# ITU AI/ML In 5G challenge: PS-0004: ML Model for depth map prediction

**Team name : SixG\_ISAC**

Team members:

1. Shubham Khunteta
2. Yeswanth Reddy Guddeti
3. Avani Agrawal
4. Ashok Kumar Reddy Chavva

## Problem statement



* Find Depth map of the surrounding using RF data
* RF data: MIMO CIR : Rx antennas × Tx antennas × Delay taps
* Delay sampled at 1.76GHz
* Rx and Tx antennas : 8x8 rectangular array
* Output: LIDAR data: point cloud of the surrounding

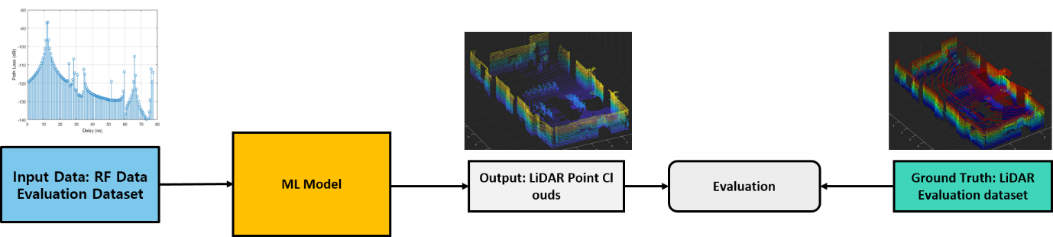


Figure 1 : Problem statement

## Pre-processing of MIMO data:

In the training phase of the model, at each location we use the LIDAR data as the ground truth and train the model to fit MIMO data to that ground truth. In essence, we are attempting to take MIMO channel data that is bi-static (physically separable TX, RX) and fit it to LIDAR data that is mono static (same antenna as TX & RX).

The Pre-processing step handles the deterministic aspects of transforming MIMO channel data which is a function of TX, RX , environment to Lidar data which is function of RX, environment.

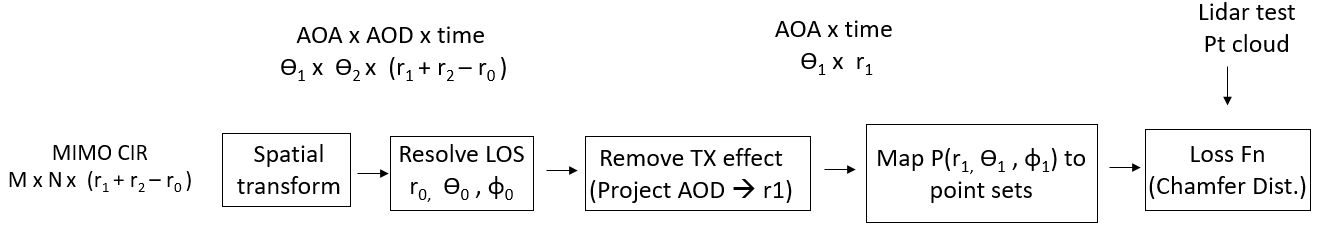


Fig.1 Block diagram of Depth estimation

### Spatial Transform

I/p : channel matrix (nRxAnt x nTxAnt)

O/p : (RxBeamID x TxBeamID)

Input is 64 x 64 channel matrix with each value corresponding to a Rx-Tx antenna pair and the output is 64x64 channel matrix with each value mapped to a AoA – AoD beam pair (RxBeamID-Tx BeamID). As we are using 2D phased array, each beam angle Ɵ has azimuth, elevation (θ,ɸ) components.

Where a 64x64 matrix with each row an antenna weight vectors for each of the 64 AoA beams.

### Find LOS

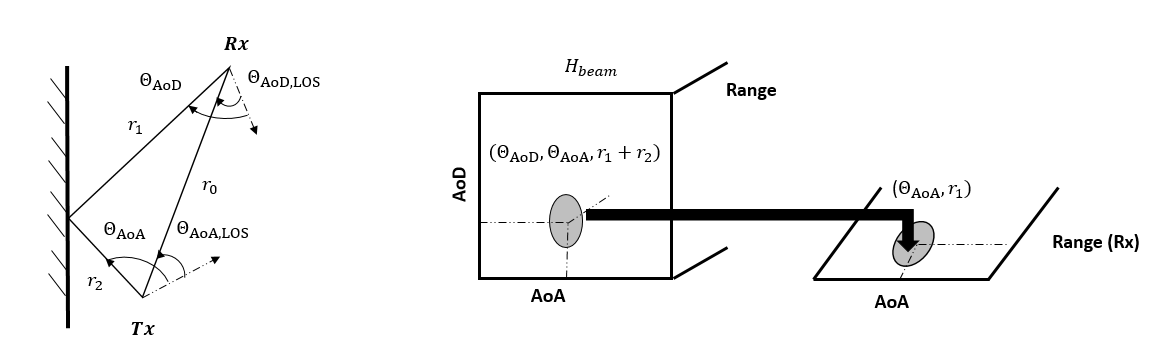
We need to identify ) in . Since Line-Of-Sight path usually has the highest power, the index of highest peak in is treated as LOS path. Store and set

### Remove effect of TX

1. Obtain of the reflections from Rx perspective by projecting entries along AoD axis back in range . This matrix gives a mono-static view of reflections from Rx.
2. Scale the power of reflections in by to convert bi-static path () to mono-static path ().

From figure below, you can see the triangle formed by reflection path & LOS.

Applying sin theorem : .

Note:

Machine Learning Model:

Input for ML model:

Power intensity values (P)) is obtained from the pre-processing block and is used as input for the ML model. Input dimension is 100 x 64 where range dimension has 100 taps and there are 64 are obtained corresponding to the 8 azimuth and 8 zenith beams from the pre-processing block.

Here, we train the model for min\_bound = [-16m, -16m, -2m] and max\_bound =[16m, 16m, 2m] as most of the data is within this limit. We need approximately 100 Range bins in order to detect any obstacle within min\_bound to max\_bound as the Sampling frequency is 1.76 GHz (=sampling time of 0.17ns). Here, we have the 8 directions in azimuth and zenith each, which makes total angular bins 64.

## Output for ML model

**For Training**, We convert the LiDAR PCD into voxel-grid between min\_bound and max\_bound. Voxel sizes are kept 0.25m in one experiment and 0.5 in another experiment. The length of the area in x and y dimension is 32m (-16m to 16m) and in z-dimension, it is 4m (-2m to 2m). With Voxel-size 0.25m, the output grid size is 128x128x16 and with Voxel-size 0.5m, the output grid size is 64x64x8.

**Model Output Data**: if a voxel of the output voxel grid contains a point or points of the Lidar PCD, we mark that voxel as 1 in the output voxel-grid matrix otherwise it is filled with 0.

With voxel-size 0.5, 25% of the elements of output voxel grid are 1 and with voxel-size 0.25, 4% of the elements are 1. With 0.25m voxel size, precision will be more compared to 0.5m if model is trained well for both the cases. With 0.25m voxel size, Output voxel grid size is 8 times higher compared to 0.5m and model need lot of data points for training to stabilize.

**For Testing**, We convert the output grid into LIDAR PCD and evaluate the chamfer distance.

## AI/ML model

We use a decoder kind of model. We first, apply a dense layer on input data which makes the connection with all the elements of input to the hidden layer neurons. And then, we apply Convolutional, Up-sampling layers to fit the grid size. Model and its hidden layer details with activation functions, filters and neurons are shown below in the figure 3.



Figure 3: ML model for depth map prediction

Results:

**Training parameters**: We train with total 3400 samples consisting of 2400 samples of area 1 and 1000 samples of area 3 and evaluate performance on the remaining total of 350 samples of the area 1 and 3. Also, We generate the PCD files for area2. We use Adam optimizer with learning rate of 0.0005 which is decaying at the rate of 0.9 every 10000 steps. We train the model for 50 epochs with a batch size of 32. We use a Custom loss function with binary cross entropy where label ‘1’ gets 20 times higher weightage than the label ‘0’ of the output voxel grid. As the data is sparse, training is skewed towards the label ‘1’.

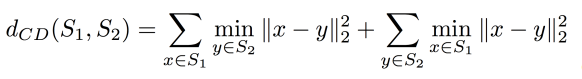
**Post-processing on the predicted output voxel grid:**

We apply two methods to post-process the data and **shortlist the second method** for evaluation which is given below:

Method-1: A threshold based method where if output voxel neuron exceed the intensity of a threshold (0.5-0.8), we mark that voxel as 1 otherwise 0, Here 1 means the obstacle is present and 0 means obstacle is not present.

Method-2: A scanning based method where we scan through all the directions starting from [0m, 0m, 0m] (receiver location) in the predicted output grid. For each direction, the voxel with maximum intensity is marked as 1 and other voxels in the same direction are marked as 0.

**Evaluation:**

We obtain the chamfer distance from the LIDAR PCD as follows: 

Where S1 is the LIDAR PCD data and S2 is the PCD of the predicted output grid. We have cropped the LIDAR PCD data within the range [-16m,16m] in X-dimension and [-16m,16m] in Y-dimension and [-2m,2m] in Z-dimension since more than 95% of the data is present within this bound. Total number of points in the ideal LIDAR PCD are around 33k and in the predicted voxel grid are around 13k. So total points under consideration for the chamfer distance are around 46k.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Avg. Chamfer distance across test samples | Voxel size = 0.5 | | Voxel Size=0.25 | |
| Area 1 | Area 3 | Area 1 | Area 3 |
| Method-1 | 22k | 25k | 24k | 27k |
| Method-2 | 18k | 25k | **20k** | **21k** |

We receive a high accuracy when training has been done area specific. Area-1 chamfer distance is around **10k** when the model has been trained alone with area 1 data.

Note: Predicted LIDAR PCD files for area 2 is provided in the submission files.

## Attached Files:

1. Report
2. Readme file
3. Code files
   1. Pre-processing code files : RF data to Pre-processing : MATLAB code is provided to get pre-processed data which is the input for the ML model
   2. Training file: training using area 1 and area 3 data
   3. Testing file: generate the output PCD for area 2 : Pre-processed data to output PCDs: Python file is provided to get the output PCD of the area-2
4. Data files:
   1. Pre-processed data:
   2. Training weights:
   3. Output PCDs data: