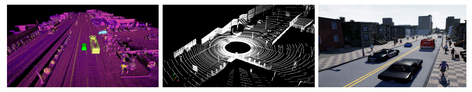
Learning to Generate Realistic LiDAR Point Clouds

Vlas Zyrianov, Xiyue Zhu, and Shenlong Wang
{vlasz2, xiyuez2, shenlong}@illinois.edu

University of Illinois Urbana-Champaign

Generative Modeling of LiDAR Data

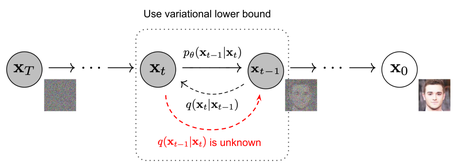
- **Goal**: Build a strong generative model for LiDAR Data
  - Probabilistic prior model of LiDAR point clouds
  - Necessary for building asset-free LiDAR Simulators



- **Desiderata**
  - **Realism.** Ensure point clouds have low sim-to-real gap.
  - **Physical Feasibility.** Generate depth for points that are non-occluded and within sensor FOV.
  - **Asset-free.** Asset collection is expensive, requiring either 3D-modeling or segmenting point clouds.
  - **Controllability.** Allow sampling to be guided by extra info.
- **Previous Work:**
  - Physical based (CARLA, LiDARSim): require assets, unscalable
  - GAN / VAE (Caccia 19): not realistic, limited controllability
- **Proposed Solution**: Diffusion-model based LiDAR Generation on equirectangular view
  - Most realistic learning-based LiDAR Generation to date
  - Allows for controllability and conditional sampling with guided diffusion.
  - Scalable. Does not require collecting assets.

Background: Diffusion Models

- **Diffusion Models** generate clean samples by removing noise.



- For our approach we adopt the NCSNv2 (Song 20) model.
  - **Training** Gradient of the data distribution is learned through a scaled denoising loss objective.

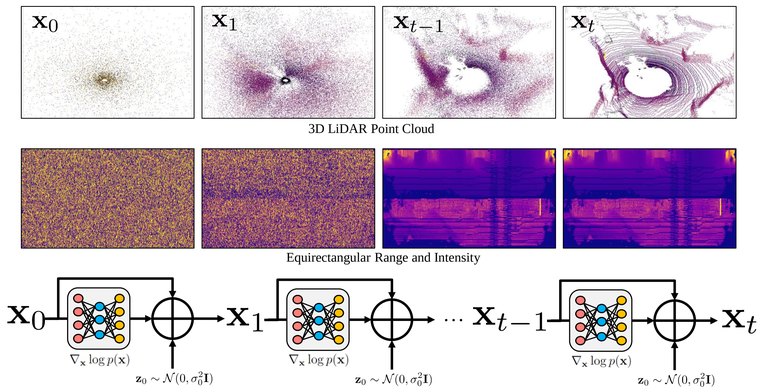
$$\frac{1}{2}\mathbb{E}_{p_{\text{data}}(\boldsymbol{x})}\mathbb{E}_{\tilde{\boldsymbol{x}}\sim\mathcal{N}(\boldsymbol{x},\sigma^2 I)}\left[\left\| s_\theta(\tilde{\boldsymbol{x}}) + \frac{\tilde{\boldsymbol{x}}-\boldsymbol{x}}{\sigma^2}\right\|_2^2\right]$$

  - **Sampling** The underlying distribution is then sampled from with Langevin Dynamics:

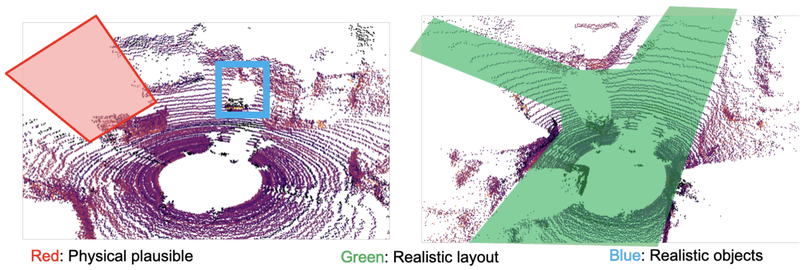
$$\boldsymbol{x}_t = \boldsymbol{x}_{t-1} + \frac{\epsilon_t}{2}\nabla_{\boldsymbol{x}}\log p(\boldsymbol{x}_{t-1}) + \sqrt{\epsilon_t}\boldsymbol{z}_t$$

Approach

We present a novel a score-matching diffusion model for LiDAR point cloud generation. Our model learns to progressively convert a noisy point cloud to a realistic lidar point cloud sample using diffusion models. We use 2.5D equirectangular range view representation for its compactness and physical feasibility



- **Input Representation** To ensure physical feasibility, we use a range-image representation, where each pixel contains depth (using a non-linear logarithmic encoding) and intensity.
- **Circular Convolutions** are used to ensure the left and right boundaries of the panorama are treated as connected neighbors.
- **Positional Encoding** Pixel location matters greatly (e.g., pixels in front of the car typically have greater depth due to the presence of a road). To encode this prior our model the angular coordinate as an additional input to the convolution.



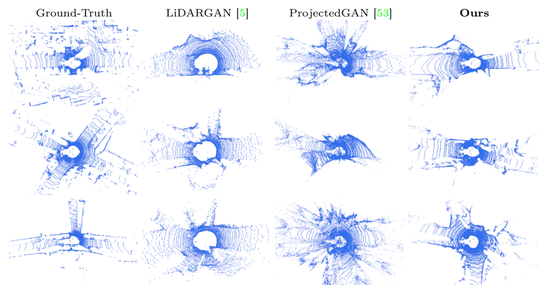
The figure above containing two unconditional samples shows that LiDARGen successfully generates physically plausible readings, containing realistic objects, within realistic street layouts.

Results

- **Quantitative Evaluation** We evaluate our model with BEV MMD, BEV JSD, and a RangetNet++ (Milioto 19) FID score.

	$\text{MMD}_{\text{BEV}}\downarrow$	$\text{FID}_{\text{range}}\downarrow$	$\text{JSD}_{\text{BEV}}\downarrow$
LiDAR GAN	3.06×10^{-3}	3003.8	–
LiDAR VAE	1.00×10^{-3}	2261.5	0.161
Projected GAN	$\mathbf{3.47\times10^{-4}}$	**2117.2**	**0.085**
Ours	$\mathbf{3.87\times10^{-4}}$	**2040.1**	**0.067**

- **Human Perceptual Study** In a human study subjects prefer our method in more than 94% of cases.



Conditional-Generation and Densification

- Learning the LiDAR data distribution $p(\boldsymbol{x})$ with LiDARGen provides a strong prior for performing conditional generation $p(\boldsymbol{x}|\boldsymbol{y})$ without retraining using guided diffusion.

$$\boldsymbol{x}_t = \boldsymbol{x}_{t-1} + \frac{\epsilon}{2}\left(s_\theta(\boldsymbol{x}_{t-1}) + \nabla_{\boldsymbol{x}}\log p(\boldsymbol{y}|\boldsymbol{x}_{t-1})\right) + \sqrt{\epsilon}\boldsymbol{z}_t.$$

- This method allows easily implementing tasks such as LiDAR Densification without retraining.