https://doi.org/10.2352/EI.2024.36.18.3DIA-105

Neural Depth Encoding for Compression-Resilient 3D Compression

Stephen Siemonsma and Tyler Bell*

Department of Electrical and Computer Engineering, University of Iowa, Iowa City, IA 52242, USA
* ***tyler-bell@uiowa.edu***

Abstract

Recent advancements in 3D data capture have enabled the real-time acquisition of high-resolution 3D range data, even in mobile devices. However, this type of high bit-depth data remains difficult to efficiently transmit over a standard broadband connection. The most successful techniques for tackling this data problem thus far have been image-based depth encoding schemes that leverage modern image and video codecs. To our knowledge, no published work has directly optimized the end-to-end losses of a depth encoding scheme passing through a lossy image compression codec. In contrast, our compression-resilient neural depth encoding method leverages deep learning to efficiently encode depth maps into 24-bit RGB representations that minimize end-to-end depth reconstruction errors when compressed with JPEG. Our approach employs a fully differentiable pipeline, including a differentiable approximation of JPEG, allowing it to be trained end-to-end on the FlyingThings3D dataset with randomized JPEG qualities. On a Microsoft Azure Kinect depth recording, the neural depth encoding method was able to significantly outperform an existing state-of-the-art depth encoding method in terms of both root-mean-square error (RMSE) and mean absolute error (MAE) in a wide range of image qualities, all with over 20% lower average file sizes. Our method offers an efficient solution for emerging 3D streaming and 3D telepresence applications, enabling high-quality 3D depth data storage and transmission.

Introduction

With the 2024 release of the Apple Vision Pro and the Persona digital avatar system, 3D telepresence has become more widely accessible than ever. Apple's Persona system is technically very impressive, but it does not stream any live-captured 3D data. Despite recent advances in 3D capture hardware and methods that allow for the real-time capture of 3D range data at high resolutions and frame rates, the vast majority of mainstream use cases only use this data locally. Without compression, this data stream could be on the order of gigabits per second, a data rate that cannot be reliably sustained on a typical wireless broadband connection.

In order to encourage the development and adoption of novel 3D streaming use cases such as 3D telepresence, more efficient methods of compressing this information are required. Optimally, this information would be stored in a way that allows compression by modern image or video codecs, which are already highly optimized to exploit spatiotemporal redundancies and are very often hardware-accelerated. Unfortunately, depth values generally have higher bit depths than color image channels, so they cannot be accurately compressed with the most common lossy image formats. Depth value bits can be directly split across multiple image channels, but this results in rapid spatial oscillations and discontinuities that do not compress well. Instead, an efficient, compression-resilient depth encoding scheme is required to smoothly spread the depth information across the red, green, and blue color channels.

This paper proposes, to our knowledge, the first learned depth-to-RGB encoding scheme that is directly optimized for a lossy image compression codec. The key to this encoding scheme is a fully differentiable pipeline that includes a pair of neural networks sandwiched around a differentiable approximation of JPEG. An illustration of this pipeline can be seen in Fig. 1. When trained end-to-end, the proposed system is able to minimize depth reconstruction losses and easily outperform a state-of-the-art, handcrafted algorithm. If successfully applied to video codecs in the future, this neural encoding scheme will allow for a new generation of 3D data streaming applications. Our neural depth encoding system could be especially impactful on low-performance and bandwidth-limited devices such as cell phones and drones.

Contributions

- We introduce a novel neural depth encoding scheme optimized for lossy compression codecs to efficiently encode depth maps into 24-bit RGB representations.
- Utilize a fully differentiable pipeline, including a differentiable approximation of JPEG, allowing end-to-end training and optimization.
- Demonstrate significant performance improvements over a state-of-the-art method in terms of root-mean-square error (RMSE) and mean absolute error (MAE) across a wide range of image qualities, while also achieving over 20% lower average file sizes.
- Offer a potential efficient solution for emerging 3D streaming and telepresence applications, enabling high-quality and efficient 3D depth data storage and transmission.

Related Work

Multiwavelength depth (MWD) [1] is one of the few state-of-the-art methods for encoding depth data across the red, green, and blue color channels of a typical RGB image. MWD works by storing two sinusoidal encodings and a normalized depth map into the RGB channels of a color image. The red channel is a sinusoidal encoding of the normalized depth map, while the green channel is

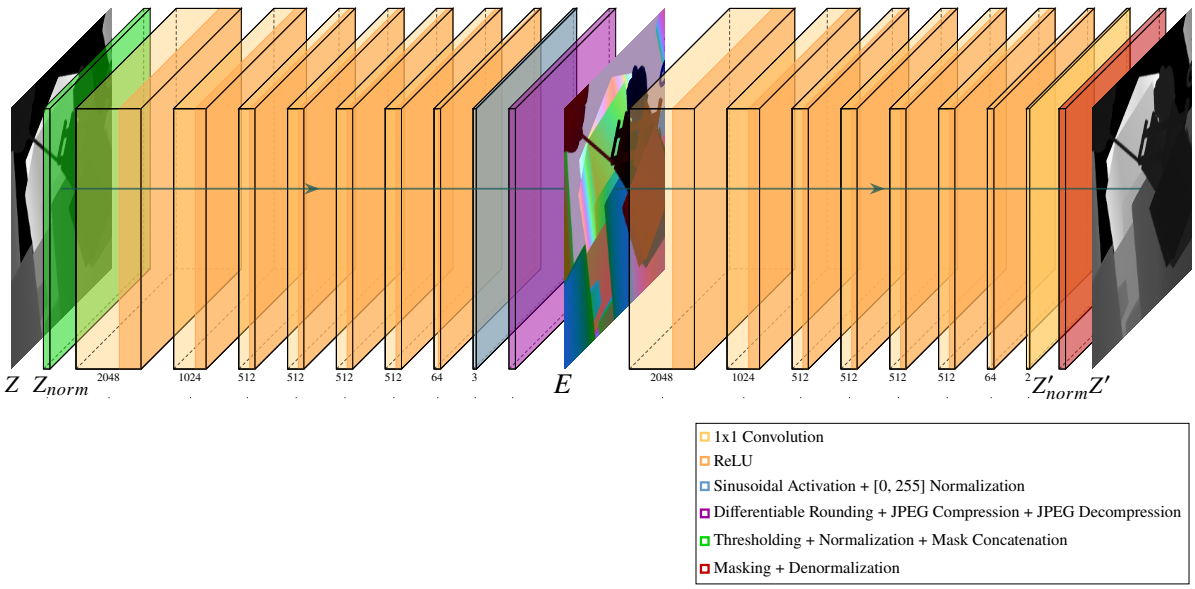

Figure 1. *Neural depth encoding, compression, and decoding pipeline. The depth map Z is first thresholded to the desired depth range and normalized between 0 and 1 to form Z_{norm}. A mask denoting the in-range and out-of-range regions is concatenated to Z_{norm} to form the input to the neural depth encoder. The output of the neural encoder is fed into a differentiable JPEG approximation, which produces a lossy RGB encoding E. This is then fed into the neural decoder that recovers Z'_{norm}, which is denormalized to the original depth range in order to produce the final depth map reconstruction Z'. Outside of training, this pipeline uses non-differentiable forms of rounding and true JPEG compression.*

a complementary cosinusoidal encoding. Finally, the blue channel simply stores a normalized depth map. The frequency of the sinusoidal encodings can be tuned to balance compressibility and accuracy (by adjusting the n_{stairs} parameter), while the normalized depth map guides the phase unwrapping process during decoding. Fig. 2 illustrates how MWD encodes a normalized depth range across the red, green, and blue color channels when n_{stairs} is set to 3, meaning that 3 periodic repetitions occur in the red and green channels. Although the encodings produced by MWD are smoothly varying and readily compressible, this handcrafted encoding scheme was never directly optimized for the image compression codecs that it sought to exploit. While deep learning technologies were used in the depth decoding pipeline of SCDeep [2] (a method derived from MWD), no published work in the area of depth-to-RGB encoding has leveraged deep learning throughout the entire encoding and decoding pipeline.

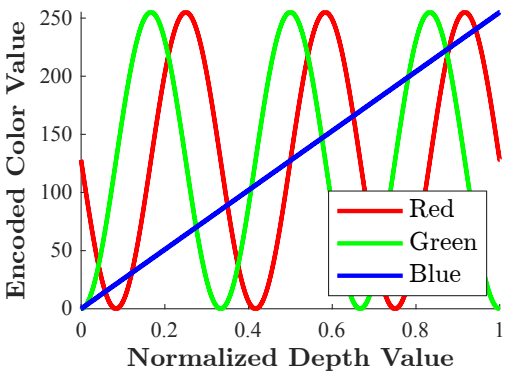

Figure 2. *RGB depth encodings along a normalized depth range by MWD. With $n_{stairs} = 3$, MWD sinusoidally encodes depth information in the red and green channels for 3 periods. The normalized depth map stored in the blue channel assists with phase unwrapping during decoding.*

In contrast with the attempts of MWD to spread depth information across RGB color channels, some image-based depth encoding schemes ameliorate the bit depth problem by using high dynamic range (HDR) image and video codecs. For instance, in Project Starline [3], a seminal 3D telepresence system, the depth streams were simply stored directly within the 10-bit luminance channels of a series of H.265 video streams. However, this of course limited this system to a maximum of only 10 bits of precision for each depth stream.

Methods

Overview

Fig. 1 gives an overview of the full neural depth encoding, compression, and decoding pipeline. Fundamentally, this pipeline needs to take in a depth map, encode it into an RGB image, compress this image with a lossy codec such as JPEG, and then recover the depth map on the other end with as few end-to-end depth losses as possible. Unlike traditional, handcrafted algorithms, our encoding scheme leverages a pair of neural networks for the depth-to-RGB encoding and RGB-to-depth decoding, sandwiched around a differentiable approximation of JPEG compression. This system is, to our knowledge, the first depth-to-RGB encoding scheme that is directly optimized against a lossy image compression codec. Due to the exclusive use of 1×1 convolutions, the encoder and decoder neural networks can be succinctly understood as a pair of multilayer perceptrons (MLP) of similar structure.

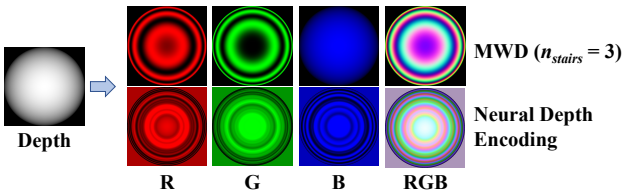

Figure 3. *Lossless depth encodings of the interior of an ideal hemisphere. This figure visually demonstrates how the depth information is spread across the red, green, and blue image channels for both MWD and the neural depth encoding method. Out-of-range values are encoded as black pixels for MWD, whereas the neural depth encoding method chose a light purple.*

Encoding

Starting with a raw depth frame, the depth values are first thresholded to a desired depth range to form Z and then normalized to the range $[0,1]$ to form Z_{norm}. Values within the thresholding range are marked with a 1, while out-of-range values are marked with a 0 in the ground truth mask, which is concatenated to Z_{norm} to form the input for the neural encoder network. The neural encoder employs a series of 1×1 convolutions that encode each depth and mask value pair into three floating point numbers destined for the red, green, and blue color channels. ReLU activations are used in all layers except the final encoder convolution, which uses a sinusoidal activation function. The sinusoidal activation restricts the outputs to the range $[-1,1]$. In addition, it was found experimentally that this periodic function encouraged faster and more optimal solution convergence. Finally, the outputs of the sinusoidal activation function are normalized to the $[0,255]$ range of an 8-bit unsigned integer. Fig. 3 demonstrates how the neural encoder and MWD spread the depth information of the interior of an ideal hemisphere across the red, green, and blue color channels. There are a couple of interesting features to note with the depth encoding methods shown here. First, since the neural depth encoder is not a handcrafted algorithm, it ended up using a shade of purple to denote out-of-range depth values, instead of the more conventional black background used by MWD. Second, it appears that the neural depth encoding oscillates much more rapidly than the MWD encoding, although this does not appreciably hurt its JPEG compressibility, as will be shown in later results.

Compression

In addition to being normalized to $[0,255]$, the outputs of the neural encoder need to be rounded to the nearest integer to simulate the quantization loss that occurs when attempting to store floating point numbers in 8-bit color channels. Unfortunately, rounding is not a differentiable operation, so we need to use a differentiable quantizer proxy. Similar to [4], we elected to use a soft quantizer:

$$x_{soft\_rounded} = round(x) + [x - round(x)]^3 \tag{1}$$

The neural RGB encoding generated at this point is then fed into a differentiable approximation of JPEG, DiffJPEG [5]. Note that we modified DiffJPEG to use bilinear interpolation instead nearest-neighbor interpolation for the chroma upsampling operation in order to more closely follow the JPEG specification. While training, there is no need to actually store a compressed JPEG file, so the lossless compression algorithms used in JPEG (i.e., run-length encoding and Huffman coding) are not explicitly simulated. After encoding and decoding are performed by DiffJPEG, the lossy RGB encoding E is generated.

Decoding

The neural decoder has a similar series of 1×1 convolutions as the encoder and takes the RGB encoding E as input. ReLU activations are used after every 1×1 convolution in the decoder, with the exception of the final convolution, which directly outputs a two-channel image. These two channels correspond to a recovered mask and normalized depth map. During training, Z'_{norm} is recovered by applying the ground truth mask to the recovered depth channel. In contrast, for testing, the system applies a threshold to the recovered mask channel, which is then applied against the recovered depth channel to form Z'_{norm}. It is the recovered mask channel that is responsible for filtering the out-of-range background regions during testing. Lastly, the masked Z'_{norm} is then denormalized back to the original depth range to recover Z'.

Training

To enable the network to learn a compression-resilient encoding scheme across a variety of input depth maps, a diverse data set was chosen. FlyingThings3D [6] is a synthetic dataset of over 25,000 stereo image pairs of digital objects flying along randomized 3D trajectories. The included disparity maps are inversely proportional to depth maps, so they were easily made into an acceptable substitute for training the neural depth encoding and decoding pipeline. The depth map analogs were resized to 224×224 during training. Image histogram equalization was used to ensure relatively even sampling of th e normalized depth range. A randomized threshold was applied to both the lower and upper bound of the depth range, after which the values were normalized. For training, the normalized depth range was expanded to $[-0.01,1.01]$ to improve the accuracy of depth reconstructions at the extrema of the $[0,1]$ testing inference range.

The JPEG quality levels were randomized and uniformly sampled from the range $[85,99.999]$. Periodic calibrations were performed during training in order to equalize the losses across

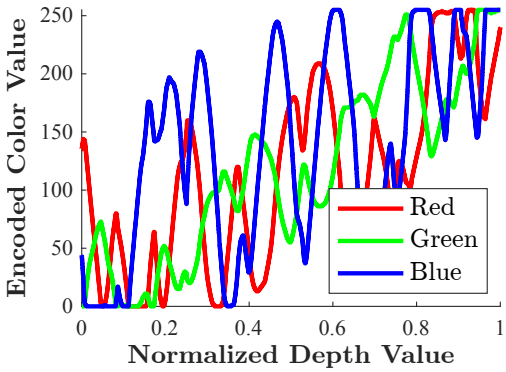

Figure 4. *RGB depth encodings along a normalized depth range by the proposed neural depth encoding method. The neural network was trained through a differentiable approximation of JPEG to produce RGB encoding values that minimize the end-to-end depth reconstruction MAE and the BCE loss of the recovered mask.*

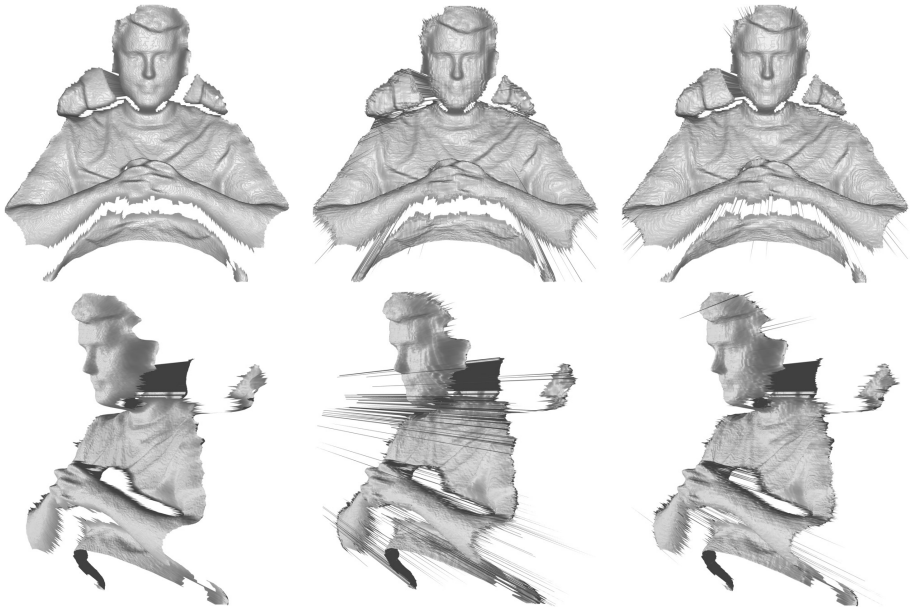

Figure 5. *Top and side view renderings of depth reconstructions generated from a single frame of a 1280×720 Azure Kinect depth video. The ground truth depth frame was encoded into 1280×720 RGB images by both MWD and the proposed neural encoding method. These encodings were then compressed with JPEG at a quality level of 90 using the Python Imaging Library (PIL). Depth values were decoded from the lossy JPEGs and rendered in 3D using MATLAB.*

the JPEG quality range, ensuring that training samples with low JPEG qualities did not dominate the losses and have an outsized impact on the solution convergence. An Adam optimizer with a cosine annealing learning rate scheduler with warm restarts was used to minimize the L1 loss between Z_{norm} and Z'_{norm} as well as the binary cross-entropy (BCE) loss between the ground truth mask and the recovered mask. In practice, the BCE loss was implemented as a single BCE with logits loss layer that includes an integrated sigmoid activation for improved numerical stability. The PyTorch model was trained with an NVIDIA GeForce RTX 3090 Ti with a batch size of 1.

An illustration of how the neural network ultimately learned to encode normalized depth values across the RGB color channels is shown in Fig. 4. This encoding plot is not nearly as orderly as the MWD plot in Fig. 2, but it shows similar kinds of oscillations. This is owed in large part to the sinusoidal activation function, which encouraged a more sophisticated encoding scheme than was otherwise achievable. Despite the rapid oscillations, this neural encoding scheme proved itself to be quite compressible and efficient.

Results

In HoloKinect [7], a 19.4-second depth video of a seated, moving subject was recorded on a Microsoft Azure Kinect at a resolution of 1280×720 as a dataset for quantitative analysis. In this work, we use these video frames to compare the performance of the proposed neural depth encoding method against MWD. The recovered renderings at a JPEG quality of 90 are shown in Fig. 5. Both depth encoding methods demonstrate reasonably accurate interior reconstruction qualities, but MWD suffers from higher errors in regions of steep depth changes such as the cheeks. In addition, MWD exhibits large, visually distracting discontinuities along the edges, which would require further postprocessing to remove. In comparison, the neural depth encoding method shows far fewer discontinuity artifacts and an overall smoother and more natural-looking surface reconstruction. It should be noted that the depth discontinuities present in the reconstructions for both methods can mostly be filtered out as a postprocessing step, but this is computationally expensive and causes some edge erosion.

Fig. 6 compares the aggregate performance of MWD and the proposed neural depth encoding method across the entire video dataset at JPEG qualities ranging from 10 to 99. Overall, the neural depth encoding method has a remarkably lower depth root-mean-square error (RMSE) and mean absolute error (MAE) than MWD across the full range of JPEG qualities. Averaging the results across the JPEG quality range, the neural depth encoding method had a 69.90% lower RMSE and 45.92% lower MAE, and it accomplished this with a 20.02% file size savings. It should be noted that the tunable n_{stairs} parameter of MWD was chosen to be 3 in order to optimize its balance between RMSE and MAE performance to make it as competitive as possible on this dataset. We experimented with n_{stairs} values as low as 1 and as high as 6, but the neural depth encoding method similarly outperforms all of these settings in terms of RMSE and MAE.

Table 1 shows numerical results covering the entire 582-frame video dataset at JPEG quality 90. The neural encoding was more compressible at this quality setting, resulting in an average image size of 48.4 kB, compared to the 62.3 kB average

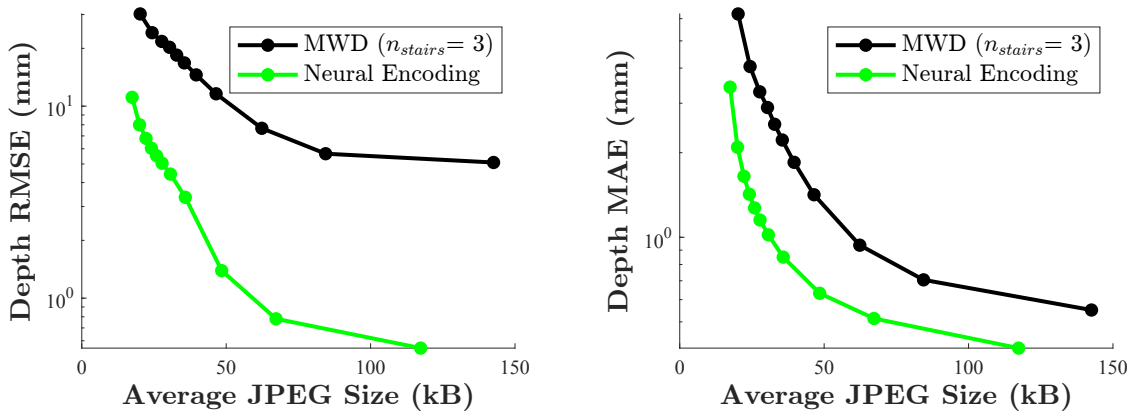

Figure 6. *Quantitative results on a Microsoft Azure Kinect depth video. Both MWD and the proposed neural encoding method were evaluated against an uncompressed, 582-frame 720P depth video taken with a Microsoft Azure Kinect. The thresholded depth range was set to 0.2 m to 1 m, for a total range of 0.8 m. The encoded depth frames were compressed with JPEG using the Python Imaging Library (PIL) before being decoded by each method. A wide range of JPEG qualities were considered (10, 20, 30, 40, 50, 60, 70, 80, 90, 95, 99), with the resulting average file sizes being plotted on the horizontal axes. The MAE and RMSE metrics were taken as aggregate measures over the entire video sequence. In order to limit the influence of differing filtering methodologies, the MAE and RMSE metrics only considered the shared intersection of the depth recovery regions of both MWD and the neural encoding method. MWD utilized an analytically optimized black level thresholding value of approximately 0.3326.*

Table 1. Quantitative depth reconstruction results on Azure Kinect depth video frames at JPEG quality 90. All quantitative measures were taken as aggregates over the entire 582-frame reference video encodings. In order to limit the influence of differing filtering methodologies, the MAE and RMSE metrics only considered the shared intersection of the depth recovery regions of both MWD and the neural encoding method. The file size, depth RMSE, and depth MAE data are a subset of those plotted in Fig. 6.

Depth Encoding Method	Avg. JPEG File Size (kB)	Depth RMSE (mm)	Depth MAE (mm)	Recall	Precision
MWD (n_{stairs} = 3)	62.3	7.6688	0.9371	100.000%	99.974%
Neural Depth Encoding	48.4	1.3912	0.6321	99.984%	99.904%

file size generated by MWD. Despite the larger file sizes, MWD trailed in terms of both RMSE and MAE. MWD suffered from a very large depth RMSE of 7.6688 mm, compared to the relatively small 1.3912 mm depth RMSE of the neural depth encoding method. The large depth RMSE of MWD is largely caused by edge artifacts such as the discontinuities seen in Fig. 5. MWD was more competitive in terms of MAE, but the neural depth encoding method still outperformed it with an MAE of only 0.6321 mm, 32.5% lower than MWD. Both methods did well in terms of recall, which we define as the percentage of ground truth depth measurements recovered without being filtered as out-of-range background. They additionally both did well in terms of precision, which we define as the percentage of recovered depth measurements corresponding to actual ground truth measurements. However, MWD slightly led in terms of recall and precision. Higher values are better for these two metrics in isolation, but MWD's high recall encourages the recovery of the most unreliable measurements along the edges, whereas the neural depth encoding method more aggressively filters these artifacts.

Discussion

The most difficult aspect of training the neural depth encoding pipeline was finding an architecture that would allow for an exploration of the solution space that included the "reuse" of particular color channel values. This was a feature inspired by the periodicity of MWD, which is believed to allow for higher quality reconstructions. Without the addition of the sinusoidal activation function at the end of the neural depth encoder, the encoder always converged on simple, monotonically increasing functions that map normalized depths to encoded red, green, and blue values. Fortunately, the sinusoidal activation allowed a neural network prone to monotonic growth to also reverse the slope of its encoding functions, resulting in a much more efficient and interesting encoding scheme. Although the solution space was more thoroughly explored this way, it is doubtful that any sort of global minimum was discovered.

Ordinarily, rapid color oscillations in an image would mean poor compressibility, but the results on the Azure Kinect video dataset demonstrate that the neural depth encoding method is significantly more compressible than MWD, all without sacrificing reconstruction quality. Some of this may be explained by the color space conversion and the subsequent 4:2:0 chroma subsampling operation of JPEG, but this phenomenon requires further analysis in future work. The current implementation purposefully did not circumvent the color space conversion of JPEG since that kind of low-level control is often unavailable with off-the-shelf image and video compression tools, which would limit the utility of this algorithm. It additionally remains to be seen if this JPEG-trained algorithm can translate well to video compression codecs where temporal stability is also very important. However, the success of similar work [8] makes us optimistic.

Summary

In summary, the proposed neural depth encoding method demonstrates very promising initial results. To our knowledge, this is the first depth-to-RGB encoding scheme directly optimized for a lossy image compression codec. The learned depth encoding scheme consistently outperformed MWD in terms of MAE and RMSE, and it achieved this with compressed image sizes that were, on average, more than 20% smaller. Despite only being trained on JPEG qualities above 85, the proposed method was able to gracefully degrade and outperform MWD across a very wide range of JPEG qualities from 10 to 99. In addition, the 3D renderings make it obvious that the neural depth encoding method suffers from far fewer edge artifacts and shows an overall improved reconstruction quality. Since the unfiltered results are an improvement over MWD, less aggressive postprocessing is required, thus easing the computational burden on lower-performance devices. All of the operations are pixel-wise, so the pre-trained network could be implemented as a pair of simple lookup tables or as a set of high-order polynomial approximations. Future work will involve implementing this algorithm as a drop-in replacement for MWD and evaluating its performance when sandwiched around a video codec.

Funding

This material is based upon work supported by the National Science Foundation Graduate Research Fellowship Program under Grant No. 1945994. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

[1] Tyler Bell and Song Zhang. Multiwavelength depth encoding method for 3D range geometry compression. *Applied Optics*, 54(36):10684, dec 2015.

[2] Matthew G. Finley, Broderick S. Schwartz, Jacob Y. Nishimura, Bernice Kubicek, and Tyler Bell. SCDeep: Single-channel depth encoding for 3D-range geometry compression utilizing deep-learning techniques. *Photonics*, 9(7), 2022.

[3] Jason Lawrence, Dan B Goldman, Supreeth Achar, Gregory Major Blascovich, Joseph G Desloge, Tommy Fortes, Eric M Gomez, Sascha Häberling, Hugues Hoppe, Andy Huibers, Claude Knaus, Brian Kuschak, Ricardo Martin-Brualla, Harris Nover, Andrew Ian Russell, Steven M Seitz, Kevin Tong, and Kevin 2021 Tong. Project Starline: A high-fidelity telepresence system; Project Starline: A high-fidelity telepresence system. *ACM Trans. Graph*, 40:16, 2021.

[4] Xiyang Luo, Hossein Talebi, Feng Yang, Michael Elad, and Peyman Milanfar. The rate-distortion-accuracy tradeoff: JPEG case study. *arXiv preprint arXiv:2008.00605*, 2020.

[5] Michael R Lomnitz. DiffJPEG, 2021.

[6] Nikolaus Mayer, Eddy Ilg, Philip Hausser, Philipp Fischer, Daniel Cremers, Alexey Dosovitskiy, and Thomas Brox. A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4040–4048, 2016.

[7] Stephen Siemonsma and Tyler Bell. HoloKinect: Holographic 3D video conferencing. *Sensors*, 22(21):8118, 2022.

[8] Berivan Isik, Onur G Guleryuz, Danhang Tang, Jonathan Taylor, and Philip A Chou. Sandwiched video compression: Efficiently extending the reach of standard codecs with neural wrappers. *arXiv preprint arXiv:2303.11473*, 2023.

Author Biography

Stephen Siemonsma is an NSF GRFP Fellow and Ph.D. candidate in Electrical and Computer Engineering at the University of Iowa. Stephen specializes in deep learning, virtual and augmented reality, spatial computing, and 3D imaging. He received his B.S.E. in Electrical and Computer Engineering in 2019 and his M.S. in Electrical and Computer Engineering from the University of Iowa in 2021. Previously, he graduated with a B.S. in Biochemistry from the University of Iowa in 2013.

Prof. Tyler Bell is an Assistant Professor of Electrical and Computer Engineering at the University of Iowa. He leads the Holo Reality Lab and is a faculty member of the Public Digital Arts (PDA) cluster. Tyler received his Ph.D. from Purdue University in 2018. His current research interests include high-quality 3D video communications; high-speed, high-resolution 3D imaging; virtual reality, augmented reality; human computer interaction; and multimedia on mobile devices.