Learning-based lossless compression of 3D point cloud geometry

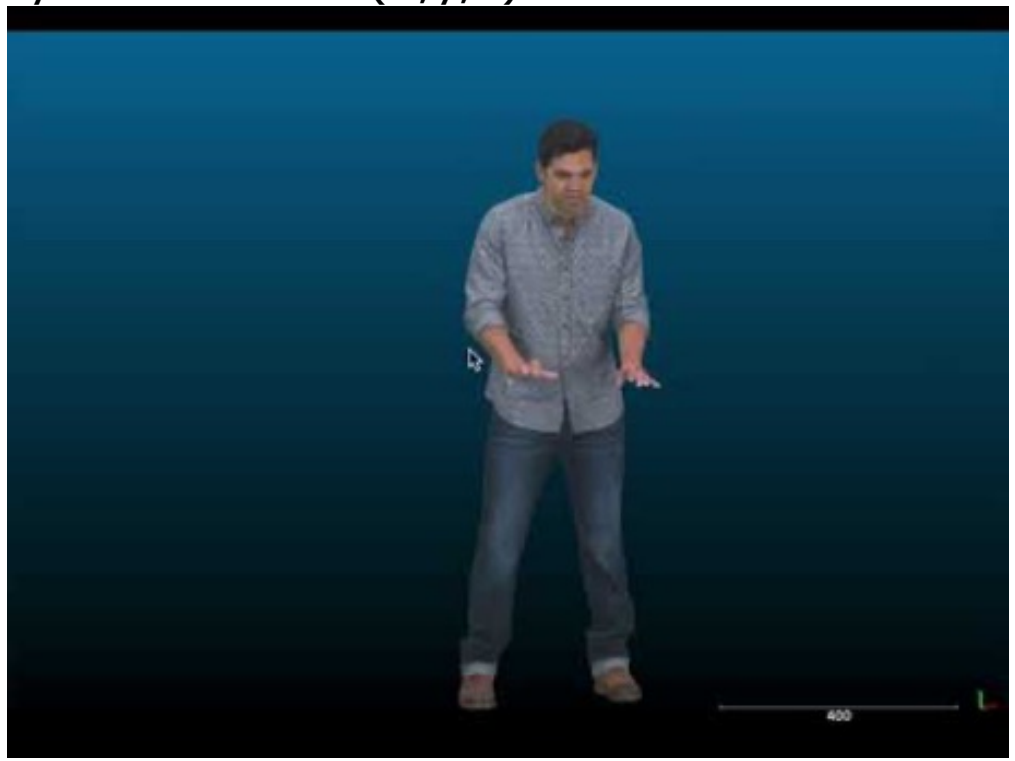
Dat Thanh Nguyen, Maurice Quach, Giuseppe Valenzise, Pierre Duhamel

December 4, 2020

Contents

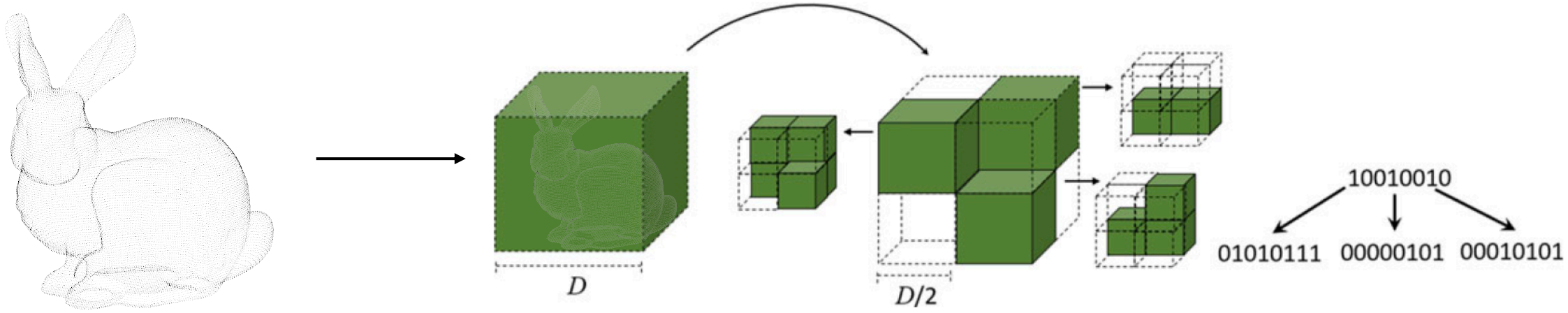
1. Introduction – Point Cloud

- Point Cloud (PC) is the preferred data structure of 3D applications (VR, AR, etc.)
- Each point in 3D space is represented by geometry information (x,y,z) and attributes
- Point clouds are sparse
- 800,000 points -> 1 000 Mbps (uncompressed)
- We focus on **geometry coding**



1. Introduction – PC geometry

- Representations of PC geometry:
  - Voxels: quantize PC into a pre-defined precision, PC → D x D x D voxels, less than 2% voxels are occupied
  - Octree: recursively split voxelized PC into 8 sub-cubes, PC → sequence of octets
- Losslessly encode voxel value using arithmetic coding? **Probability model**



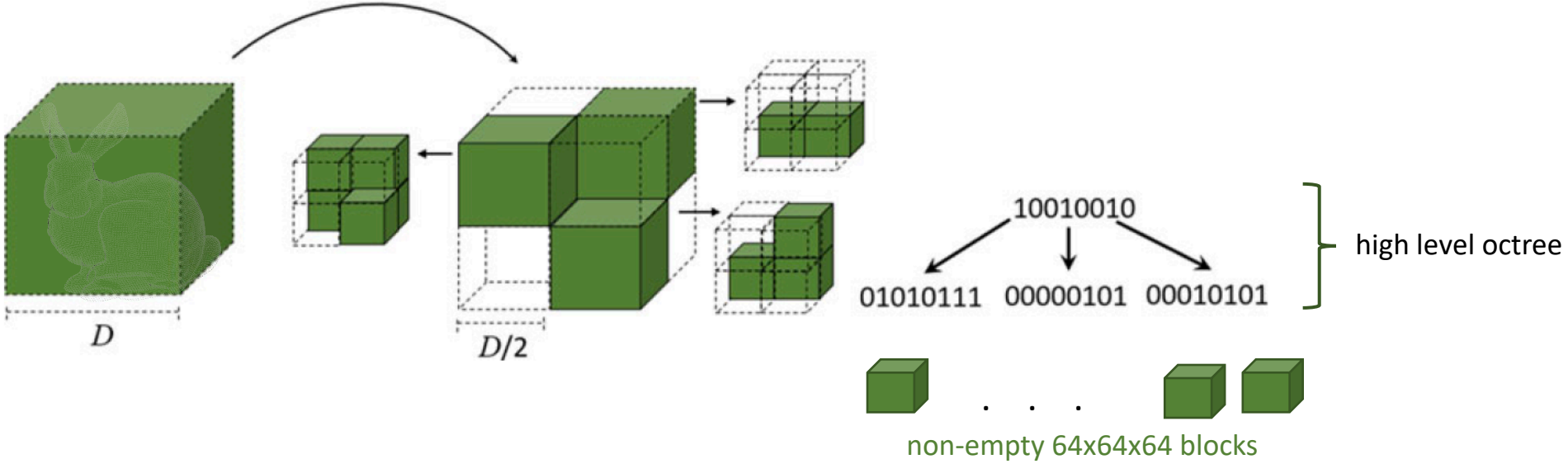
Point Cloud

DxDxD voxelized PC

Octree

1. Introduction – Problem definition

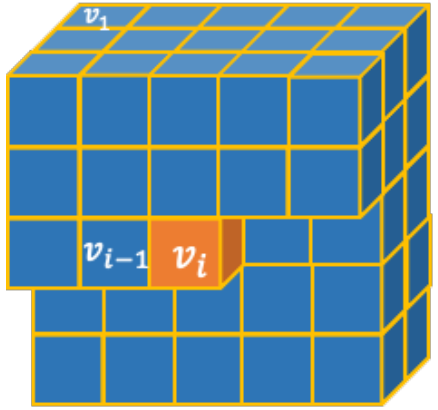
- Partition point cloud into high level octree + non-empty 64x64x64 blocks
- High level octree is transmitted as side information
- We focus on encoding **64x64x64 blocks**



2. VoxelDNN

- Context adaptive arithmetic coding on voxel domain
- Probability model p(v) for each block d x d x d voxels:

$$p(v) = \prod_{i=1}^{d^3} p(v_i | v_{i-1}, v_{i-2}, \ldots, v_1)$$

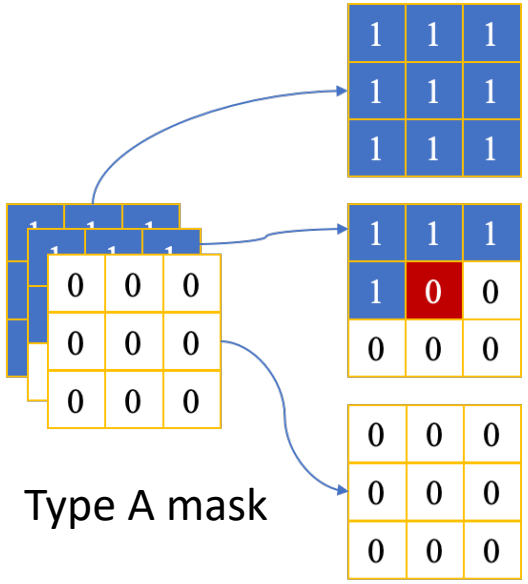


3D voxel context

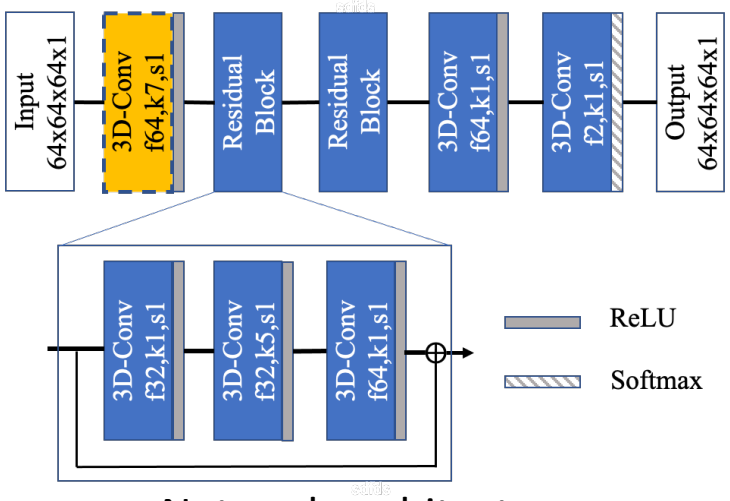
- Estimate each term using a Deep Neural Network:
  - Input: block of d x d x d voxels $(v_1 \; to \; v_{d^3})$
  - Output: conditional probability distribution of each v_i: $\hat{p}(v_i | v_{i-1}, v_{i-2} \ldots v_1)$
  - Causality constraints: $\hat{p}(v_i)$ only depend on previous voxels: $v_{i-1}, v_{i-2}, \ldots v_1$

2. VoxelDNN

- Causality is enforced by using masked filters in each convolutional layer: type A and type B mask [1]
- 2 residual connections to avoid vanishing gradient and speed up convergence
- Cross-Entropy (CE) loss: $$H(p,\hat{p}) = \mathbb{E}_{v\sim p(v)}\left[\sum_{i=1}^{d^3} -\log \hat{p}(v_i)\right].$$



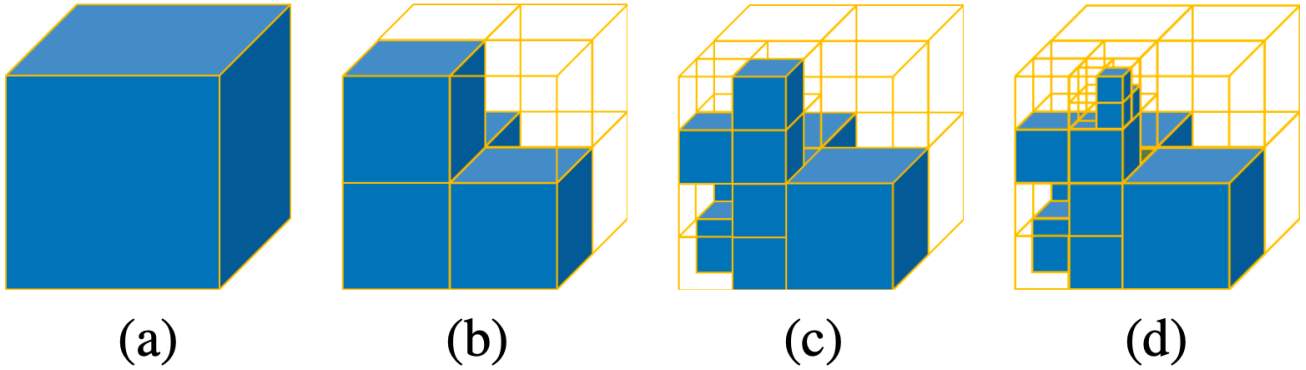
Type A mask



Network architecture

2. Multi-resolution encoder

- Encode voxels in each block sequentially from the first voxel to the last voxel
- Partition blocks into multiple child blocks at different size to eliminate the sparisity
- Rate-optimized multi-resolution algorithm: select the partitioning solution that has minimum bitrate.



From left to right: 1, 2, 3, and 4 partitioning level

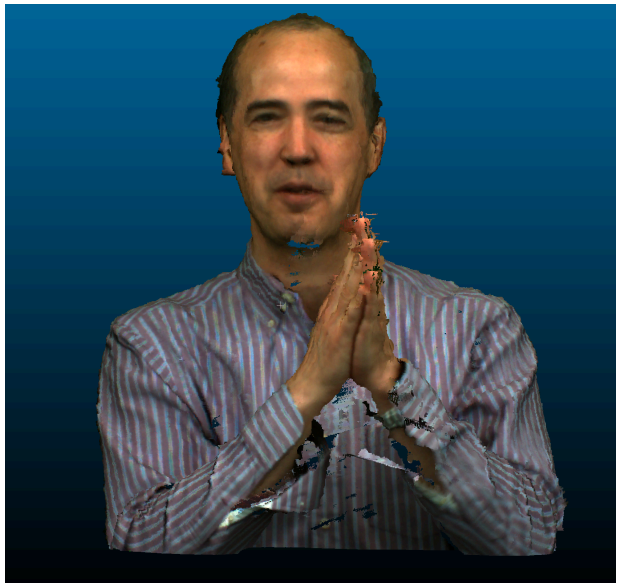
3. Experimental setup

- The training data is mixed from:
  - MPEG 8i*: longdress10, soldier10: 4297 blocks 64
  - Microsoft MVUB*: andrew10, david10, sarah10: 4820 blocks 64
  - ModelNet*: 200 heaviest PCs: 11147 blocks 64
  - Total: 18291 training blocks, 1973 blocks for validation
- Training:
  - 50 epochs with early stopping, batch = 8
  - Adam optimizer, lr=1e-3
- Test dataset: sequences from MPEG 8i and Microsoft MVUB

*MPEG 8i: http://plenodb.jpeg.org/pc/8ilabs
Microsoft MVUB: http://plenodb.jpeg.org/pc/microsoft
ModelNet40: https://modelnet.cs.princeton.edu

3. Experimental setup

- Some testing Point Clouds



Phil 10 bits

Ricardo 10 bits

Loot 10 bits

Redandblack 10 bits

3. Experimental results

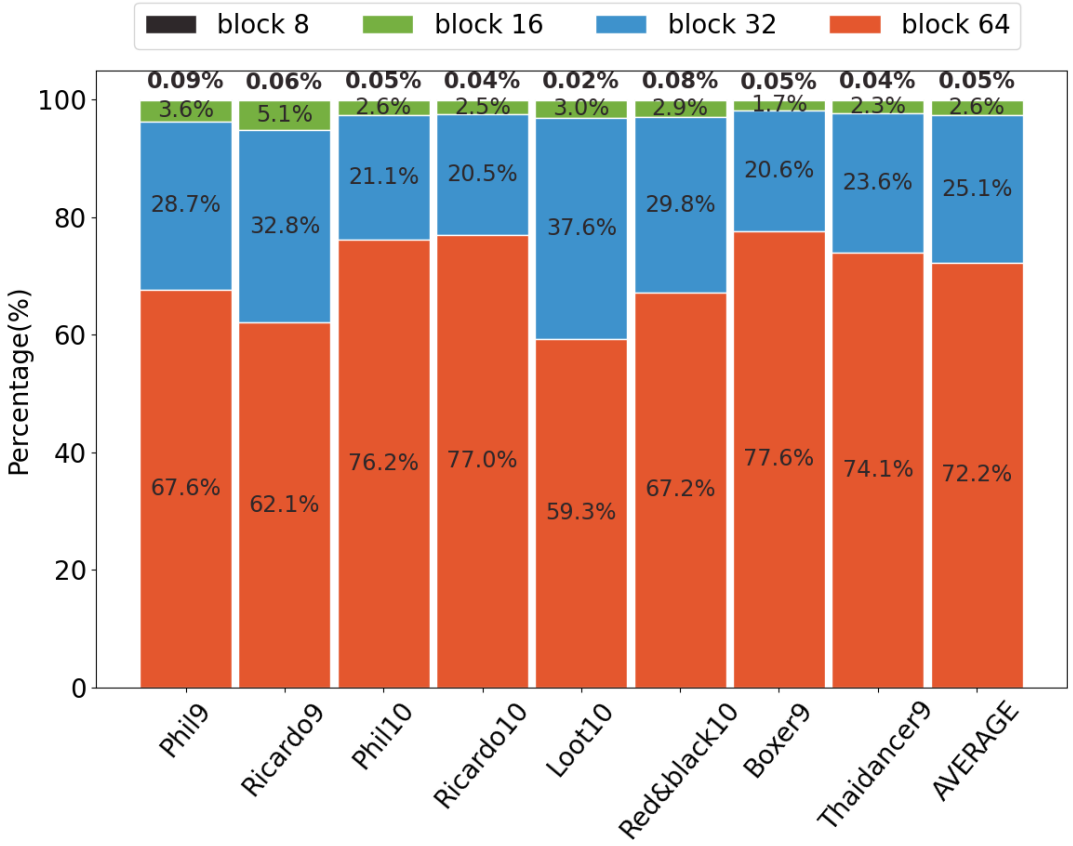
- Average bits per occupied voxel (bpov) for 4 partitioning level: G-PCC[2], P(PNI)[3]

Table 1: Average rate in bpov of VoxelDNN at different partitioning levels compared with MPEG G-PCC and P(PNI).

		P(PNI)	**G-PCC**	**block 64**		**block 64 + 32**		**block 64 + 32 + 16**		**block 64 + 32 + 16 + 8**	
Dataset	Point Cloud	bpov	bpov	bpov	Gain over G-PCC	bpov	Gain over G-PCC	bpov	Gain over G-PCC	bpov	Gain over G-PCC
MVUB	Phil9	1.88	1.2284	0.9819	20.07%	0.9317	24.15%	0.9203	25.08%	0.9201	25.10%
	Ricardo9	1.79	1.0422	0,7910	24.10%	0.7276	30.19%	0.7175	31.16%	0.7173	31.17%
	Phil10	-	1.1617	0.8941	23.04%	0.8381	27.86%	0.8308	28.48%	0.8307	28.49%
	Ricardo10	-	1.0672	0.8108	24.03%	0.7596	28.82%	0.7539	29.36%	0.7533	29.41%
	Average	**1.84**	**1.1248**	**0.8694**	**22.71%**	**0.8142**	**27.61%**	**0.8056**	**28.38%**	**0.8053**	**28.41%**
MPEG 8i	Loot10	1.69	0.9524	0.7016	26.33%	0.6464	32.13%	0.6400	32.80%	0.6387	32.94%
	Redandblack10	1.84	1.0889	0.7921	27.26%	0.7383	32.20%	0.7317	32.80%	0.7317	32.80%
	Boxer9	-	1.0815	0.8034	25.71%	0.7620	29.54%	0.7558	30.12%	0.7560	30.14%
	Thaidancer9	-	1.0677	0.8574	19.70%	0.8145	23.71%	0.8091	24.22%	0.8078	24.34%
	Average	**1.77**	**1.0476**	**0.7886**	**24.75%**	**0.7403**	**29.34%**	**0.7341**	**29.92%**	**0.7334**	**29.99%**

3. Experimental results

- Percentage of encoded points in each block size



4. Conclusion

- VoxelDNN:
  - Hybrid octree/voxel-based lossless compression method
  - The first deep generative model in voxel space
  - Multi-resolution encoder
- 28% gain over MPEG G-PCC standard
- Future works:
  - More powerful generative model
  - Jointly encode geometry and attributes

References

- [1] van den Oord, Aäron, and Nal Kalchbrenner. "Pixel RNN." (2016)
- [2] Schwarz, S., Preda, M., Baroncini, V., Budagavi, M., Cesar, P., Chou, P. A., ... & Llach, J. (2018). Emerging MPEG standards for point cloud compression. IEEE Journal on Emerging and Selected Topics in Circuits and Systems, 9(1), 133-148.
- [3] Garcia, Diogo C., et al. "Geometry coding for dynamic voxelized point clouds using octrees and multiple contexts." IEEE Transactions on Image Processing 29 (2019): 313-322.

Q&A