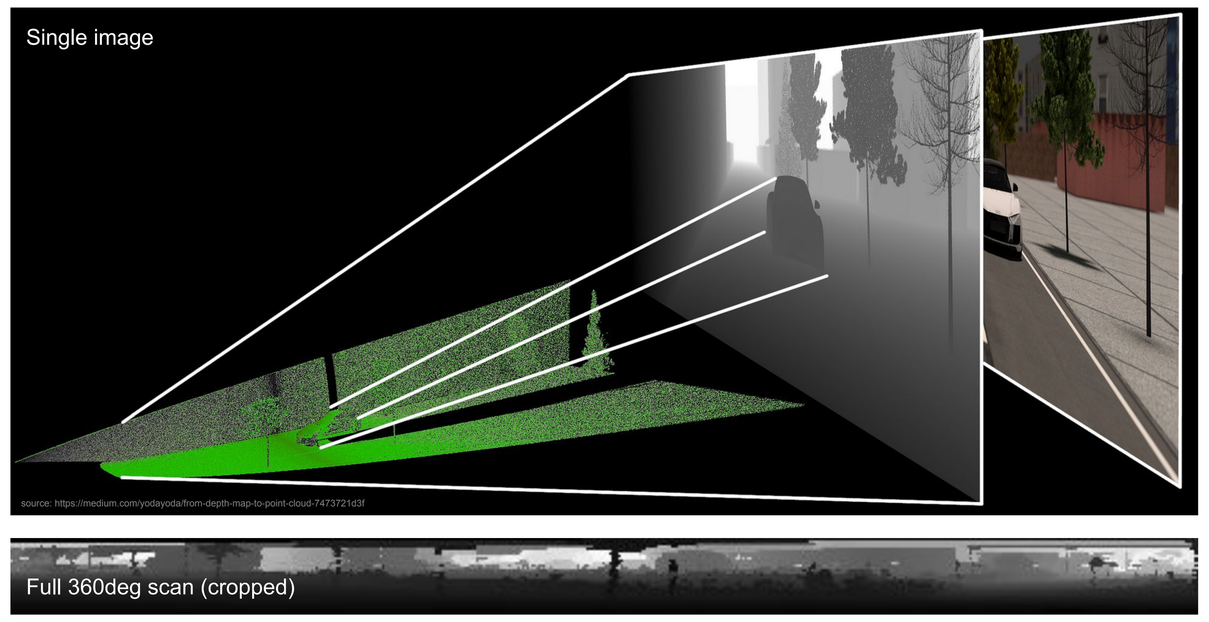
LIDAR 3D POINT CLOUD DEEP COMPRESSION

Esteban Zamora (s222383), Lucas M. Sandby (s223093), Rolando Esquivel-Sancho (s230025), Steven Tran (s223189)
DTU Autonomous Systems, Technical University of Denmark

BACKGROUND

3D point clouds are hard to compress to very low volume directly using generic methods due to their sparsity and unordered structure. Assuming a LIDAR sensor scan, we can leverage the 2D image representation for efficient compression (range, intensity, angle).

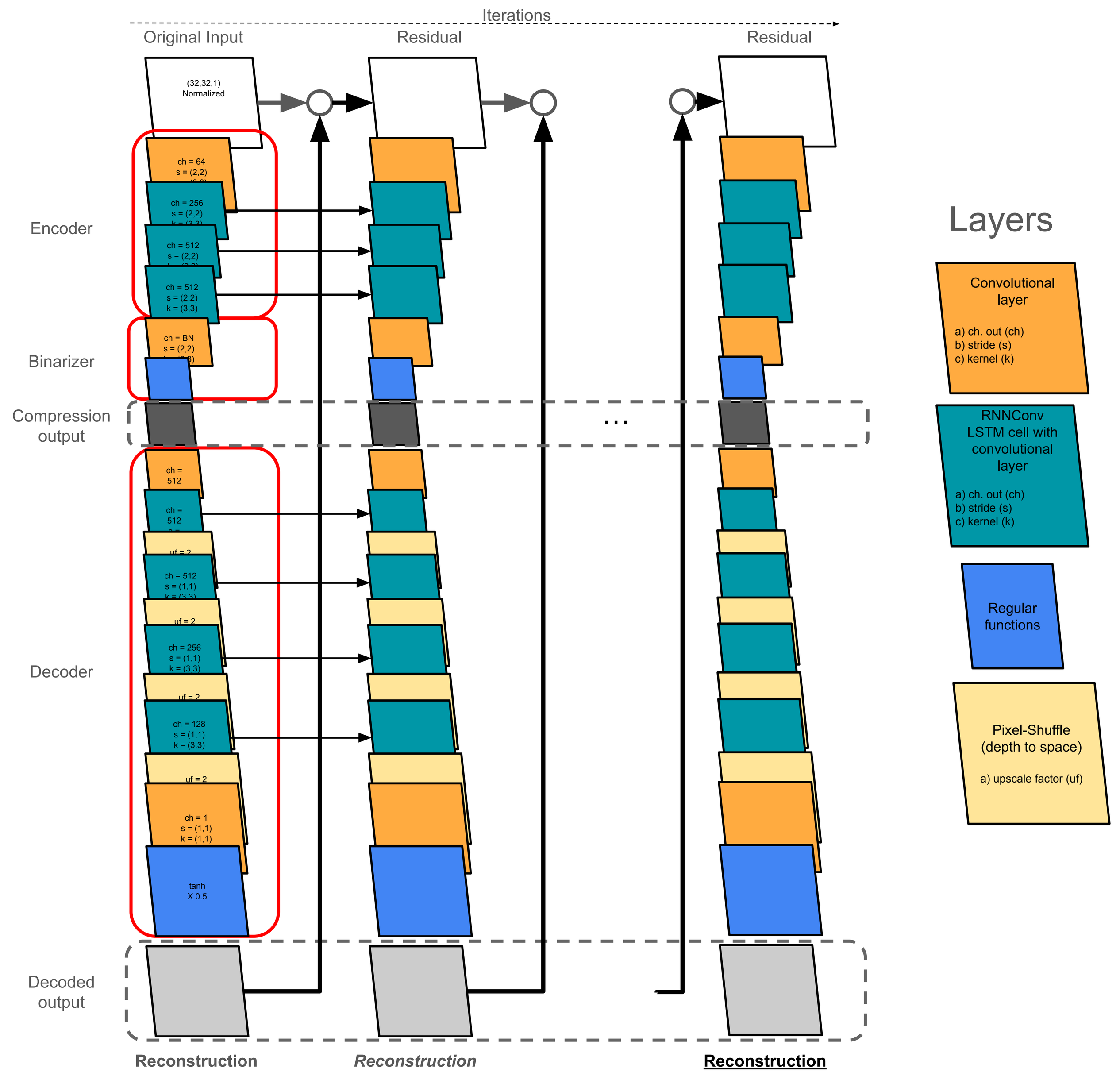


The following is a self-supervised deep compression approach to compress and reconstruct 3D LiDAR scans with decreased loss compared to traditional algorithms (JPEG, Octree, Draco).

Main contributions:

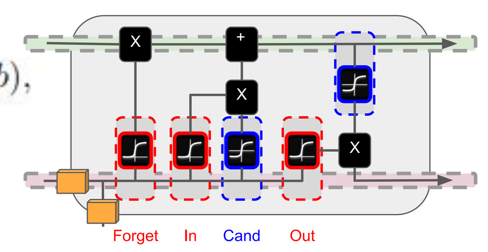
- A convolutional LSTM to compress a point cloud scan from a 3D LiDAR using PyTorch
- Inference using ROS (Robot Operating System)
- Evaluation on the trade-off between inference speed and reconstruction quality in contrast to latent space and network size.

ARCHITECTURE



LSTM CELL

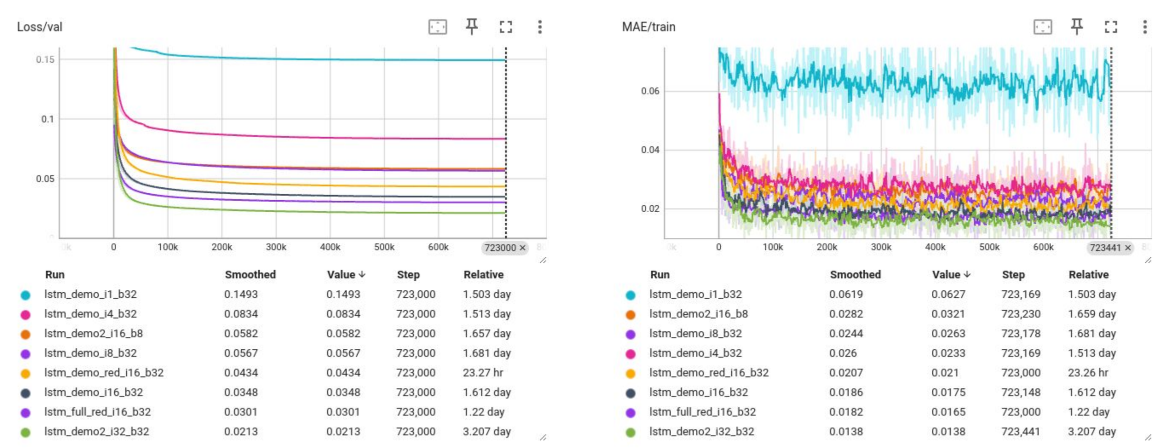
$$[f, i, o, j]^T = [\sigma, \sigma, \sigma, tanh]^T((Wx_t + Uh_{t-1} + b),$$
$$c_t = f \odot c_{t-1} + i \odot j,$$
$$h_t = o \odot tanh(c_t)$$



TRAINING

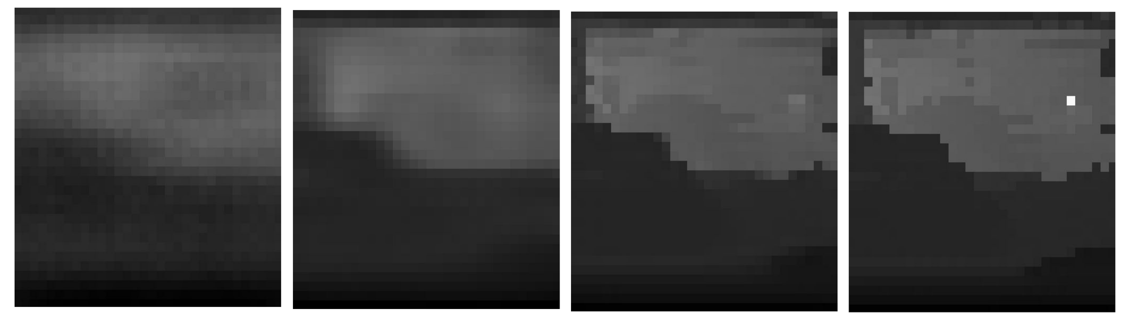
- Dataset: Velodyne LIDAR range images (32x1812)
  - Training: 32300, Validation: 32000
- Random cropping images (32x32)
- Batch size: 128, Number of epochs: 3000
- Adam Optimizer with Cosine Annealing Scheduler
- Loss: Normalized Mean Average Error (MAE)

Training metrics - Tensorboard



Training image reconstruction

E = 1 | E = 10 | E = 100 | E = 1000

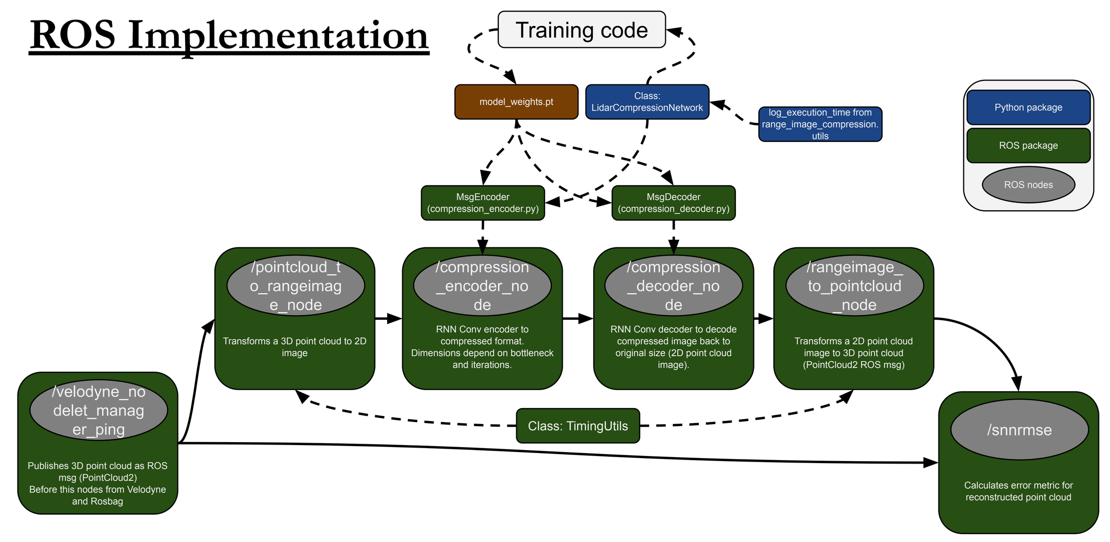


E: Epoch

INFERENCE

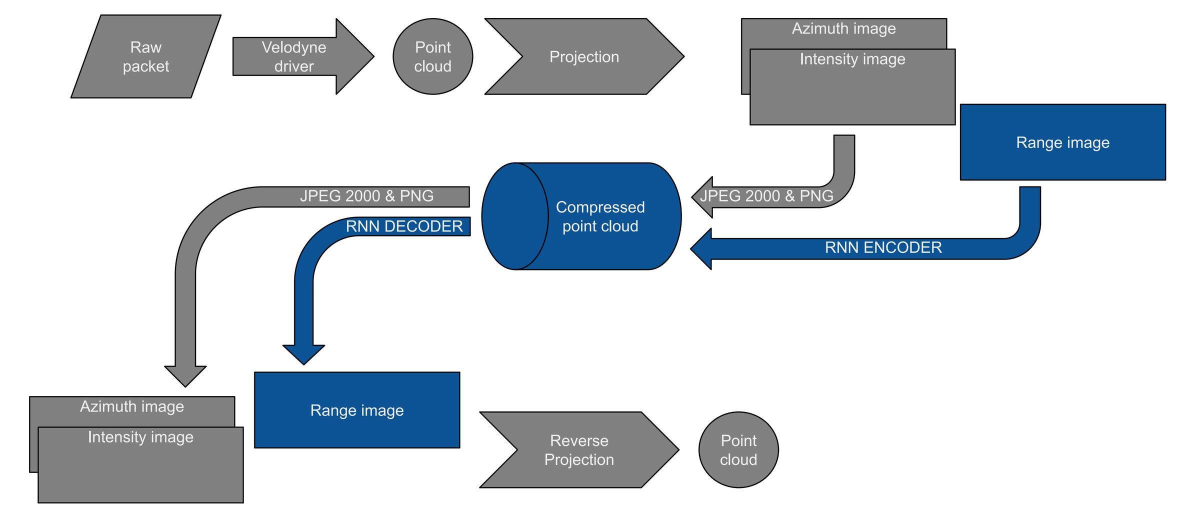
Inference is implemented in ROS and uses the same python package as for training. The test architecture is shown below:

ROS Implementation



Flow Chart

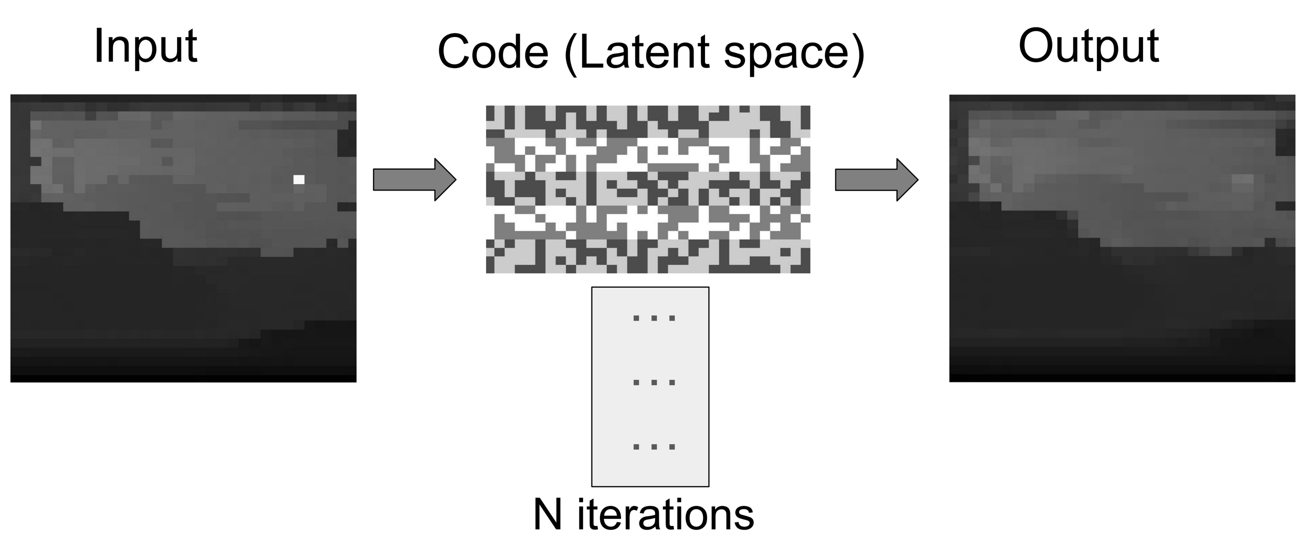
Data flow and sequential steps for inference



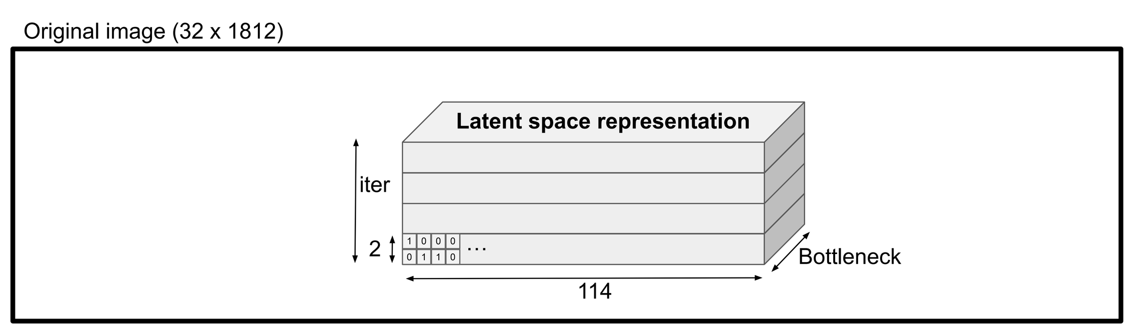
DATA COMPRESSION

- Lossy compression of point cloud range images

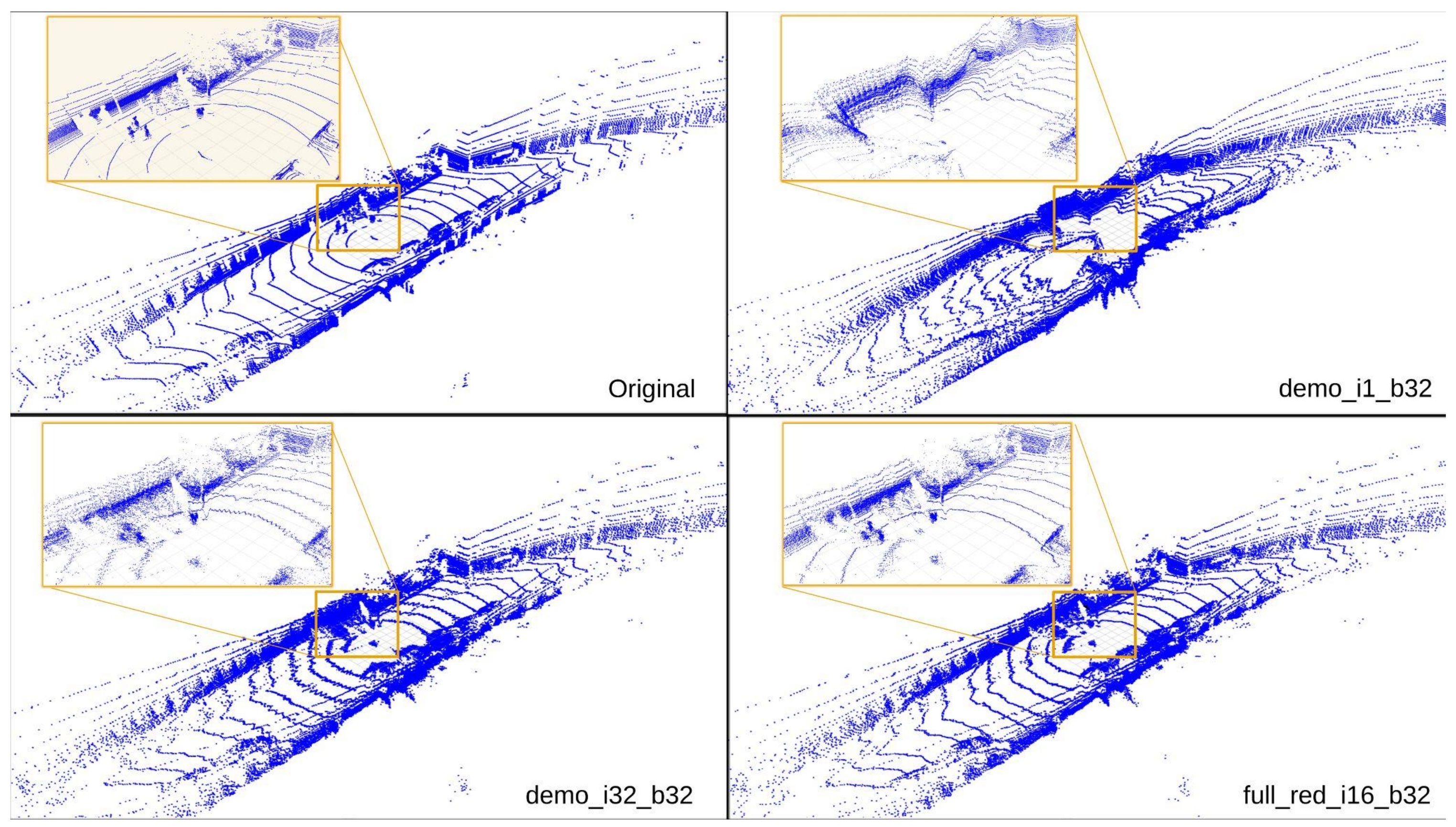
Encoding and decoding



Encoded data-structure

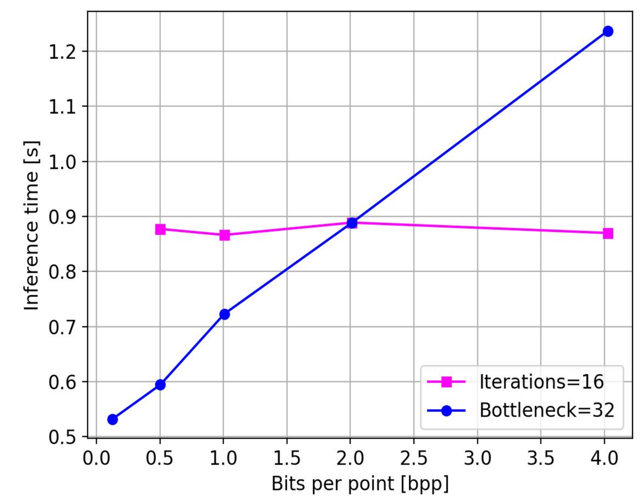


RESULTS - Qualitative

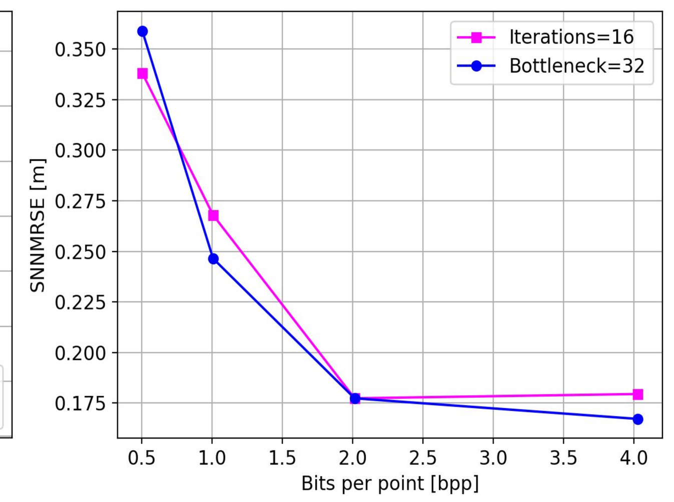


- The larger network (full_ref_i16_b32) shows visually more similarity to the original PCL, this result can also be verified with the quantitative evaluation.
- It is easy to observe the loss of information in the One-Shot Network (demo_i1_b32).

RESULTS - Quantitative

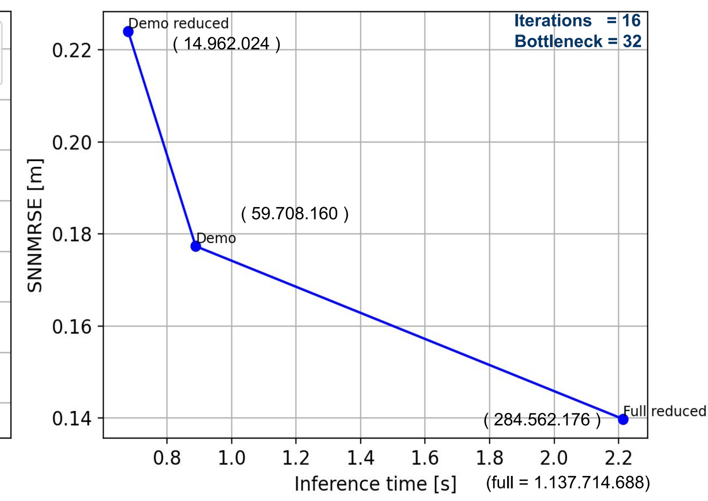


Increasing iterations is the main reason to increased inference time



After 2 bpp the error doesn't decrease as much.

$$\mathrm{SNNRMSE}(P,Q) = \sqrt{0.5 * \mathrm{RMSE}_{\mathrm{NN}}(P,Q) + 0.5 * \mathrm{RMSE}_{\mathrm{NN}}(Q,P)}.$$



The size of the network is correlated positively correlated to inference time and negatively correlated to error.

CONCLUSION

We have successfully implemented a convolutional LSTM to compress a point cloud scan from a 3D LiDAR using PyTorch and ROS for inference. Evaluating our results the error (SNNMRSE) appears to be dependent on the size of the latent space, not specifically the number of iterations or the bottleneck. Also, the error is also dependent on the number of weights, but at the cost of inference time.

REFERENCES

[1] Till Beemelmanns, Yuchen Tao, Bastian Lampe, Lennart Reiher, Raphael van Kempen, Timo Woopen, and Lutz Eckstein, "3d point cloud compression with recurrent neural network and image compression methods," in 2022 IEEE Intelligent Vehicles Symposium (IV), 2022, pp. 345–351.

[2] George Toderici, Damien Vincent, Nick Johnston, Sung Jin Hwang, David Minnen, Joel Shor, and Michele Covell, "Full resolution image compression with recurrent neural networks," IEEE, Conference on Computer Vision and Pattern Recognition (CVPR), 2017.