Spring 2023

Location and Environment aware mmWave Beam Selection using Vision Transformer

Srajan Gupta

San Jose State University

Location and Environment aware mmWave Beam Selection using Vision Transformer

A Project

Presented to

The Faculty of the Department of Computer Science

San José State University

In Partial Fulfillment

of the Requirements for the Degree

Master of Science

by

Srajan Gupta

May 2023

The Designated Project Committee Approves the Project Titled

Location and Environment aware mmWave Beam Selection using Vision Transformer

by

Srajan Gupta

APPROVED FOR THE DEPARTMENT OF COMPUTER SCIENCE

SAN JOSÉ STATE UNIVERSITY

May 2023

Dr. Navrati Saxena	Department of Computer Science
Dr. Robert Chun	Department of Computer Science
Dr. Abhishek Roy	Director, Mediatek Inc, USA

ABSTRACT

Location and Environment aware mmWave Beam Selection using Vision Transformer

by Srajan Gupta

5G networks explore mmWave technology to achieve faster data transfer and higher network capacity. The reduced coverage area of mmWaves creates the need to deploy large antenna arrays. However, beam sweeping across a large number of antenna arrays typically involves high overhead and latency. In a vehicle-to-everything (V2X) system, beam selection becomes a frequent process in the case when vehicles are moving at high speed, leading to frequent connection delays. Modern-day vehicular systems are integrated with advanced sensors like global positioning system (GPS), light detection and ranging (LIDAR), radio detection and ranging (RADAR), etc. Machine learning models can be trained using data from these sensors to help predict the optimal beam pair. This paper proposes a novel Vision Transformer (ViT) machine-learning model for beam selection using GPS and LIDAR data. We also introduce a GPS-based Virtual Environment Capture (GVEC) solution to overcome the noise in the LIDAR data. The proposed solution outperforms previous approaches when tested on noisy LIDAR data, achieving an accuracy of 92% while searching among the top 10 beams.

Index Terms: ***Beam-selection, Vision Transformer, mmWaves, vehicle-to-everything, LIDAR, GPS***

ACKNOWLEDGMENTS

I want to express my deepest appreciation to my project advisor, Dr. Navrati Saxena, for her expertise, continuous guidance, and valuable support throughout this project and the rest of my academic journey. I'm also immensely grateful to Dr. Abhishek Roy for his meticulous and detailed review of my work, using his extensive field expertise and time. His insightful feedback and suggestions have greatly improved the quality of my work. I would also like to extend my gratitude to Dr. Robert Chun for his invaluable input and suggestions that have played a critical role in shaping my work. In addition, I would like to take this opportunity to thank my family and friends for their unwavering support and encouragement throughout my academic journey. Their love and support have been a constant source of motivation, and I'm grateful for their belief in me.

TABLE OF CONTENTS

CHAPTER

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

Introduction

The latest cellular wireless technologies, such as 5G and the upcoming 6G [3], rely heavily on millimeter waves (mmWaves). These waves are narrow in nature resulting in more precise targeting of signal, this helps in achieving improved capacity, low interference, and energy efficiency [4]. The usage of narrow mmWaves requires large antenna arrays to achieve sufficient coverage [5]. Unfortunately, in large antenna arrays, the beam sweeping process required for beam selection becomes time-consuming, incurring additional latency that negatively impacts user experience. Furthermore, in order for narrow beams to work properly, it is necessary to have accurate beam alignment between the base station and the user equipment (UE), which is referred to as beam management [6, 7]. Beam management involves selecting the best beam pair, consisting of a transmitter-side beam and a corresponding receiver-side beam, that can offer the best connectivity. One way to select a beam pair is through beam sweeping or beam computing, which involves thoroughly searching through all available beam pairs to identify the one that provides the strongest signal. This process becomes time-consuming, causing connection slowness and delays that negatively impact the user experience. This calls for an efficient beam selection approach. Moreover, modern vehicles require continuous network connectivity for their many essential features, but their mobility makes frequent connection establishment challenging, resulting in frequent disruptions due to beam sweep.

Multiple kinds of research have been done over the last few years on utilizing the sensor data from vehicles namely LIDAR, GPS, RADAR, and cameras to train deep-learning models to predict the best beam. Many milestones in getting higher accuracy with reduced beam sweep time have been achieved as an outcome. But, there still exist some open issues. Unfortunately, most of these existing works rely on error-prone LIDAR data, which does not consider long-range dependencies and is ineffective in heavy rain, snow, and fog [8, 9]. It is because this kind of weather leads to the attenuation phenomenon of the return signal's power and the emergence of false targets, resulting in inaccurate LIDAR data. Given that foggy weather is common in certain regions, an efficient solution for accurate prediction in all weather conditions is needed. This motivates us to look into efficient beam management over noisy data, across all weather conditions. Therefore, new data pre-processing techniques and newer machine-learning models must be explored. In this regard, Google's vision transformer (ViT) model [10] shows promise as a potential solution. ViT is robust to errors in the dataset, as it is designed to capture long-range dependencies in the data through the multi-headed self-attention mechanism [11]. Compared to convolutional neural networks (CNNs), which are highly sensitive to small variations in input data, especially when handling long-range dependencies, Vision Transformer (ViT) models prioritize the most relevant parts of the input sequence while suppressing irrelevant and noisy information. ViT has demonstrated good performance on datasets with noisy or incomplete labels, making it a viable choice for real-world applications where data is often noisy like our case with LIDAR data. We aim to develop a novel approach to the Vision Transformer neural network to train on LIDAR and GPS data to predict top k beams.

The major contributions of this project are as follows:

- We use a novel approach of using a ViT for training on LIDAR and GPS data for beam selection.
- To introduce real-time challenges faced by LIDAR, we have added 25% of records to our test dataset with random noise in LIDAR data.
- To maintain high prediction accuracy irrespective of LIDAR data corruption, we have proposed a novel solution to replace these noisy data records with records

generated using GPS-based Virtual Environment Capture (GVEC).

- We use a novel data preprocessing technique to interpret 3D LIDAR point cloud data as 2D image data.
- We have compared our work with existing works using different metrics and graphs.

The rest of the paper is organized as follows: Section II talks about related work done in the field of mmWave beam selection involving different sensors and different machine-learning approaches. Section III gives details about how our problem setup looks in the real world and also frames the beam selection problem into solving a mathematical problem. Section IV describes our proposed solution and model architecture. Section V talks about GPS-based virtual environment capture (GVEC) which discusses various techniques for capturing environmental details without the use of LIDAR. Section VI gives details about our experimental setup including our selection of dataset, preprocessing approach, and evaluation criteria. Section VII shows the simulation experiments and results of our proposed model based on some reliable metrics and also shows a comparison of our model to the previous works. Finally, Section VIII concludes the paper.

CHAPTER 2

Related Works

To provide context for our proposed approach, we begin with a brief overview of prior research in the field.

2.1 Using Sub-6GHz Channel Information

These techniques focus on using the information extracted from the sub-6 GHz channel to select the best mmWave beam pair. Although sub-6GHz channels and mmWave channels have different characteristics, under some conditions, they have a strong temporal coherence to mmWaves and their information can be mapped to mmWaves. [12] frames the beam selection problem as a multiple measurement vector (MMV) sparse recovery problems. mmWaves have a limited scattering nature and this can be exploited to perform a compressed beam selection. In compressed beam selection, a technique called weighted sparse recovery can be used. [13] used sub-6GHz channel information to train a deep neural network for beam selection. This study employs numerous radio frequency links on the user side and employs digital domain processing techniques to reduce the interference between links. It also uses parallel DNN training techniques to utilize the transfer learning capabilities to reduce the training overhead and delays. [7] uses Power Delay Profile (PDP) which is one of the channel state information available in sub-6GHz channels. One of the advantages of using PDP was this information was available even before the mmWave links were established which will be the case in real-time.

2.2 Using GPS Data

Given the surrounding and the road side units (RSUs) are fixed, the best beam pair for a specific position will remain constant. The position data can be obtained using a GPS sensor. GPS sensors are available in all UE's, be it our cellular phones or vehicles. [14] uses the GPS coordinates of the UE to predict the best beam pair.

It feeds the location data obtained using GPS to different models and compares the accuracy of all models. Lookup tables, k-nearest neighbors (KNN), and neural networks were the three different algorithms used. Lookup tables are nothing but mapping table, which maps every uniform square region to the best beam pair that was calculated previously using exhaustive beam search. For every square region, the beam selected most frequently for the cells within that region using exhaustive search is assigned as the optimal beam pair in the lookup table for that region. But machine learning algorithms like KNN and neural networks were more smart and learned from patterns rather than some static table for prediction. The results showed that the neural networks algorithm performed the best followed by KNN, and the lookup table performed the worst. But this approach was only useful for line-of-sight (LOS) based conditions and does not take into account the reflections from the environment, also in the case of cellular phones it does not take into account the orientation of the people using them which might lead to the wrong best beam pair selection. [15] uses both the position as well as the orientation of the user to train the deep neural network for beam selection. But there still exists the open issue of the reflection from the dynamically changing environment.

2.3 Using LIDAR Data

LIDAR stands for light detection and ranging and is a complex sensor available in modern vehicles used to capture the 3D image of the environment. LIDAR emits laser beams when these beams hit some obstacles and reflect back, it captures the reflection time, and using this it generates a 3D image of the surrounding environment also known as the 3D point cloud. If this LIDAR data is combined with GPS data to train deep learning algorithms, both position and surrounding reflections will be taken into account during predictions leading to better accuracy in both LOS and

Non-LOS conditions. [16] uses LIDAR and GPS for training recurrent neural network (RNN) models for beam selection. [17] uses CNN and a distributed LIDAR-based architecture. In this architecture instead of every vehicle having its own expensive LIDAR sensor, we have a centralized LIDAR at the base station which captures the environment including the vehicles in its vicinity. The accuracy of this work was less compared to other works where LIDAR is mounted on each vehicle, but its setup cost is much lower requiring just one LIDAR. [2] uses CNN along with a non-local attention mechanism, it also uses a novel knowledge distillation loss function which helped it in getting higher accuracy. [18] uses a federal learning approach to train the CNN model. In federated learning, all the vehicles collect the sensor data periodically and send it to the base station, which uses this data to further train and fine-tune the model. The model weights after fine-tuning are sent back to the vehicles. During normal operation, the vehicles can use these weights to predict the best beam pairs locally. Thus, the training is done in a live and distributed manner while predictions are done locally. While previous research using LIDAR sensors has yielded promising results, these studies assume that LIDAR data is free of noise. In reality, LIDAR sensors often struggle to function accurately in adverse weather conditions such as rain, fog, and snow [8, 9]. Given that these weather conditions are quite common in many parts of the world, this can limit the effectiveness of LIDAR-based systems. Additionally, LIDAR sensors are unable to detect reflections from shiny surfaces, which can further limit their accuracy. Finally, the cost of LIDAR sensors remains a significant factor to consider. To address these challenges, researchers must develop solutions that can improve the accuracy of sensor-based systems, even in adverse weather conditions.

2.4 Using Camera

[19] uses RGB images from the camera mounted at the RSUs for the beam and blockage prediction. First, it uses transfer learning to train the ResNet-18 model to detect the user in the scenario, if detected it returns the best beam pair for that user location using the pre-defined codebook. If the user is not detected it classifies it as a blockage. This approach can predict blockage but did not have the capability to select the best beam pair in case of blockage. [20] proposes a similar approach but adds GPS information to get a more accurate position of the user in a non-blockage situation. [21] uses the image taken from ordinary cameras to build a panoramic point cloud view using a 3D scene reconstruction technique. This way it can take into account the buildings and other obstacles in the surroundings while training using the Deep Neural Network. But this being an offline approach works only when the environment remains static like buildings and does not work for dynamically changing environments like roads with running vehicles. [22] uses RGB images to detect the UEs and then uses an angle prediction model to detect the angle between the UEs and the RSUs. Finally, it combines the angle information with the code book information to predict the best beam pairs. Camera based systems are not as accurate as LIDAR in capturing environment, also any blockage to the sight of the camera lens can lead to failure in capturing the sight of the environment.

CHAPTER 3

System Model And Problem Formulation

Fig. 1 shows the system model for our problem. There are two major components in the system, one is the roadside units (RSUs), which are the base stations located on the curbside of every street, and another one is the vehicles which are the user equipment here with the antenna mounted at the center on the roof. The RSU is connected with all vehicles in its area using 5G mmWave orthogonal frequency division multiplexing (OFDM) subcarriers. Antennas are assumed to be analogous in nature.

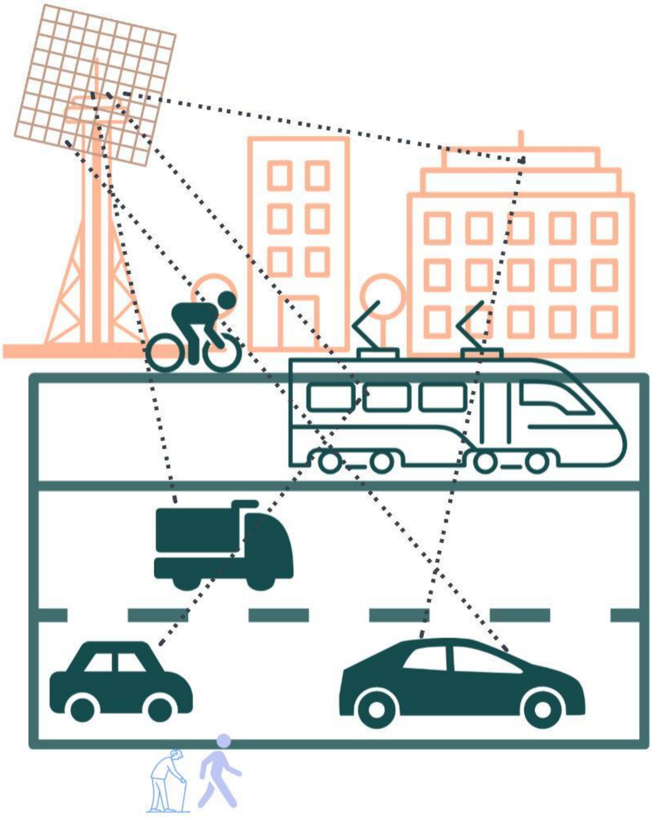

Figure 1: System Model

We assume the codebook to be fixed in nature and these are denoted as

$$\mathcal{C}_t = \{\mathbf{f}_i\}_{i=1}^{C_t} \, at \ the \ transmitter \ end$$
$$\mathcal{C}_r = \{\mathbf{w}_j\}_{j=1}^{C_r} \, at \ the \ receiver \ end$$

There are $(\mathrm{i}, \mathrm{j}) \in \mathcal{C}_t \times \mathcal{C}_r$ pairs of the precoder and the combiner vectors. $\mathbf{H}_k$ is a matrix representing the channel between the transmitter and the receiver over the kth subcarrier [23] calculated as

$$\mathbf{H}_k = \sqrt{N_t N_r} \sum_{\ell=1}^{L} \alpha_\ell \mathbf{a}_r \left(\phi_\ell^A, \theta_\ell^A\right) \mathbf{a}_t^* \left(\phi_\ell^D, \theta_\ell^D\right),$$

Here α_ℓ is the complex path gain, N_t is the number of antennas at the transmitter, and N_r is the number of antennas at the receiver. ϕ_r^D, ϕ_r^A are the azimuth angle for departure and arrival respectively. θ_r^D, θ_r^A are the elevation angle for departure and arrival respectively. For each pair, we need to estimate the channel gain [23]. The channel gain at the kth subcarrier is given by:

$$y_{(i,j)} = \mathbf{w}_j^H \mathbf{H}_k \mathbf{f}_i$$

For some specific pair (i,j) the sum of channel gains over all subcarriers is given by:

$$y_{(i,j)} = \sum_{k=0}^{K-1} \left|\mathbf{w}_j^H \mathbf{H}_k \mathbf{f}_i\right|^2$$

Here y is the channel gain, which is the received signal power at the receiver. $\mathbf{w}_j^H$ is the complex conjugate transpose of a vector $\mathbf{w}_j$ represents the weights used by the receiver to combine the signals received from different antennas. $\mathbf{H}_k$ is a matrix representing the channel between the transmitter and the receiver. $\mathbf{f}_i$ is a vector representing the weights used by the transmitter to transmit the signal from its multiple antennas. Here the beam selection problem can be said to choose the receiver-transmitter pair that maximizes the above channel gain. This is represented by the below equation

$$\widehat{(i,j)} = \arg\max_{(i,j)} y_{(i,j)}$$

The beam sweeping method requires searching through a vast number of pairs $\mathcal{C}_t \times \mathcal{C}_r$, causing significant connection delay. To improve user experience, we can use metadata from sensors like LIDAR and GPS to predict the top k pairs, among which the optimal beam pair will likely be found. This reduces the search to only k pairs, saving time.

CHAPTER 4

Our Proposed Solution

4.1 Vision Transformer Architecture

The 2021 ICLR conference research paper titled ”An Image is Worth 16*16 Words: Transformers for Image Recognition at Scale” introduced the vision transformer (ViT) model [10]. This model can be used for various image recognition tasks, such as object detection, image segmentation, image classification, and action recognition. ViTs are able to do this by converting images into sequences of patches, which are then flattened into vectors and projected to the desired input dimension. This allows the model to learn image structure independently, which can improve its performance on these tasks. In addition, ViTs are also able to learn long-range dependencies between different parts of an image, which can also improve their performance.
There are multiple steps in a vision transformer architecture as shown in Fig. 2.

1. Split the input image into fixed-size patches, These patches resemble the word tokens in NLP problems.

2. Flatten the patches to form a linear sequence.

3. Embed the flattened patches to create low-dimensional linear embedding.

4. After this, in order to retain the positional information, positional embedding is added to the patch embedding. The positional embedding can be done in 1D or 2D but we did not observe any significant performance difference between the two therefore we will stick to using 1D.

5. Feed the sequence generated by this embedding as an input to the transformer encoder.

6. Pre-train the ViT model on a large dataset with output labels in case of a classification task.

7. Fine-tune the model.

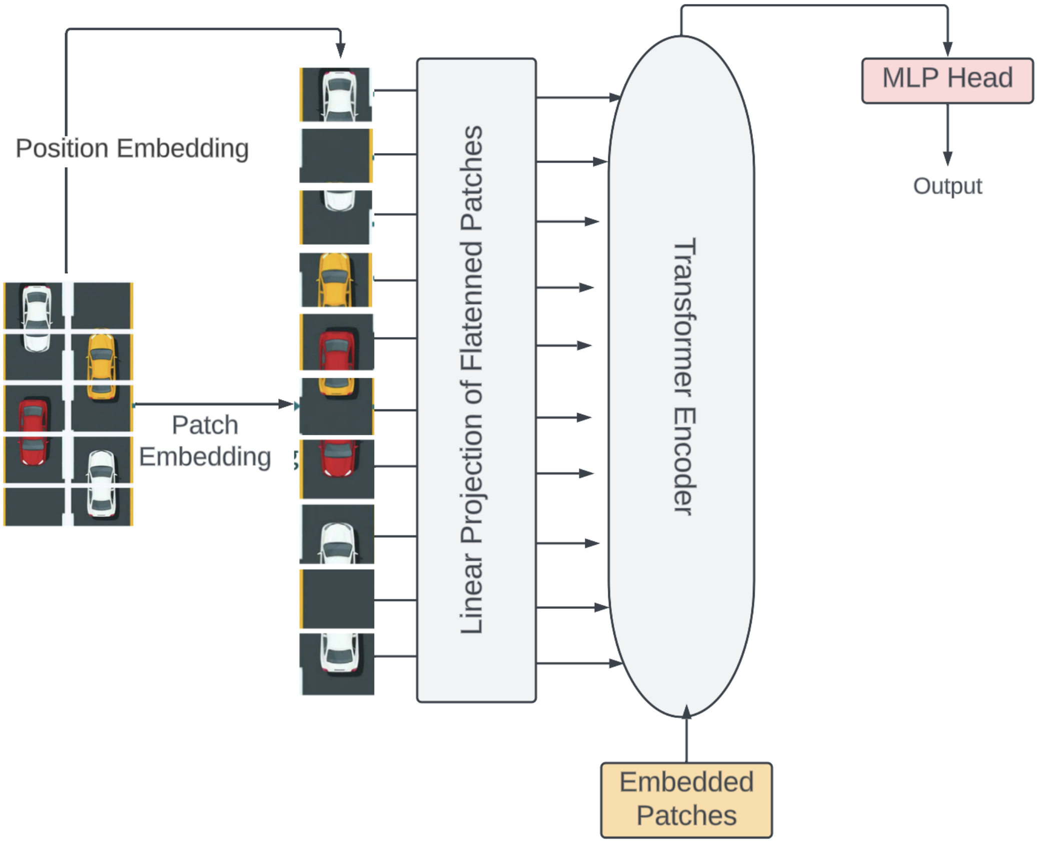

Figure 2: Vision Transformer Architecture

4.2 Vision Transformer Encoder

The encoder part of the transformer is one of the most important components, and we will discuss here the internals of it in more detail. It consists of three main components as seen in Fig. 3:

1. Layer Normalization

There might be variations present in the input images present in training data. The normalization layers help to stabilize the model during the training process.

2. Multi-headed self-attention

Multi-headed self-attention is the main component of the ViT encoder block. It helps the model to focus more on the relevant parts of images and learn more from them. For finding the relevant parts of the image an attention map is created. An attention map is a matrix that assigns weights or scores to every element or token in the input sequence during the computation of the self-attention layer. These weights determine the relative importance of each element or token to the final output of the layer. In a normal self-attention mechanism, each vector within an image interacts with all other vectors to capture dependencies between them. But in ViT we have multi-heads in self-attention which means that it can assign each head to work on different parts of the image instead of a single head working on the whole image. This leads to more effective learning of the heads.

3. Multi-layer perception (MLP)

After the self-attention layer, the MLP component is applied to each encoder block. Using this, the representations learned by the multi-head attention block are transformed into higher-level abstraction. These higher-level abstractions are more useful for downstream tasks.

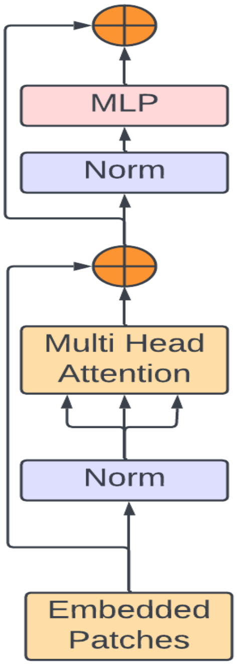

Figure 3: Transformer Encoder Architecture

4.3 Vision Transformer Multi-Headed Self-Attention

The architecture diagram of the Multi-Headed Self-Attention layer can be seen in Fig. 4. The Self-Attention layer in a ViT starts by converting the input images into three components: queries, keys, and values denoted by Q, K, and V respectively.. Formally for input z containing N images of dimension D, $Q = zW_q$, $K = zW_k$ and $V = zW_v$. Here W_q, W_k and W_v are learnable matrices. After this, an attention score A is calculated which signifies how much correlation exists between two input sequences and their respective queries and keys [10]. The softmax function is used here to bring the attention score values in the range of 0-1.

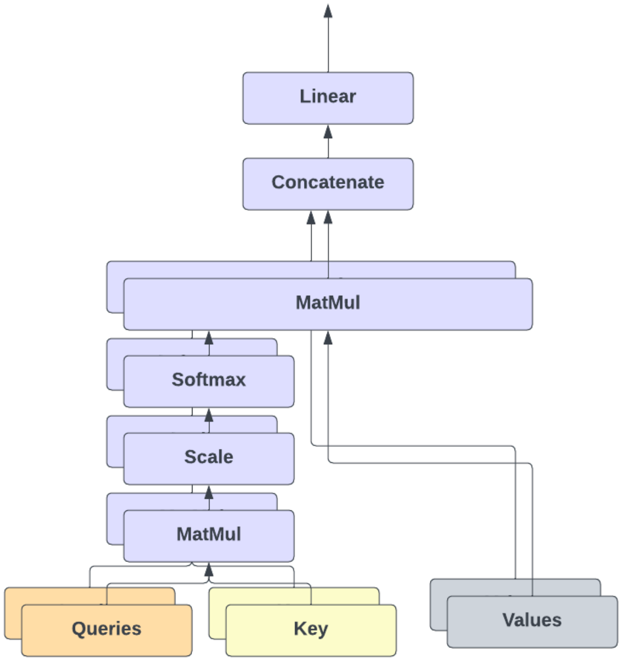

Figure 4: Multi-headed Self-Attention Architecture

$$A = \text{softmax}\left(\mathbf{QK}^\top / \sqrt{D_h}\right)$$

Finally, we add up the weighted sum across all positions after multiplying the attention score with value embedding to obtain the output of the self-attention block [10].

$$\text{SA}(\mathbf{z}) = A\mathbf{V}$$

The multi-head attention mechanism as seen in Fig. 6 is a further extension of self-attention seen in Fig. 5 that allows the ViT to simultaneously focus on various regions of the image, and to learn the intricate relationships between these regions. It is obtained by concatenating the self-attention matrices and multiplying them by another linear transformation, $Umsa$, to produce the final output of the MSA mechanism [10]. In Fig. 5 we can see attention mechanism focuses more on the person in the picture, whereas in Fig. 6 we can see that, there are three heads of the Multi-headed attention, each one of them focuses on different parts, first one focuses on the person, the second one focuses on the clouds, and the third one focuses on the mountains.

$$\mathrm{MSA}(\mathbf{z}) = [\mathrm{SA}_1(z); \mathrm{SA}_2(z); \cdots ; \mathrm{SA}_k(z)] \, \mathbf{U}_{msa}$$

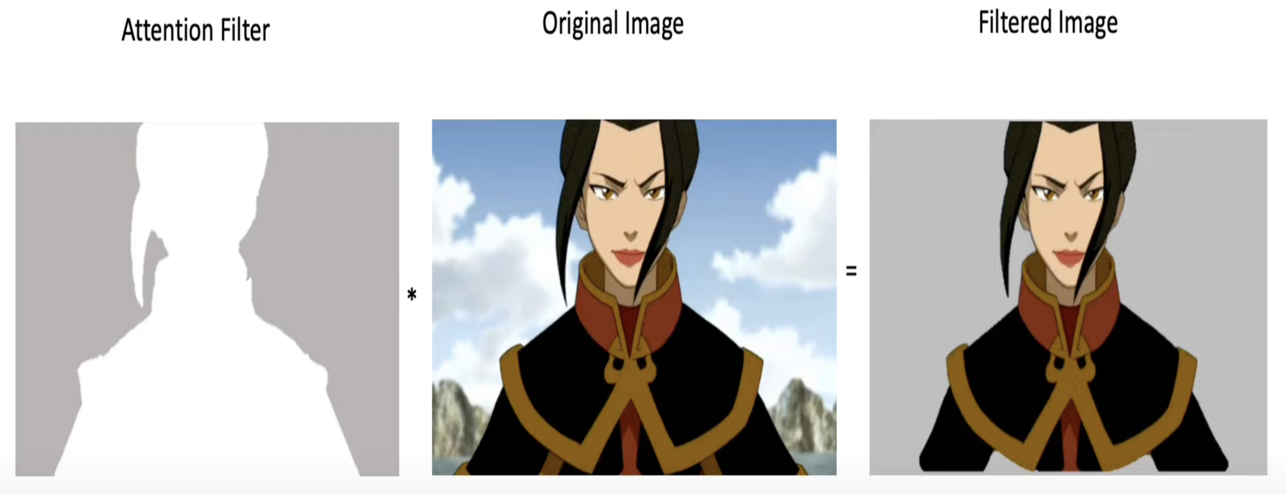

Figure 5: Attention Mechanism example [1]

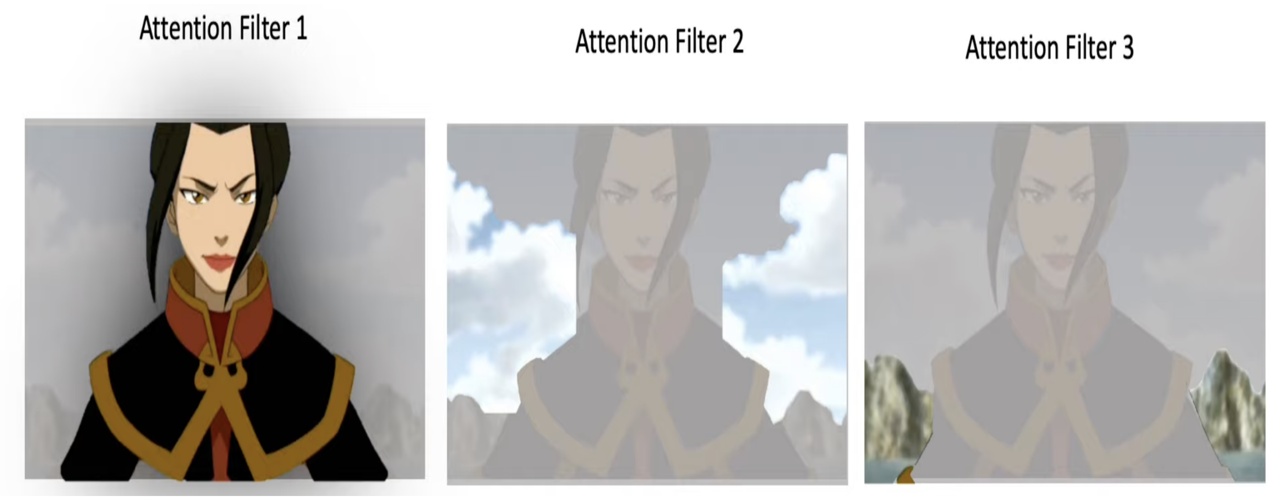

Figure 6: Multi-headed Attention Mechanism example [1]

4.4 Our Final Model - Combining GPS data to ViT

After the layers comprising the ViT architecture, we add the GPS data. GPS is 3-dimensional data containing the X, Y, and Z coordinates of the current vehicle.

The Z coordinate was not much helpful in improving our model accuracy, therefore we dropped it to reduce the number of features. Once GPS data is appended, a few linear layers are added along with the ReLU activation function to train on the GPS data, followed by a final dense layer to create the final output of dimension 256x1 denoting the predicted channel gains for each beam pair. The block diagram of the final model can be seen in Fig. 7.

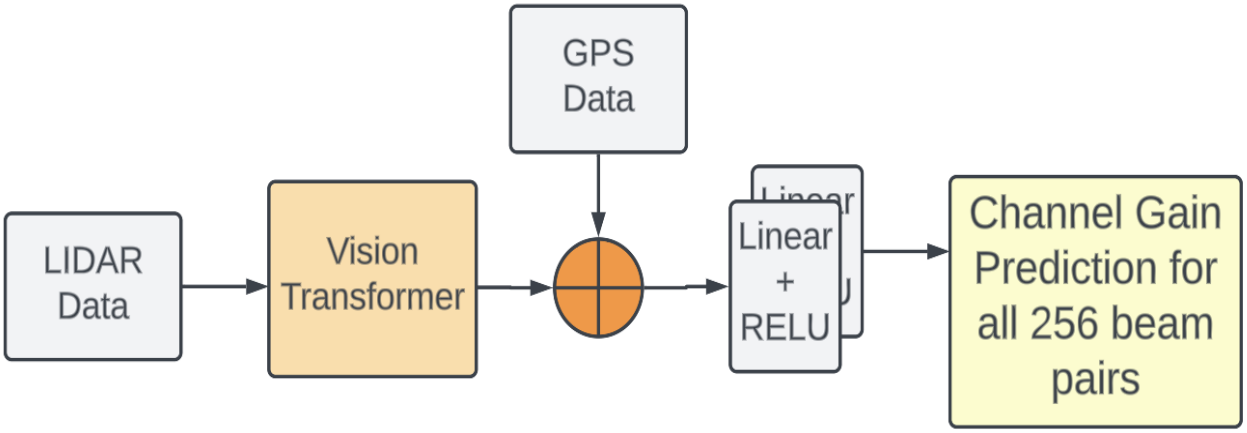

Figure 7: Final Model Block Diagram

4.5 Our Model Parameters

Table 1: Parameters used for training ViT

Index	**Hyperparameter Name**	**Value**
1	Input image dimension	20,200,3
2	Reshaped Input image dimension	72,72,3
3	Dimension of Patches	6,6,3
4	Number of Patches	144
5	Type of Positional Embedding (1D or 2D)	1D
6	No. of Heads	4
7	Transformer Layers	4
8	Loss Function	KL Divergence
9	MLP layer Dimension	64, 32
10	Learning Rate	0.001 (Adam Optimizer)
11	Epochs	200

4.6 Vision Transformer(ViT) vs Convolutional Neural Network (CNN)

Vision Transformers, introduced in 2020 by [10], revolutionized image classification tasks due to their multi-headed self-attention concept, precise understanding of data embedding, and tolerance to noisy data compared to traditional CNN models. CNNs treat all pixels equally, but transformers calculate varying importance for each pixel, making them better suited for tasks where different parts of the input data have varying importance. In mmWave beam selection using LIDAR, foreground objects like other vehicles and obstacles are more important than background objects like buildings, trees, and empty streets. However, CNNs apply the same filters to all pixels, leading to less accurate detection. Moreover, CNNs can only capture short-range dependencies, while the relationship between vehicles at a larger distance can be lost. ViT, which can capture long-range dependencies, is a better choice in this scenario. Having Multi Heads in self-attention is particularly beneficial for mmWave beam selection using LIDAR as it allows the model to attend to multiple input features simultaneously, combining them for better decisions. In contrast, regular attention can only focus on one feature at a time, potentially leading to suboptimal results. So specifically in our case where there will be noise present in LIDAR data, all the above-discussed properties of ViT will make it perform better than CNN.

CHAPTER 5

GPS-BASED Virtual Environment Capture (GVEC)

As we have discussed previously the 5G mmWave beams are directional and narrow in nature. Therefore the beam alignment should be proper between the RSUs and the UEs for good connectivity. In a stable environment, it is easy to establish this alignment. But in streets where a lot of cars, trucks, ambulances, and pedestrians, the beam path gets hindered by all these obstacles making the alignment process complex. Further adding up to the complexity, many vehicles will have reflecting surfaces, so the beam will not only be hindered but also reflected in another direction creating new beam trajectories. Among all these obstacles some would be stationary like buildings, signboards, etc., and won't make any effect on pre-learned beam paths. But surrounding vehicles will be moving and can be of different sizes and shapes, therefore their dynamic positions must be taken into consideration. To record the surroundings one way is to use sensors like LIDAR. The LIDAR sensor is an expensive device that may not be available in all vehicles, and it does not perform well in inclement weather conditions such as extreme fog, snow, and rain [8, 9]. Therefore there is a need to find a solution that can record the surroundings without using LIDAR. To improve the realism of our experiment we increased our test dataset size by adding 25% (2400) additional records that have random noise present in the LIDAR data. These new records were randomly sampled from our existing test data, and the LIDAR data was made noisy by overwriting it with randomly generated 0s or 1s using a Python random number generator function. To address the problem of errors in the data, we proposed a novel solution called *GPS Based Virtual Environment Capture (GVEC).*

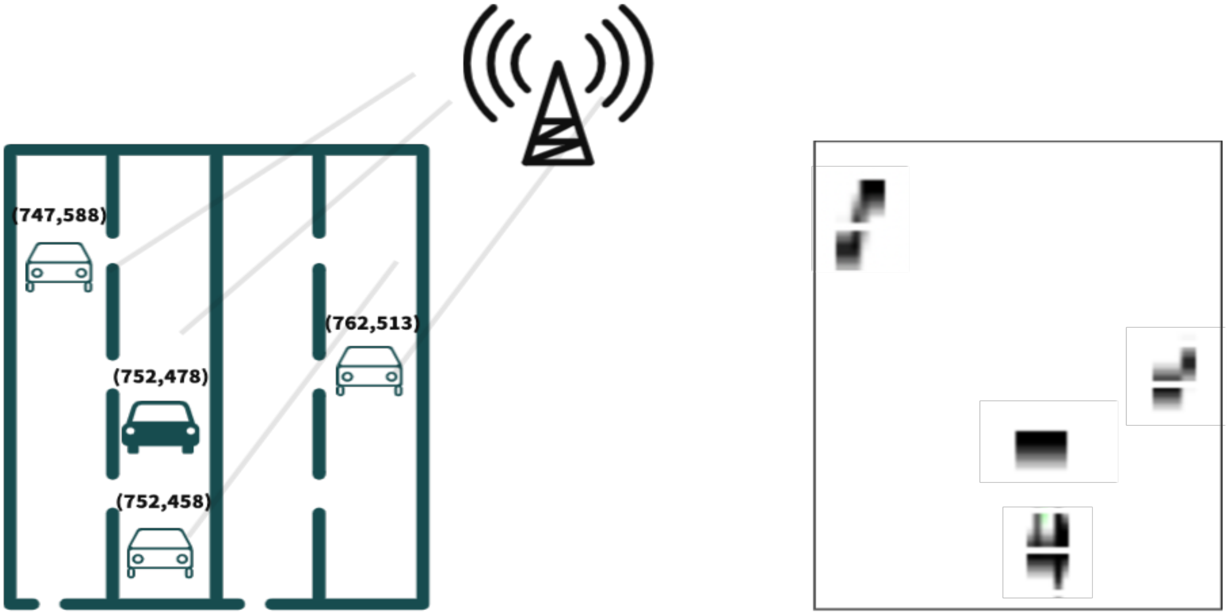

Figure 8: GPS-BASED Virtual Environment Capture (GVEC)

Fig. 8 shows a GVEC scenario. In GVEC all the vehicles that are close to each other are assumed to be connected to the same base station. These vehicles on a low bandwidth channel can share their GPS coordinates, type, size, etc. with the base station. The base station can use this information to create a virtual image of the environment. There are different encoding schemes in which this information can be recorded. Our system is a four-lane road environment but the schemes we will be discussing below are equally eligible to other road environments too.

5.1 Using Grid-Based Representation

GVEC Grid-Based Representation involves three steps:

1. First, we calculate the relative positions of nearby vehicles in quantized dimensions of $[20, 200, 1]$.

2. Next, we mark the empty matrix with 1's at the positions where the neighboring vehicle may be present in the $[20, 200, 1]$ grid. The given environment is a 4-lane

road, we assume each vehicle captures 5 points (= 20/4) across rows and 21 points across columns in a 2-dimensional grid. A sample representation can be seen in Fig. 9.

3. Finally, we convert the 2D grid into an RGB image. Later ViT converts images into patches as seen on the right side in Fig. 8. These steps aid our proposed framework to replace noisy LIDAR data with GVEC data.

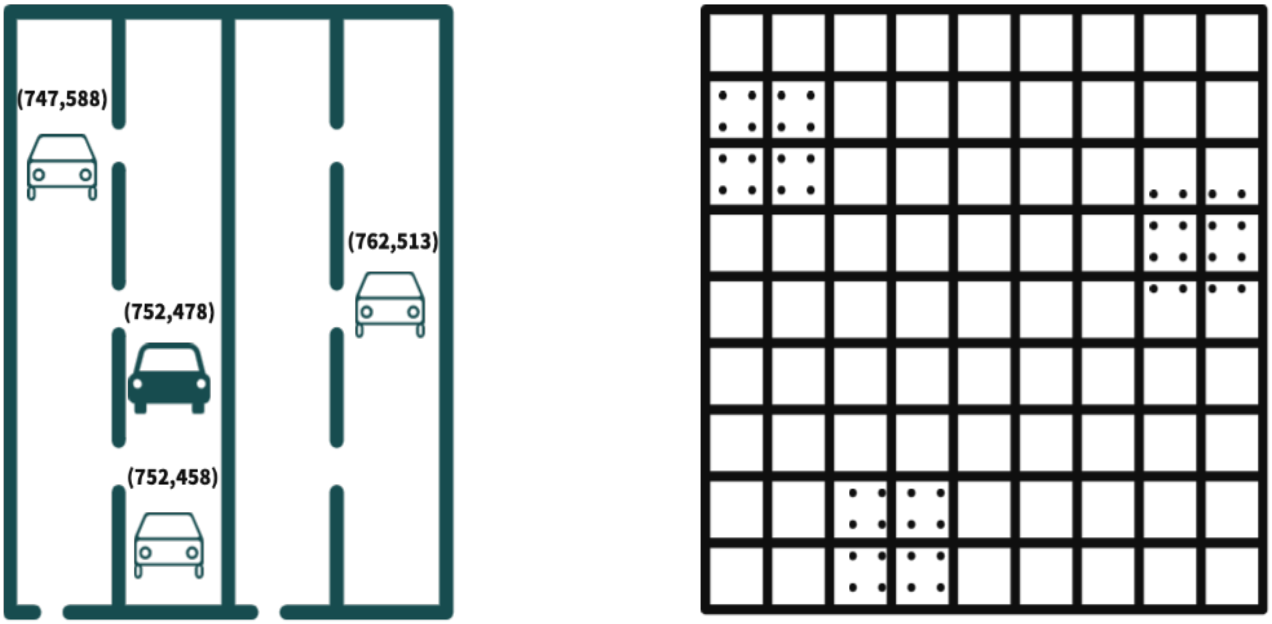

Figure 9: Encoding Scheme: Grid Based Representation

5.2 Using Horizontal Lane Number and Vertical Coordinates

Since we are talking of a laned road scenario, the horizontal position of a vehicle can be shown using a lane number (0,1,2,3) and the vertical position within the lane can be shown using the cartesian y coordinate relative to the current vehicle. So each neighboring vehicle is represented as (lane number, relative vertical location) as shown in Fig. 10. Example: (0, 110), (1,-20).

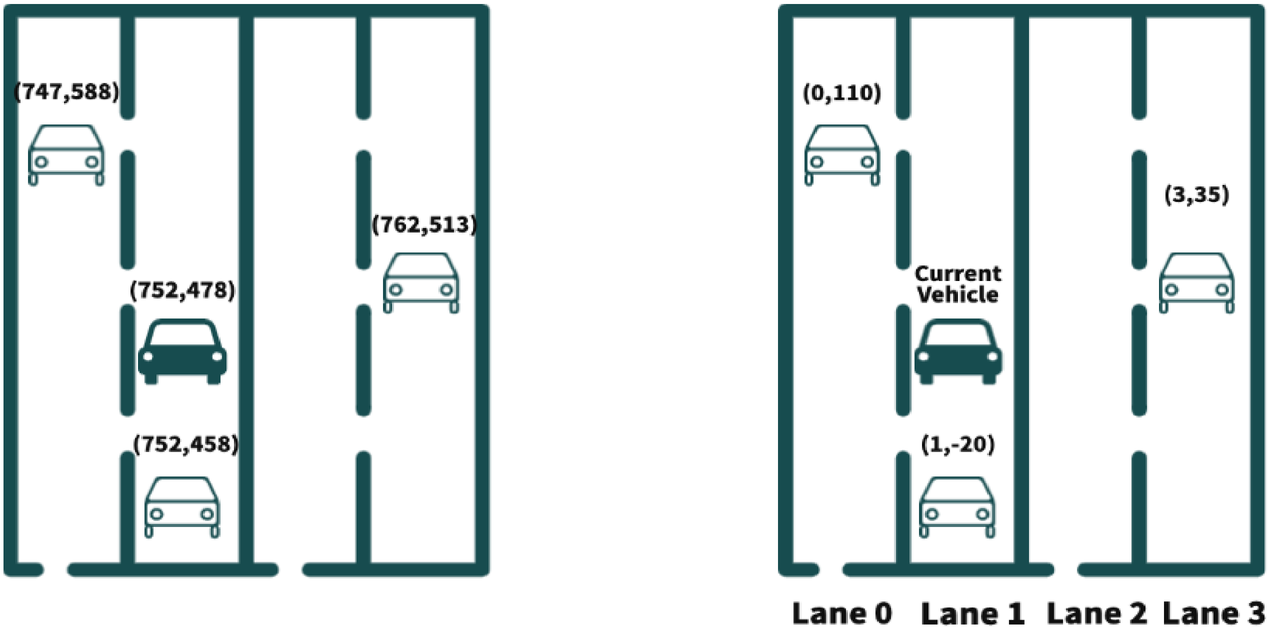

Figure 10: Embedding Scheme: Horizontal Lane Number and Vertical Coordinates

5.3 Using Angle and Straight Line Distance

Here we assume that the origin is located at the center of the current vehicle. We can measure the angle of the neighboring vehicle with respect to the current vehicle as the origin. The next thing we need to know is how much distance is the neighboring vehicle located in that direction. We can use a straight line distance between the neighboring vehicle and the current. So each neighboring vehicle is represented as (angle, straight line distance) as shown in Fig. 11. Example: (87.4 degrees, 110.11), (270 degrees, 20).

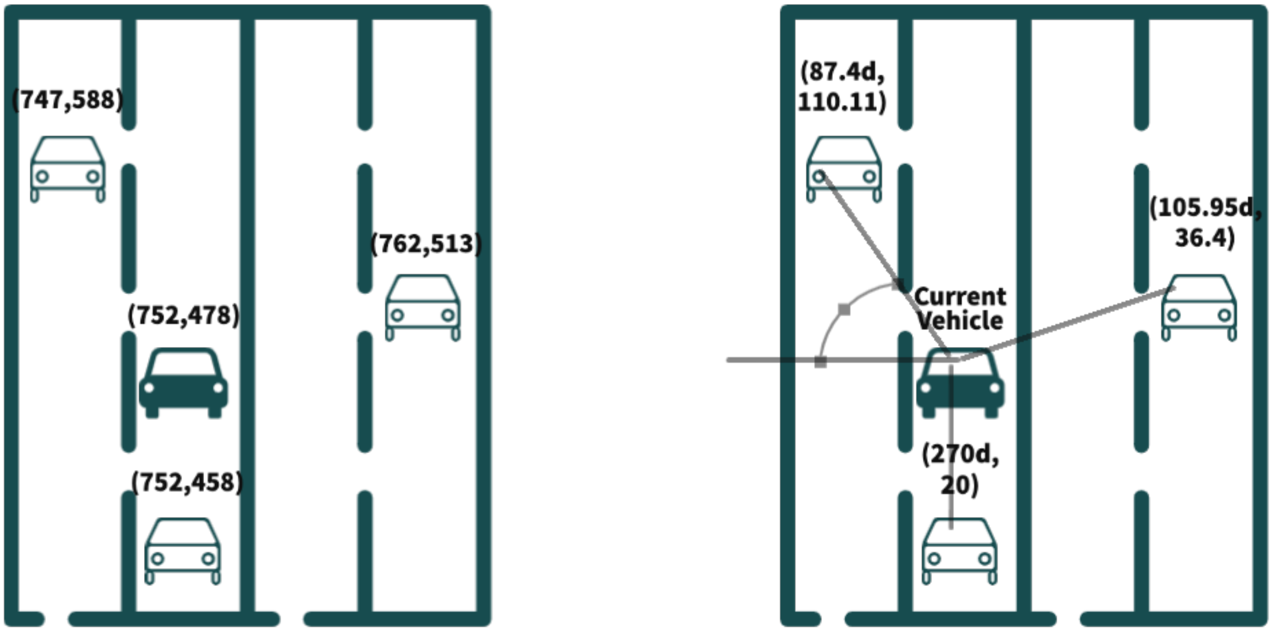

Figure 11: Encoding Scheme: Angle and Straight Line Distance

CHAPTER 6

Our Experimental Setup

6.1 Dataset Generation Process

Applying machine learning to communication system challenges becomes hard due to the non-availability of appropriate datasets which resemble real-life data and also due to the need for data from different sensors in our problem statement. We need a dataset that has GPS sensor data for location information, LIDAR data for information regarding the environment, and ray tracing data for getting network parameters to calculate channel gain. Such datasets which contain all this information in one place are called multimodal datasets. We chose Raymobtime [23] as our dataset. Raymobtime is a simulation technique that makes use of a traffic simulator with ray-tracing (RT) to provide realistic datasets for communication systems, particularly millimeter (mmWave) MIMO systems. The 3-dimensional scenario is integrated into Wireless InSite [24] after being exported from Cadmapper [25]. SUMO [26] is used to create the traffic simulation, which combines data from the Cadmapper 3D model and the locations of streets from OpenStreetMaps. Fig. 12 shows a diagrammatic representation of the data generation process. The Raymobtime dataset is a credible dataset that has been used previously in ITU AI/ML in 5G Challenge for various problem statements including the beam selection problem [2, 18].

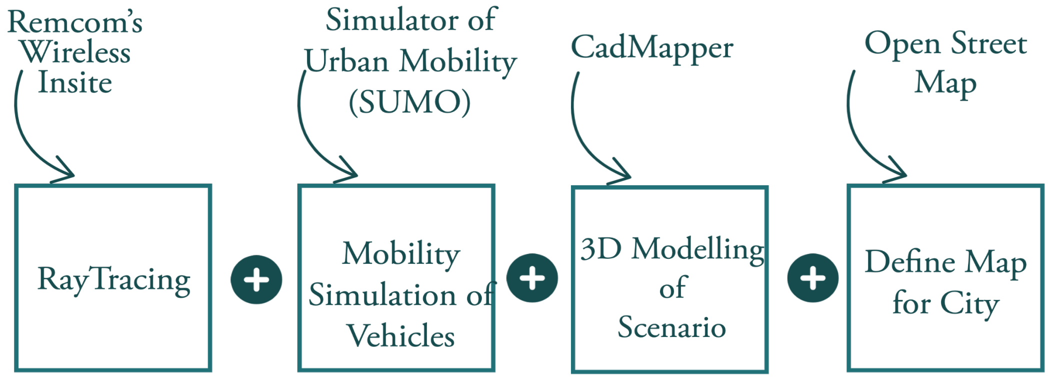

Figure 12: Raymobtime data generation process

The timeline for capturing data is divided into episodes and scenes as seen in Fig. 13. Each episode consists of multiple scenes. The periodic sampling method was used to record scene data with a time interval of 0.1 seconds. At the start of every episode, 10 random cars are selected and the receivers are mounted at the top center of the cars. Each receiver also known as User Equipment (UE) is assigned a unique number that remains constant throughout the episode. When the new episodes begin, this assignment process happens again and these numbers might change. The RSU also known as the base station has a fixed position and is located on the curbside of every street.

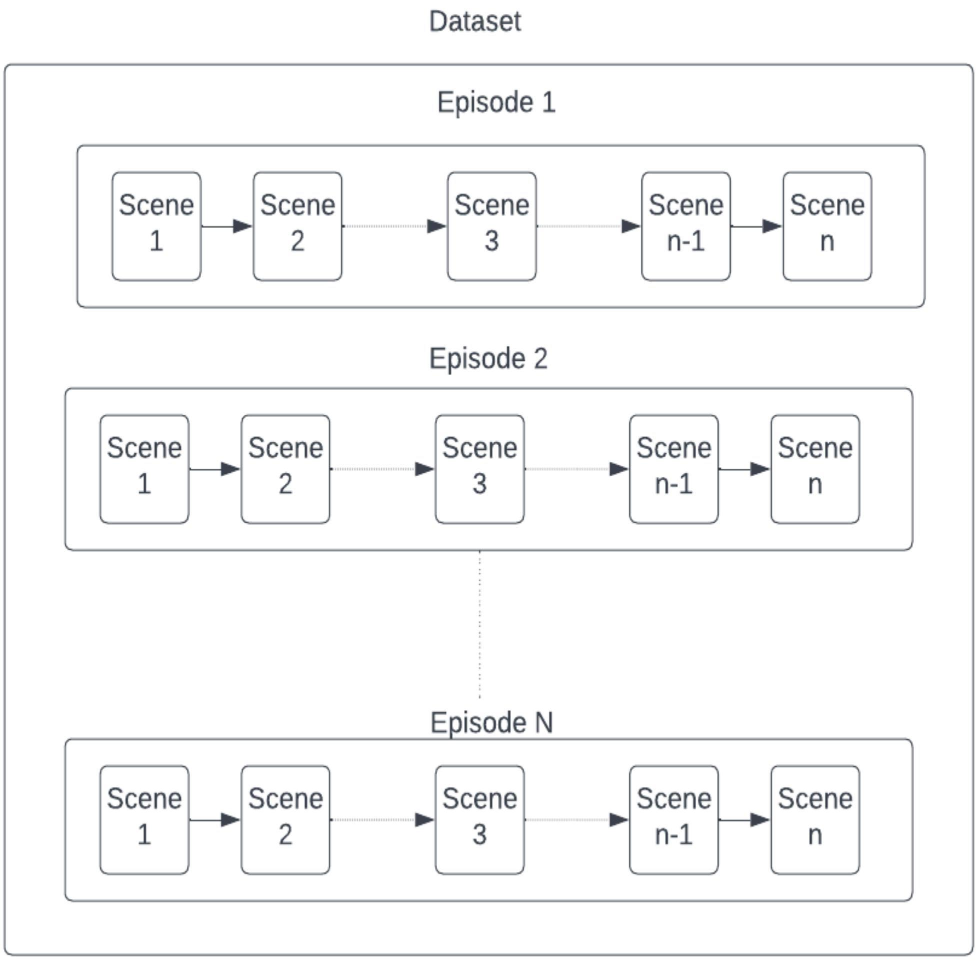

Figure 13: Scenes and Episodes in Data generation timeline

6.2 Dataset Files and Column Description

For our use case, there are three files of importance:

1. ray_tracing_data_s008_carrier60GHz.hdf5: This contains the channel information, it will be used in calculating the channel gain for each beam pair. The description of each column within this file can be seen in Table 2.

Index	Column Name	Unit	Description
1	Received power	dBm	Power level of the received signal
2	Time of arrival	seconds	Time when the signal arrives at receiver
3	Elevation angle of departure	degrees	Angle of signal departure from horizontal
4	Azimuth angle of departure	degrees	Angle of signal departure from reference direction
5	Elevation angle of arrival	degrees	Angle of signal arrival from horizontal
6	Azimuth angle of arrival	degrees	Angle of signal arrival from reference direction
7	LOS	Binary '1' or '0'	Flag '1' for a line of sight, '0' for non-line of sight: Indicates signal transmission path type
8	Ray phase	degrees	The phase of the signal at the receiver

Table 2: RayTracing file column description

2. lidar_data_s008.npz: This contains the LIDAR 3D point cloud data, which will

be used to capture the environment. The description of each column within this file can be seen in Table 3.

Index	**Column Name**	**Unit**	**Description**
1	PCD(X, Y, Z)	Float	These coordinates represent the location of the point in a 3D Cartesian coordinate system.

Table 3: LIDAR file column description

3. CoordVehiclesRxPerScene_s008.csv: This contains the GPS coordinates of the current vehicle and all the surrounding vehicles on the scene. The description of each column within this file can be seen in Table 4.

Index	Column Name	Unit	Description
1	Val	Char	Indicates if a channel is Valid[V] or Invalid[I].
2	EpisodeID	Int	Unique identifier for an episode in the network scenario.
3	SceneID	Int	Unique identifier for a scene within an episode.
4	VehicleArrayID	Int	Number of receivers in a vehicle.
5	VehicleName	String	Name of the vehicle in the network scenario.
6	X	Float	X coordinate of a point in space.
7	Y	Float	Y coordinate of a point in space.
8	Z	Float	Z coordinate of a point in space.
9	LOS	Binary '1' or '0'	Indicates if a signal has a line of sight path or not.

Table 4: GPS file column description

We will be using the s008 dataset of Raymobtime for training and the s009 dataset for testing. Both of them are generated for the Rosslyn, Virginia location. The position for the base station is fixed as (746.0, 560.0, 4.0). The number of valid records in s008 and s009 is 11194 and 9638 respectively. For generating the noisy dataset we have added 25% i.e. 2400 extra records with noisy LIDAR values to the test dataset, making the test data set the size 12038 records.

6.3 LIDAR Data Preprocessing

LIDAR which stands for Light Detection and Ranging uses the laser light reflected from the surroundings to measure the distance of obstacles in the surroundings. LIDAR data is represented by a 3D point cloud $P = (Xp, Yp, Zp)$. To reduce feature dimensionality, the data is mapped to a 2-dimensional grid of size $[20, 200, 1]$ as seen in Fig. 14. Here each position is assigned the value of 1 if there is an obstacle in the corresponding location, or 0 if it is empty. The RSU and vehicle locations are marked as -1 and -2 respectively. The 2-dimensional grid size is decided based on the range of Xp and Yp values, this process is called quantization.

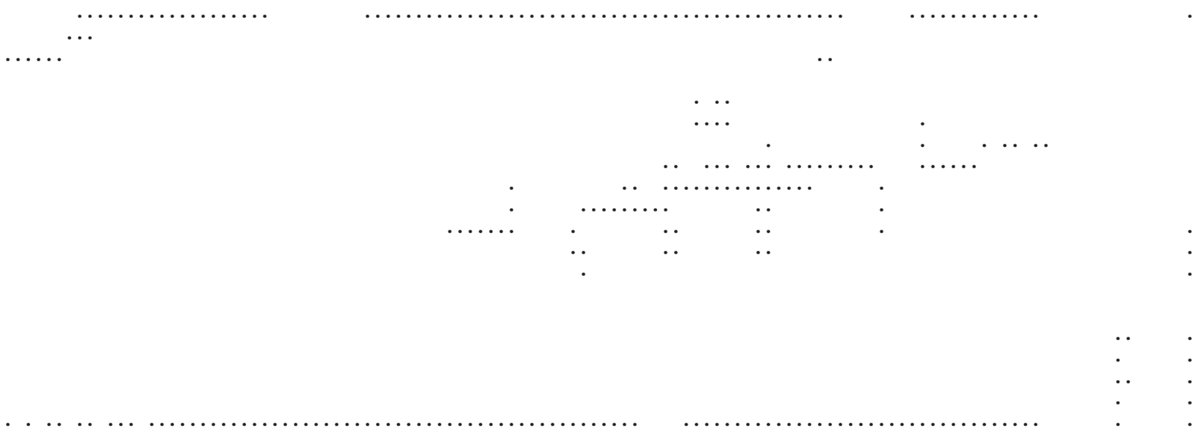

Figure 14: 2-dimensional grid representation of LIDAR data

This is then converted to an RGB image-based representation, with obstacles shown in black $(0, 0, 0)$, blank cells in white $(255, 255, 255)$, the RSU in red $(255, 0, 0)$, and the vehicle in green $(0, 255, 0)$. This makes data appropriate to give as input to ViT, which requires image data. The corresponding RGB representation for Fig. 14 can be seen in Fig. 15.

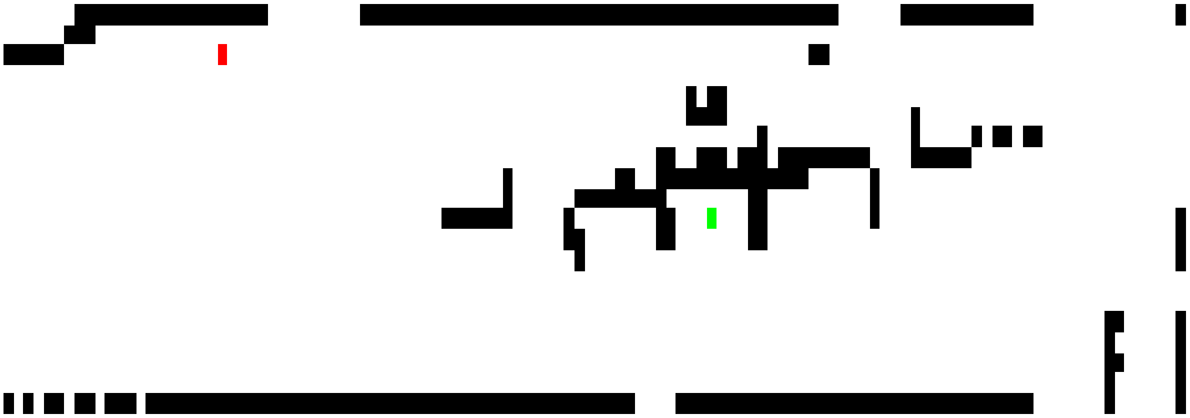

Figure 15: RGB image representation of LIDAR data

Before giving it as input to ViT this image data is converted into patches of smaller dimensions to reduce the number of features. The corresponding patch representation for the RGB image in Fig. 15 can be seen in Fig. 16.

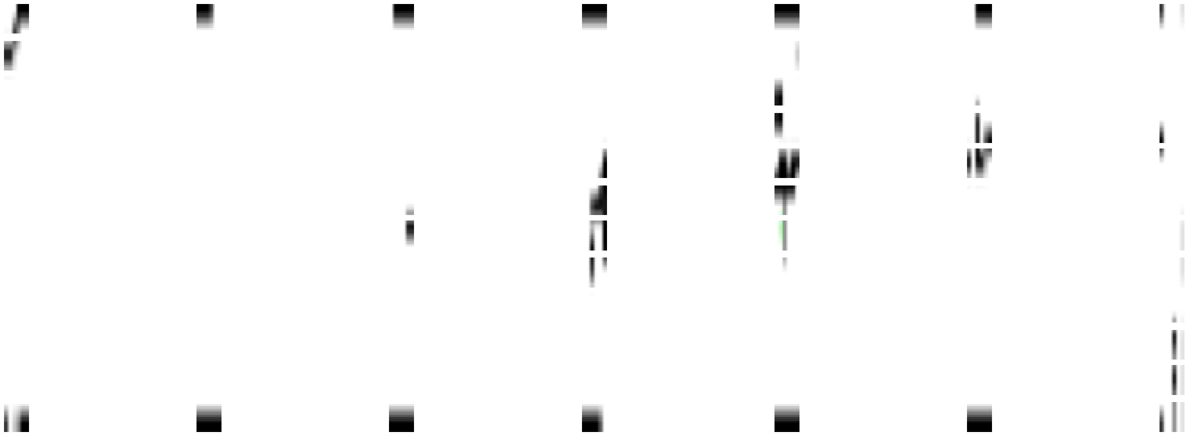

Figure 16: Breaking image into patches before input to ViT

6.4 Evaluation Criteria

One of the metrics to evaluate the model is top-k accuracy [18]. Top-k accuracy is defined as the expectation that the best beam pair (i^*, j^*) lies in the top-k beam pairs set $\mathcal{S}_k$ returned by our model. Mathematically it can be written as

$$A(k) = \mathbb{E}\left[\{(i^*, j^*) \in \mathcal{S}_k\}\right]$$

Another useful metric is top-k throughput [2], which is mathematically defined as

$$R = \left(\sum_{t=1}^{T} \log_2 \left(1 + y_{\tilde{i}\tilde{j}} \right) \right) / \left(\sum_{t=1}^{T} \log_2 \left(1 + y_{i^* j^*} \right) \right)$$

where T is the number of records in test data, $(\tilde{i}, \tilde{j})$ is the best beam pair within top k predicted beam pairs, and (i^*, j^*) is the best actual beam pair. It's hard to pinpoint single beam pairs i.e. top 1 accuracy, but it is more feasible to predict top 5 and top 10 beam pairs. Searching among 10 beam pairs is still much faster than searching 256 beam pairs in our case, therefore we will calculate results for top 1, top 5, and top 10 accuracies and throughput while evaluating our model and comparing it with the previous works.

CHAPTER 7

Simulation Experiments and Results

We have used the benchmark dataset named Raymobtime [23] to train and test our model. Within Raymobtime we have used dataset s008 for training and s009 for testing. Both these sets are multimodal and consist of Raytracing, GPS, LIDAR, and camera image data. The training dataset s008 consists of 11194 data records, the original test dataset s009 consists of 9638 data records and the noisy test dataset generated using s009 contains 12038 records.

7.1 Original Dataset

This experiment is conducted on the original dataset where there is no noise in LIDAR data. Results are compared across our proposed model (ViT) and CNN. Our test results can be seen in Table 5, and Table 6. On the original dataset, ViT achieved an accuracy of 92.23% percent, 86.13%, and 59.87% for the top 10, top 5, and top 1 beam pairs respectively. We also got a throughput of 97.22%, 94.22%, and 79.29% for the top 10, top 5, and top 1 beam pairs respectively. Both the accuracies and throughputs are within the 1% range of previous CNN approaches [2]

Table 5: Top k Accuracy comparison for proposed ViT vs CNN [2] on Original dataset

Model Name	**Top K Accuracies**		
	T1	**T5**	**T10**
Proposed Model (ViT)	59.87%	86.13%	92.23%
CNN [2]	59.16%	87.01%	92.13%

Table 6: Top k Throughput comparison for proposed ViT vs CNN [2] on Original dataset

Model Name	**Top K Throughputs**		
	T1	**T5**	**T10**
Proposed Model (ViT)	79.29%	94.23%	97.22%
CNN [2]	77.98%	94.32%	97.45%

The graphs in Fig. 17 display the accuracy vs Top k beam pairs curve. Our model performs similarly to [2] for T1-T2, while [2] performs slightly better for T3-T6, and they both give the same performance for T6-T10.

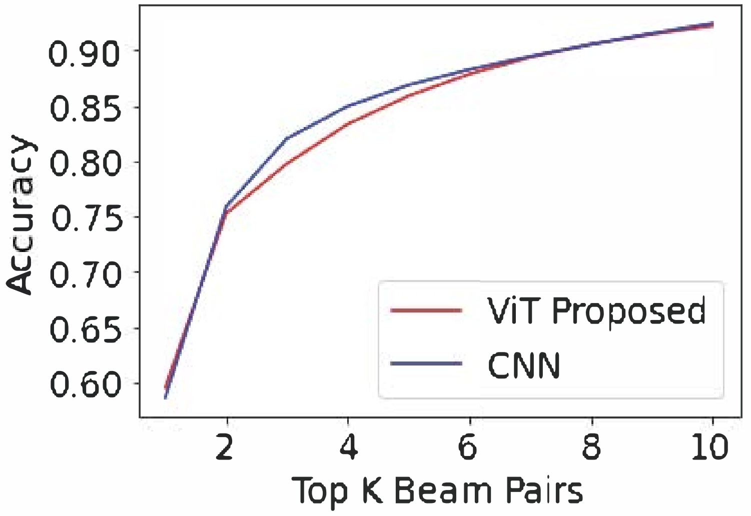

Figure 17: Accuracy vs Top k beam pairs curve on Original dataset

The graphs in Fig. 18 display the throughput vs Top k beam pairs curve. Our model performs slightly better to [2] for T1-T2, while [2] performs slightly better for T3-T6, and they both give the same performance for T7-T10.

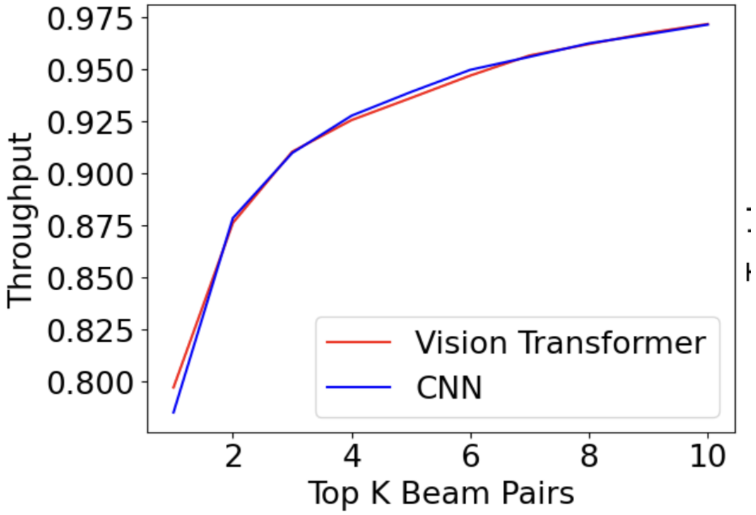

Figure 18: Throughput vs Top k beam pairs curve on Original dataset

The graphs in Fig. 19 display the Test loss vs Epoch curve for the original dataset. Our model initially had slower convergence compared to [2]. However, after 75 epochs, our model's convergence increased and it was able to achieve better loss reduction than [2] after 200 epochs.

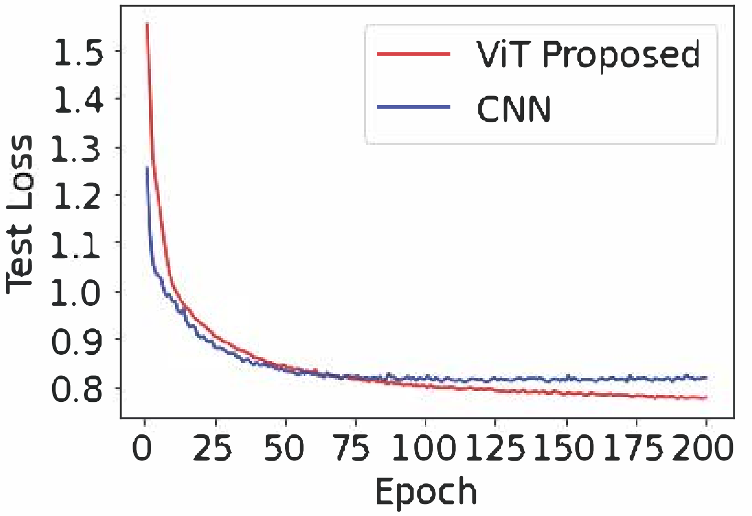

Figure 19: Test Loss vs Epoch curve on Original dataset

All these observations indicate that our proposed approach is as good as any state of art models proposed in previous works when tested on the original dataset.

7.2 Noisy Dataset

In this experiment, we first introduced noise in our data by adding 2400 records having random noise in LIDAR test data. These extra records are randomly extracted from test data s009, and the LIDAR part of these records is overwritten by randomly generated 0s and 1s after extraction. Finally these extra records are appended back to the test data s009. Then we used the GVEC techniques discussed above to improve the model performance and compared it with previous works.

7.2.1 Noise Reduction using Grid-Based Representation

To reduce the noise, we have replaced the noisy LIDAR records with the LIDAR-like records generated using GVEC Grid-Based Representation. This will bring back noise down to a level our models can tolerate. After this, we train ViT and CNN on this noise-reduced dataset and compare their performance. Our test results can be seen in Table 7 and Table 8. Working with a noisy dataset, our model beats previous work [2] by 17.31%, 7.07%, and 4.96% for T1, T5, and T10 accuracy respectively. Also in terms of throughput, it beats the previous work [2] by 15.38%, 5.33%, and 3.44% for T1, T5, and T10 throughput respectively. Our model was able to maintain nearly identical accuracy and throughput despite the presence of noise.

Table 7: Top k Accuracy comparison for proposed ViT vs CNN [2] on Noisy dataset - GVEC Grid-Based Representation

Model Name	**Top K Accuracies**		
	T1	**T5**	**T10**
Proposed Model (ViT)	59.33%	85.82%	92.19%
CNN [2]	42.02%	78.75%	87.23%

Table 8: Top k Throughput comparison for proposed ViT vs CNN [2] on Noisy dataset - GVEC Grid-Based Representation

Model Name	**Top K Throughputs**		
	T1	**T5**	**T10**
Proposed Model (ViT)	78.80%	93.38%	97.19%
CNN [2]	63.42%	88.65%	93.75%

The graph in Fig. 20 displays the accuracy vs Top k beam pairs curve. Our model curve is continuously above the curve for [2], beating it for the entire T1 to T10 accuracies.

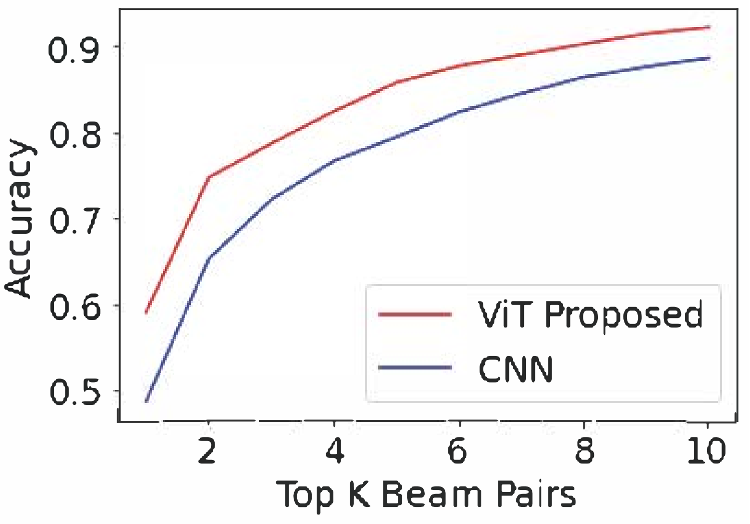

Figure 20: Accuracy vs Top k beam pairs curve on Noisy dataset - GVEC Grid-Based Representation

The graph in Fig. 21 displays the throughput vs Top k beam pairs curve. Our model curve is continuously above the curve for [2], beating it for the entire T1 to T10 throughputs.

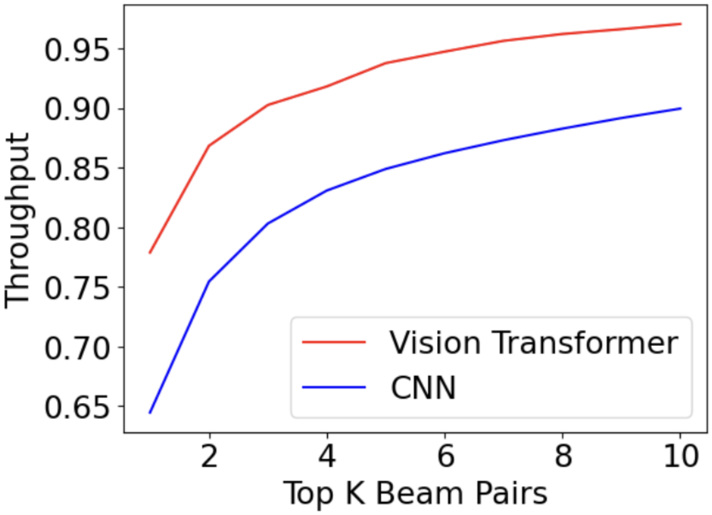

Figure 21: Throughput vs Top k beam pairs curve on Noisy dataset - GVEC Grid-Based Representation

The graphs in Fig. 22 display the Test loss vs Epoch curve. We can see that our proposed model curve continues to converge until 200 epochs and minimizes loss to the same point as on the original data. However, [2] diverges after 30 epochs, causing an increase in loss.

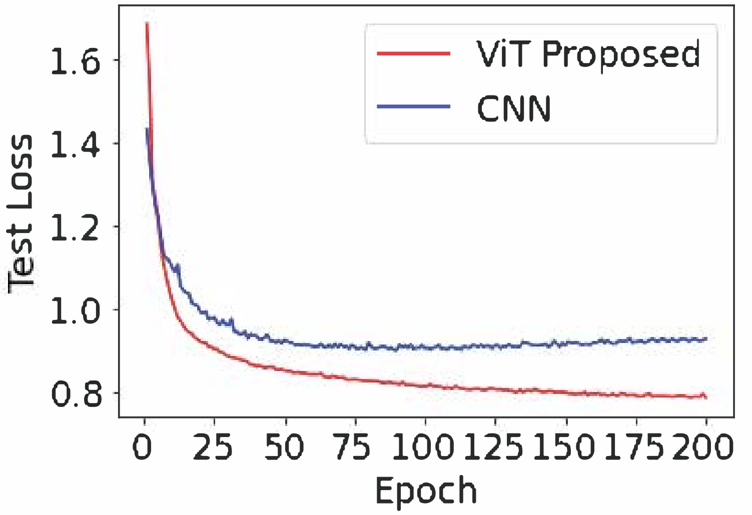

Figure 22: Test Loss vs Epoch curve on Noisy dataset - GVEC Grid-Based Representation

7.2.2 Ensemble Model using Horizontal Lane Numbers and Vertical Coordinates

In this experiment, we have used an ensemble model, in which model 1 is trained using the GPS+LIDAR data, and the other model 2 is trained using GPS + GVEC (Lane Number, Relative Vertical Y Coordinate) data. For the non-noisy part of the data, predictions from Model 1 are used, and for the noisy part predictions from Model 2 are used. The experiment is performed using both ViT and CNN as model 1 and the results are compared.

Table 9 shows the results for accuracy for T1, T5, and T10. Working with a noisy dataset, our model beats previous work [2] by 3.63%, 0.35%, and 0.41% for T1, T5, and T10 accuracy respectively. Table 10 shows the results for throughput for T1, T5, and T10. In terms of throughput, it beats the previous work [2] by 2.76%, 0.02%, and 0.33% for T1, T5, and T10 throughput respectively. Our model showed a slight improvement in terms of both accuracy and throughput compared to previous work [2].

Table 9: Top k Accuracy comparison for proposed ViT vs CNN [2] on Noisy dataset - GVEC Horizontal Lane Numbers and Vertical Coordinates

Model Name	**Top K Accuracies**		
	T1	**T5**	**T10**
Proposed Model (ViT)	57.05%	83.82%	91.11%
CNN [2]	53.42%	83.47%	90.70%

Table 10: Top k Throughput comparison for proposed ViT vs CNN [2] on Noisy dataset - GVEC Horizontal Lane Numbers and Vertical Coordinates

Model Name	**Top K Throughputs**		
	T1	**T5**	**T10**
Proposed Model (ViT)	77.19%	93.04%	96.75%
CNN [2]	74.43%	93.02%	96.42%

The graphs in Fig. 23 display the accuracy vs Top k beam pairs curve for this experiment. Our model curve is above the curve for [2] for T1-T3, it is coincident with the curve for [2] for T4-T6, and beats [2] again between T7-T10.

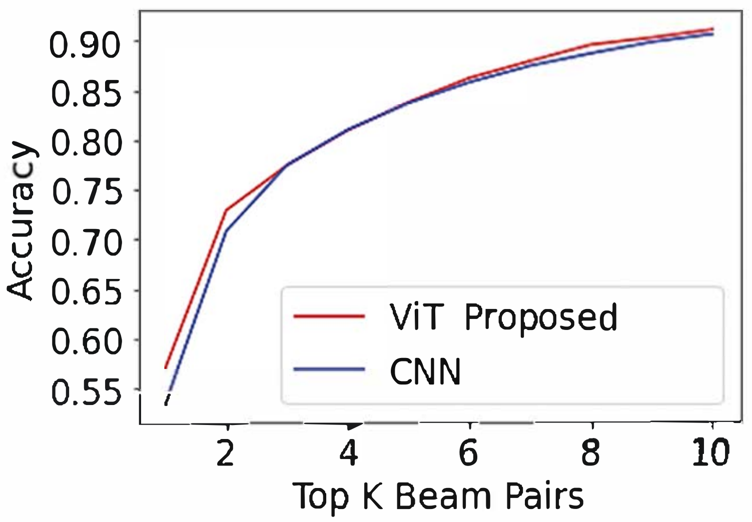

Figure 23: Accuracy vs Top k beam pairs curve on Noisy dataset - GVEC Horizontal Lane Numbers and Vertical Coordinates

The graphs in Fig. 24 display the Throughput vs Top k beam pairs curve for this experiment. The Throughput curve looks much like a replica of the accuracy curve. Our model curve is above the curve for [2] for T1-T3, it is coincident with the curve for [2] for T4-T6, and beats [2] again between T7-T10.

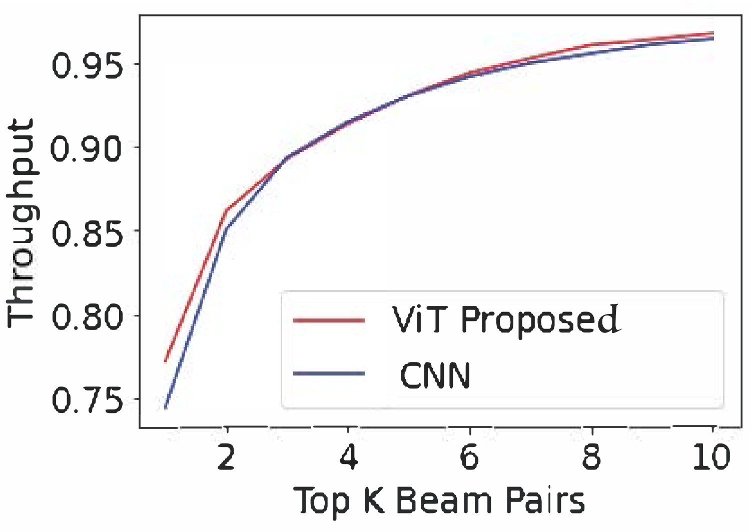

Figure 24: Throughput vs Top k beam pairs curve on Noisy dataset - GVEC Horizontal Lane Numbers and Vertical Coordinates

7.2.3 Ensemble Model using Angle and Straight Line Distance

In this experiment, we have used an ensemble model, in which model 1 is trained using the GPS+LIDAR data, and model 2 is trained using GPS + GVEC (Angle Straight Line Distance) data. For the non-noisy records prediction from model 1 are used, and for the noisy part of the data predictions from model 2 are used. The experiment is performed using both ViT and CNN as model 1 and results are compared.

Table 11 shows the results for accuracy for T1, T5, and T10. Working with a noisy dataset, our model beats previous work [2] by 4.56%, 1.35%, and 0.87% for T1, T5, and T10 accuracy respectively. Table 12 shows the results for throughput for T1, T5, and T10. In terms of throughput, it beats the previous work [2] by 3.19%, 0.75%, and 0.53% for T1, T5, and T10 throughput respectively. Our model showed a slight improvement in terms of both accuracy and throughput compared to the previous work [2].

Table 11: Top k Accuracy comparison for proposed ViT vs CNN [2] on Noisy dataset - GVEC Angle and Straight Line Distance

Model Name	**Top K Accuracies**		
	T1	**T5**	**T10**
Proposed Model (ViT)	58.41%	84.07%	91.58%
CNN [2]	53.85%	82.72%	90.71%

Table 12: Top k Throughput comparison for proposed ViT vs CNN [2] on Noisy dataset - GVEC Angle and Straight Line Distance

Model Name	**Top K Throughputs**		
	T1	**T5**	**T10**
Proposed Model (ViT)	78.01%	93.11%	96.77%
CNN [2]	74.82%	92.36%	96.24%

The graphs in Fig. 25 display the Accuracy vs Top k beam pairs curve for this experiment. Our model curve is continuously above the curve for [2] for the entire T1-T10, showing a slight improvement in accuracies.

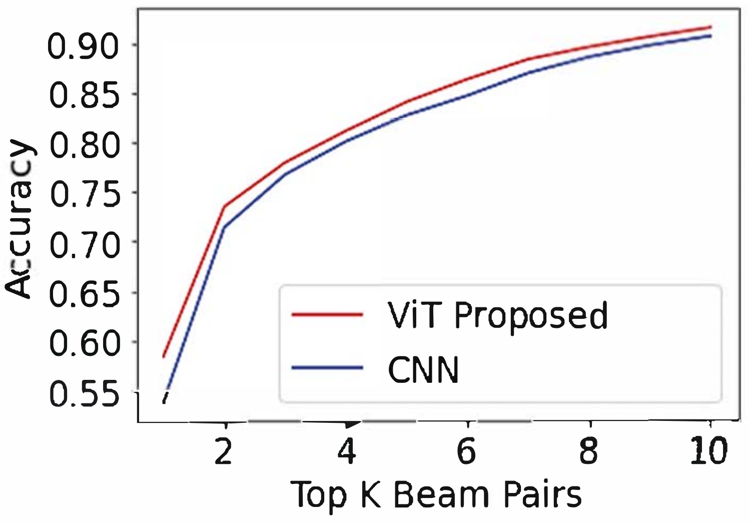

Figure 25: Accuracy vs Top k beam pairs curve on Noisy dataset - GVEC Angle and Straight Line Distance

The graphs in Fig. 26 display the Throughput vs Top k beam pairs curve for this experiment. Our model curve is continuously above the curve for [2] for the entire T1-T10, showing a slight improvement in throughput.

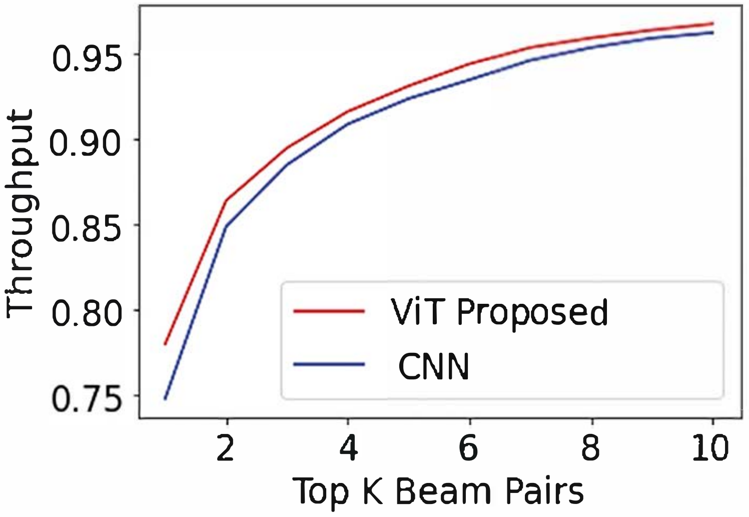

Figure 26: Throughput vs Top k beam pairs curve on Noisy dataset - GVEC Angle and Straight Line Distance

7.3 Resource Consumption

The resource consumption by the proposed ViT model and previous CNN model [2] can be seen in Table 13. We can see our proposed model uses 0.7 GB more system random access memory (RAM), while the previous CNN model [2] uses 0.2 GB more graphics processing unit (GPU) RAM. Overall there is not a significant difference in resource consumption between both models. In terms of training time, ViT model being more complex due to presence of multi-headed self-attention takes more time to train than CNN model [2]. But since most training is done offline here, models can be trained in advance and used. For online training scheme, training time for ViT should be reduced further.

Table 13: Resource Consumption by ViT vs CNN [2]

Model Name	**System RAM**	**GPU RAM**	**Disk**	**Offline Training Time**
Proposed Model (ViT)	10.6 GB	4.5 GB	31.6 GB	23 Min
CNN [2]	10.8 GB	3.8 GB	31.6 GB	6 Min

7.4 Comparison of different GVEC Encoding Schemes

After extensive testing of various GVEC-based encoding schemes and solutions, our research has shown that using a Grid-Based representation of GVEC for noise reduction performs better than other encoding schemes. This method replaces noisy LIDAR data with virtually generated records, reducing the level of noise. We compared ViT and CNN models on noisy data and found ViT to outperform CNN. This is because ViT can handle this reduced noise level and perform well due to its multi-headed self-attention mechanism, while CNN cannot. ViT maintains similar accuracy and throughput even with noisy data as it would have done on original no-noise data. Thus, we can clearly observe that our proposed approach is more reliable and accurate than previous methods, especially in noisy environments, making it a promising method for mmWave beam selection.

CHAPTER 8

Conclusion

This research paper presents a novel approach in machine learning that combines LIDAR and GPS data for beam selection, utilizing the vision transformer (ViT) model instead of conventional convolutional neural networks (CNNs). The primary focus is on addressing real-time challenges and introducing a solution to correct noisy LIDAR records using GPS-based Virtual Environment Capture (GVEC). The proposed approach not only demonstrates comparable performance to the state-of-the-art methods on the original dataset but also it outperforms these methods on the noisy dataset, achieving a favorable balance between accuracy, throughput, and resource utilization. These promising results make the approach highly suitable for practical applications.

Nevertheless, there are several open issues that require further research. Firstly, extensive testing on a large-scale dataset is necessary to fully explore the true potential of the ViT model. Currently, the solution only caters to user equipment (UE) equipped with LIDAR sensors. However, there are various UE types, such as cellular phones and older vehicles, which lack LIDAR sensors. Therefore, alternative solutions must be developed for these devices. It is also essential to investigate whether GVEC alone can be used as a complete replacement for LIDAR sensor. Furthermore, deploying such a system in a real environment requires establishing agreements between network providers and vehicle manufacturers to facilitate data sharing. Addressing this issue is crucial for the practical implementation of the proposed approach.

LIST OF REFERENCES

[1] A. I. Hedu, ‘‘Visual guide to transformer neural networks - (episode 2) Multi-Head & Self-Attention,’’ Dec. 2020. [Online]. Available: https://www.youtube.com/watch?v=mMa2PmYJlCo

[2] M. Zecchin, M. B. Mashhadi, M. Jankowski, D. Gündüz, M. Kountouris, and D. Gesbert, ‘‘Lidar and position-aided mmwave beam selection with non-local cnns and curriculum training,’’ *IEEE Transactions on Vehicular Technology*, vol. 71, no. 3, pp. 2979--2990, 2022.

[3] W. Saad, M. Bennis, and M. Chen, ‘‘A vision of 6G wireless systems: Applications, trends, technologies, and open research problems,’’ *IEEE Netw.*, vol. 34, no. 3, pp. 134--142, May 2020.

[4] M. Agiwal, A. Roy, and N. Saxena, ‘‘Next generation 5G wireless networks: A comprehensive survey,’’ *IEEE Communications Surveys & Tutorials*, vol. 18, no. 3, pp. 1617--1655, 2016.

[5] C. Liu, M. Li, S. V. Hanly, P. Whiting, and I. B. Collings, ‘‘Millimeter-Wave small cells: Base station discovery, beam alignment, and system design challenges,’’ *IEEE Wirel. Commun.*, vol. 25, no. 4, pp. 40--46, Aug. 2018.

[6] R. W. Heath, N. González-Prelcic, S. Rangan, W. Roh, and A. M. Sayeed, ‘‘An overview of signal processing techniques for millimeter wave mimo systems,’’ *IEEE Journal of Selected Topics in Signal Processing*, vol. 10, no. 3, pp. 436--453, 2016.

[7] M. S. Sim, Y.-G. Lim, S. H. Park, L. Dai, and C.-B. Chae, ‘‘Deep Learning-Based mmwave beam selection for 5G NR/6G with sub-6 GHz channel information: Algorithms and prototype validation,’’ *IEEE Access*, vol. 8, pp. 51 634--51 646, 2020.

[8] Y. Liu, Y. Tian, B. Sun, Y. Wang, and F.-Y. Wang, ‘‘Parallel LiDARs meet the foggy weather,’’ *IEEE Journal of Radio Frequency Identification*, vol. 6, pp. 867--870, 2022.

[9] N. Charron, S. Phillips, and S. L. Waslander, ‘‘De-noising of lidar point clouds corrupted by snowfall,’’ in *2018 15th Conference on Computer and Robot Vision (CRV)*, May 2018, pp. 254--261.

[10] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby,

“An image is worth 16x16 words: Transformers for image recognition at scale,” in *International Conference on Learning Representations*, 2021. [Online]. Available: https://openreview.net/forum?id=YicbFdNTTy

[11] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention is all you need,” in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, ser. NIPS’17. Red Hook, NY, USA: Curran Associates Inc., 2017, p. 6000–6010.

[12] A. Ali, N. González-Prelcic, and R. W. Heath, “Millimeter wave Beam-Selection using Out-of-Band spatial information,” *IEEE Trans. Wireless Commun.*, vol. 17, no. 2, pp. 1038--1052, Feb. 2018.

[13] H. Chen, C. Sun, F. Jiang, and J. Jiang, “Beams selection for MmWave Multi-Connection based on Sub-6GHz predicting and parallel transfer learning,” in *2021 IEEE/CIC International Conference on Communications in China (ICCC)*, July 2021, pp. 469--474.

[14] J. Morais, A. Behboodi, H. Pezeshki, and A. Alkhateeb, “Position aided beam prediction in the real world: How useful gps locations actually are?” 2022.

[15] S. Rezaie, C. N. Manchón, and E. de Carvalho, “Location- and orientation-aided millimeter wave beam selection using deep learning,” in *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, 2020, pp. 1--6.

[16] S. Jiang, G. Charan, and A. Alkhateeb, “Lidar aided future beam prediction in real-world millimeter wave v2i communications,” *IEEE Wireless Communications Letters*, vol. 12, no. 2. [Online]. Available: https://par.nsf.gov/biblio/10404312

[17] A. Klautau, N. González-Prelcic, and R. W. Heath, “LIDAR data for deep Learning-Based mmwave Beam-Selection,” *IEEE Wireless Communications Letters*, vol. 8, no. 3, pp. 909--912, June 2019.

[18] M. B. Mashhadi, M. Jankowski, T.-Y. Tung, S. Kobus, and D. Gündüz, “Federated mmwave beam selection utilizing LIDAR data,” *IEEE Wireless Communications Letters*, vol. 10, no. 10, pp. 2269--2273, Oct. 2021.

[19] M. Alrabeiah, A. Hredzak, and A. Alkhateeb, “Millimeter wave base stations with cameras: Vision-aided beam and blockage prediction,” in *2020 IEEE 91st Vehicular Technology Conference, VTC Spring 2020 - Proceedings*, ser. IEEE Vehicular Technology Conference. Institute of Electrical and Electronics Engineers Inc., May 2020.

[20] G. Charan, T. Osman, A. Hredzak, N. Thawdar, and A. Alkhateeb, “Vision-position multi-modal beam prediction using real millimeter wave

datasets,'' in *2022 IEEE Wireless Communications and Networking Conference (WCNC)*. IEEE Press, 2022, p. 2727–2731. [Online]. Available: https://doi.org/10.1109/WCNC51071.2022.9771835

[21] W. Xu, F. Gao, S. Jin, and A. Alkhateeb, ``3D Scene-Based beam selection for mmwave communications,'' *IEEE Wireless Communications Letters*, vol. 9, no. 11, pp. 1850--1854, Nov. 2020.

[22] Z. Ying, H. Yang, J. Gao, and K. Zheng, ``A new Vision-Aided beam prediction scheme for mmwave wireless communications,'' in *2020 IEEE 6th International Conference on Computer and Communications (ICCC)*, Dec. 2020, pp. 232--237.

[23] A. Klautau, P. Batista, N. González-Prelcic, Y. Wang, and R. W. Heath, ``5G MIMO data for machine learning: Application to Beam-Selection using deep learning,'' in *2018 Information Theory and Applications Workshop (ITA)*, Feb. 2018, pp. 1--9.

[24] ``Wireless InSite® propagation software applications ---,'' https://www.remcom.com/wireless-insite-em-propagation-applications, accessed: 2023-3-30.

[25] ``CADMAPPER - worldwide map files for any design program,'' https://cadmapper.com/, accessed: 2023-3-30.

[26] ``SUMO,'' https://mininet-wifi.github.io/sumo/, accessed: 2023-3-30.