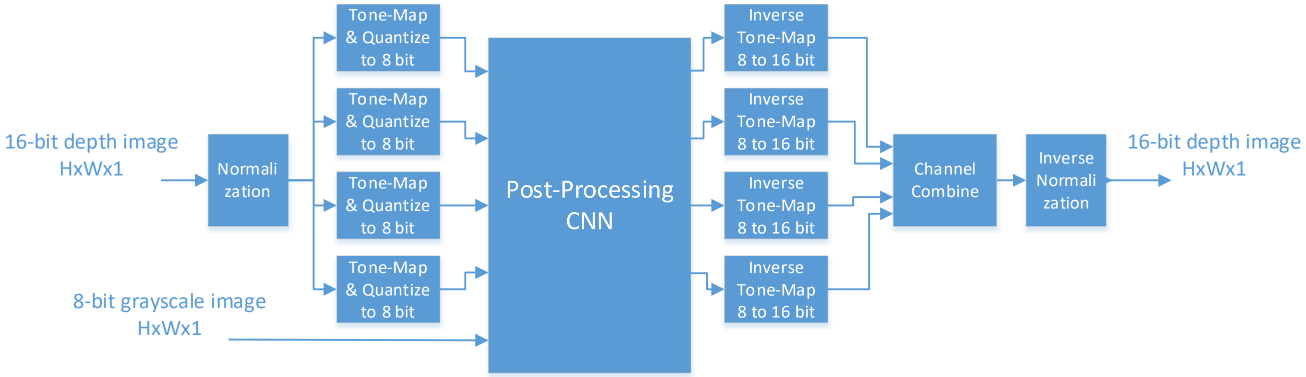
16-bit depth image
HxWx1
Normalization
Tone-Map & Quantize to 8 bit
Tone-Map & Quantize to 8 bit
Tone-Map & Quantize to 8 bit
Tone-Map & Quantize to 8 bit
8-bit grayscale image
HxWx1
Post-Processing CNN
Inverse Tone-Map 8 to 16 bit
Inverse Tone-Map 8 to 16 bit
Inverse Tone-Map 8 to 16 bit
Inverse Tone-Map 8 to 16 bit
Channel Combine
Inverse Normalization
16-bit depth image
HxWx1

Figure 6: Multi-channel tone mapping for post-processing high dynamic range depth maps.

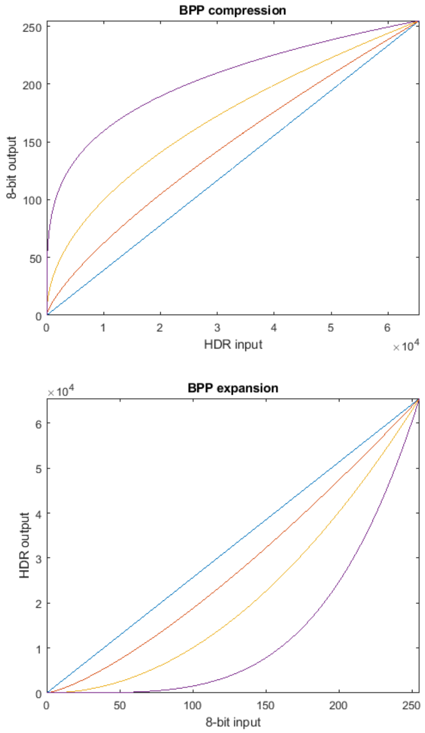

Figure 7: Example gamma functions for depth tone mapping; forward mappings (compression) at top, and reverse mappings (expansion) at bottom.

Experiments

Experimental Setup

We performed experiments using data obtained from a consumer depth sensing camera based on stereo imaging, namely RealSense cameras, which provides 16-bit depth data as well as a registered gray-level image. Small datasets were collected for training and evaluation of the post-processing network. As mentioned, depth maps captured with IR illumination serve as ground-truth, while depth maps captured without IR illumination serve as the input depth map in our experiments. Depth maps captured without IR illumination contain more "missing" samples, i.e. samples where stereo disparity could not be calculated with sufficient confidence. The loss function for training the depth completion CNN is defined as the relative MSE between output and ground-truth, computed over samples that are valid in the ground-truth depth map. We used a small data set of 182 images captured by our team, split into 162 training and 20 test images for experiments.

All results were obtained with our TensorFlow software implementation. The effects of the different quantization methods are compared to direct floating-point processing. The bit precision limitations of the hardware accelerator were simulated by applying quantization to the data in the appropriate places in the TensorFlow model code.

Experimental Results

Quantitative results over the evaluation test set are shown in Table 1. The Mean Absolute Error (MAE) and Peak Signal-to-Noise Ratio (PSNR) between the depth map and its ground-truth were used as objective metrics. Both the MAE and PSNR are computed only over samples that are valid in the ground-truth image, averaged over the images in the test set. The table shows that a significant reduction in MAE (and improvement in PSNR) is achieved using post-processing (all rows except the first), relative to the unprocessed input (first row). The second row shows the MAE and PSNR obtained with the network using floating point processing, as a reference for experiments with quantization. The third row shows that direct quantization to 8-bit precision (by scaling values and taking the MSB) leads to a significant loss of quality relative to floating point processing. The fourth row shows the result of the direct channel split method, which is close to the result with floating point processing. The fifth row shows the result obtained with the proposed tone mapping in a single channel. The bottom row shows the result obtained with the proposed multi-channel technique, using four channels with each using a tone mapping operator prior to 8-bit quantization. The proposed methods achieve the best MAE and PSNR results, even exceeding floating-point processing. This suggests there is some benefit to using multi-channel processing and tone mapping of the input depth map in itself, even if 8-bit quantization wasn't required.

Method	**MAE**	**PSNR (dB)**
Input depth map (no post-processing)	695.24	33.07
Single channel floating point (no quantization)	438.62	37.41
Single channel with 8-bit quantization	456.84	36.96
Direct channel split into MSB and LSB	438.99	37.49
Single channel with tone map operator + 8-bit quantization	431.00	37.63
Four channels with tone map operators + 8-bit quantization	425.53	37.60

Table 1: MAE and PSNR over depth maps in the test set (relative to corresponding ground-truth depth maps).

Figure 8 shows visual results using the proposed quantization methods. In each case, the grayscale image is shown at top left. The output of the network using floating point inference is shown at the top right. The bottom images show the output of the network when using the proposed tone mapping technique and using multiple 8-bit quantized input and output channels, where the bottom left image shows the result using single channel and the bottom right image shows the result using four channels. Some contouring artifacts due to quantization can be observed in the bottom left image, whereas these are not visible in the bottom right.

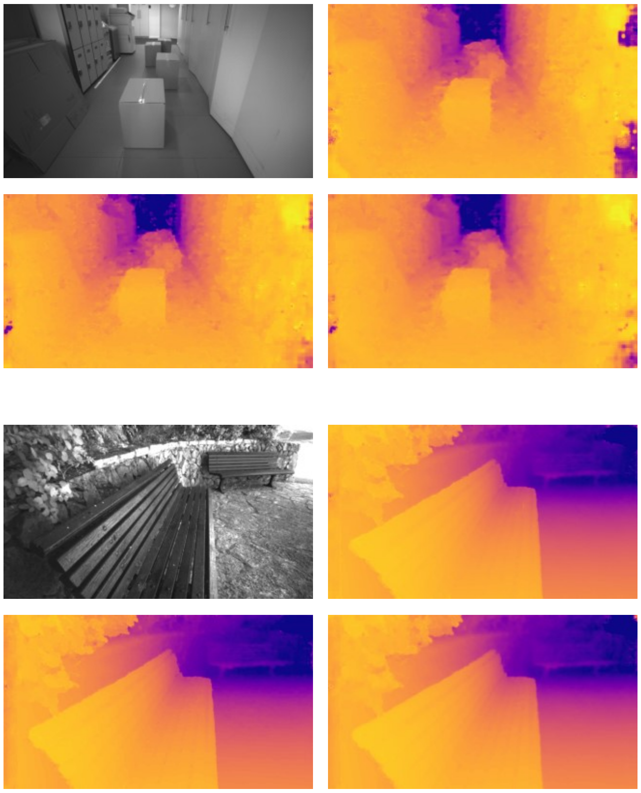

Figure 8: Example depth map processing results. In each example the gray-level input is at top left; floating-point processing output at top right; single-channel processing with trained tone mapping and 8-bit quantization at bottom left; four-channel processing with trained tone mappings and 8-bit quantization at bottom right.

Conclusions

In this paper, we discuss methods for efficient depth map post-processing using using a neural net accelerator with reduced bit precision. The proposed methods build upon depth map processing methods using deep convolutional neural nets. We describe simple techniques that support processing of depth maps with high bit precision on CNN inference engines constrained to 8-bit input and output data. We proposed the use of trainable tone mapping functions and multiple input/output channels as input to the CNN, and recombining the results at the output of the CNN. Our initial results indicate that the quality of the processed depth maps using these techniques can equal or even improve upon the quality of output using floating point processing, even when the data is quantized and processed at 8-bit precision in each channel of the CNN.

In our experiments, we used a very lightweight encoder-decoder CNN model. However, the proposed quantization techniques are largely agnostic to the CNN model architecture itself and can be applied to a variety of models, providing that the number of model input and output channels is sufficient. These processing techniques are simple and can likely be implemented by a variety of inference engines. Hence, the proposed techniques achieve low-power and efficient inference on depth maps in practical applications.

References

[1] S. Yan et al., "DDRNet: Depth Map Denoising and Refinement for Consumer Depth Cameras Using Cascaded CNNs," *European Conference on Computer Vision (ECCV)*, 2018.

[2] J. Park et al., "Non-Local Spatial Propagation Network for Depth Completion," *European Conference on Computer Vision (ECCV)*, 2020, arXiv:2007.10042.

[3] J. Uhrig et al., "Sparsity Invariant CNNs," *International Conference on 3D Vision (3DV)*, 2017, arXiv:1708.06500.

[4] N. Silberman et al, "Indoor Segmentation and Support Inference from RGBD Images," *European Conference on Computer Vision (ECCV)*, 2012.

[5] O. Ronneberger et al., "U-Net: Convolutional Networks for Biomedical Image Segmentation," *MICCAI 2015*, Springer LNCS Vol. 9351, arXiv:1505.04597.

[6] M. Asama et al., "A Machine Learning Imaging Core using Separable FIR-IIR Filters," arXiv:2001.00630, 2020.

[7] E. Reinhard, G. Ward, S. Pattanaik, P. Debevec, "High dynamic range imaging: acquisition, display, and image-based lighting," Elsevier/Morgan Kaufmann, 2005.

Author Biographies

Peter van Beek is a machine learning engineer with the Intel Imaging and Camera Technologies Group. He is supporting development of deep learning inference engines as well as optimization of camera imaging pipelines. Previously, he was with the Intel Autonomous Driving Group and Mobileye. Before joining Intel, Peter was a technical lead at Sharp Laboratories of America. He is a senior member of the IEEE and received a PhD from Delft University of Technology.

Chyuan-Tyng Wu received a Ph.D. in Electrical and Computer Engineering from Purdue University. His research was focused on computer vision and a depth information capture system. After college, he is working as an Algorithm Engineer in Imaging and Camera Technologies Group at Intel. His work includes multiple fixed function developments for the imaging signal processor and emerging IP incubation for computational photography and the machine learning applications.

Avi Kalderon received his BA and MBA from College of Management of Israel. He has more than 20 years of experience making cameras outstanding. His main focus has been image quality tuning, developing technology, and managing image quality teams for image signal processors for a wide range of products (mobile phones, laptops, security and automotive markets), from R&D stage to commercial products. Avi works/worked as technical IQ leader, currently at Intel and previously at Samsung.