ITUEvents

ITU-ML5G-PS-012: ML5G-PHY: Beam Selection (Universidade Federal do Pará, Brazil)

26 June 2020

ITU

AI/ML in 5G Challenge

Applying machine learning in communication networks

ai5gchallenge@itu.int

Bronze sponsor:

Organized by:

ITU Artificial Intelligence/Machine Learning in 5G Challenge

An Overview of the ITU-ML5G-PS-012 "ML5G-PHY [beam selection]"

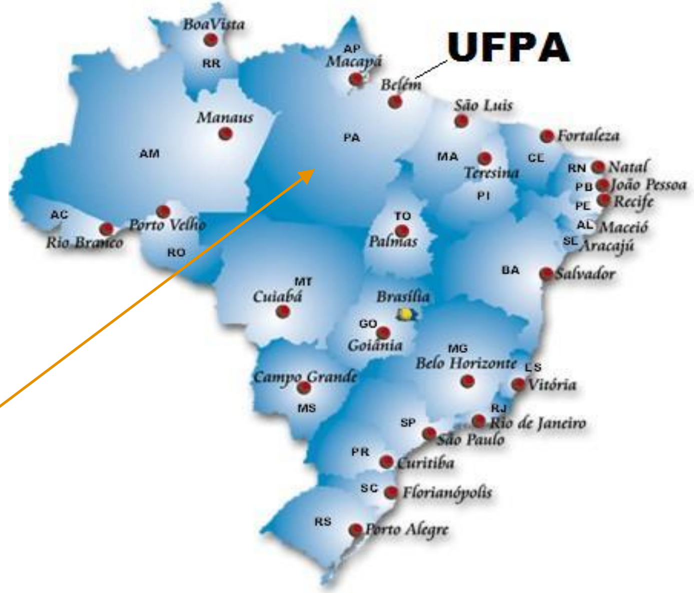
Aldebaro Klautau
Federal University of Pará (UFPA) / LASSE
http://ai5gchallenge.ufpa.br

Joint work with Profs. Diego Gomes (UNIFESSPA), Francisco Müller (UFPA), Nuria González-Prelcic (NCSU), Robert Heath (UT) and several students

June 26, 2020

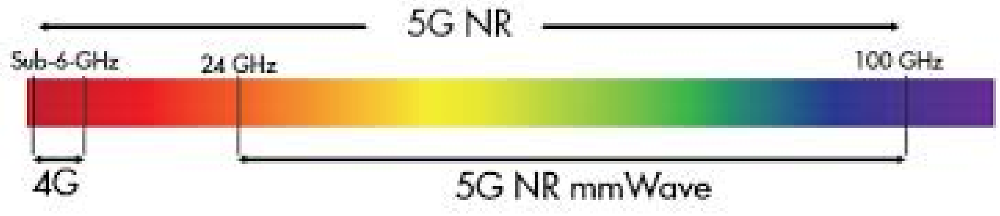
UFPA
Federal University of Pará

- Established in 1957
- Largest academic and research institution in Amazon (Pará state in North Brazil)
- One of the largest Brazilian universities with total population (students + staff) of ~61k people
- One of the missions is the sustainable development of the region through science and technology



Importance of beam selection in 5G MIMO

Using large bandwidths at millimeter waves (mmWaves) is key to achieve high bit rates



AWGN channel capacity:
$C = BW \log_2 (1 + SNR)$

Signal-to-noise ratio

Bandwidth (Hz)

Disadvantage: stronger attenuation in mmWaves than in sub-6 GHz

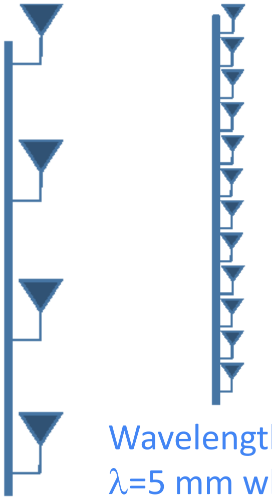
Remedy: use multiple antennas in MIMO systems → enable generation of programmable directional beams that increase reach and minimize interference

Steering mmwave beams with antenna arrays

Array form factor decreases when frequency incresses

Sub-6 GHz

mmWave

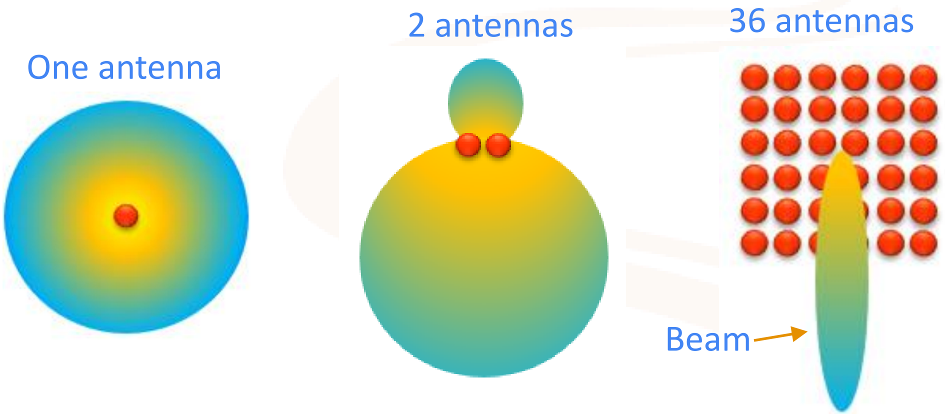


Wavelength λ=c/f

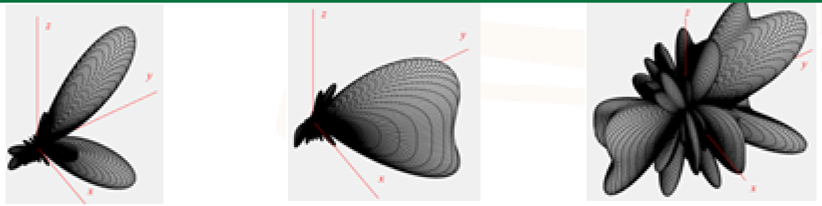
λ=5 mm when f=60 GHz

Space between antenna elements = $\lambda/2$

Radiation patterns:



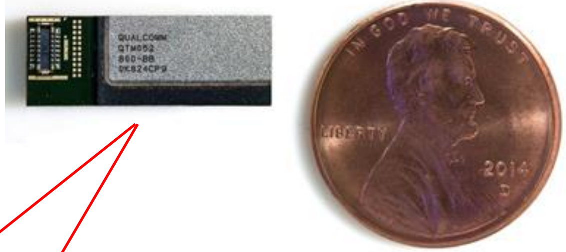
Vector of angles imposes the radiation pattern of phased antenna arrays and (pointy) beam steering



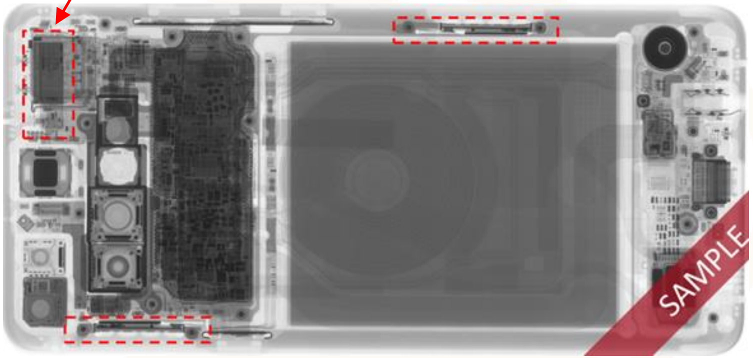
Example of antenna array for 5G phones

Features:
- mmWave frequency bands (28 GHz, etc.)
- Supports beam forming and tracking
- Small form fator
- Estimated 64 antenna elements

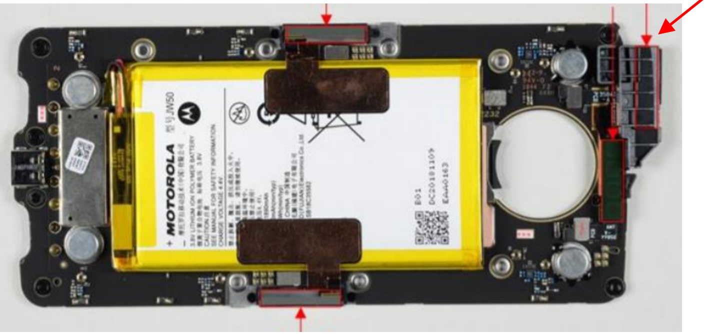
Qualcomm QTM052 antenna module



Samsung Galaxy S10 5G (3 modules)



Motorola Moto Mod 5G (4 modules)

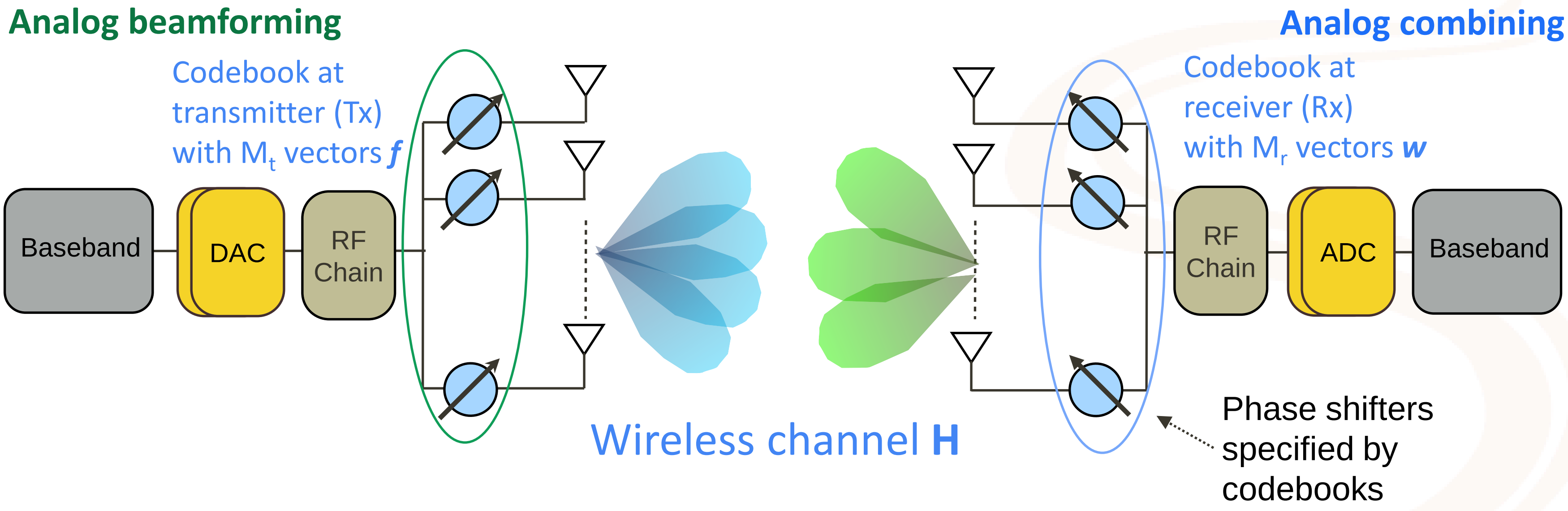


Hard to deal with blocking → several antenna modules
Vehicle-to-infrastructure (V2I) scenario → +predictability

[1] SystemPlus Consulting – Motorola Mod 5G teardown
[2] Tech Insights - https://www.techinsights.com/blog/qualcomm-qtm052-mmwave-antenna-module
[3] https://www.microwavejournal.com/articles/31448-first-5g-mmwave-antenna-module-for-smartphones

Beam training (selection) in 5G mmWave

After initial ***selection***, beam can be ***tracked*** with less overhead



Brute force to find best: try all possible M_t x M_r pairs of indices

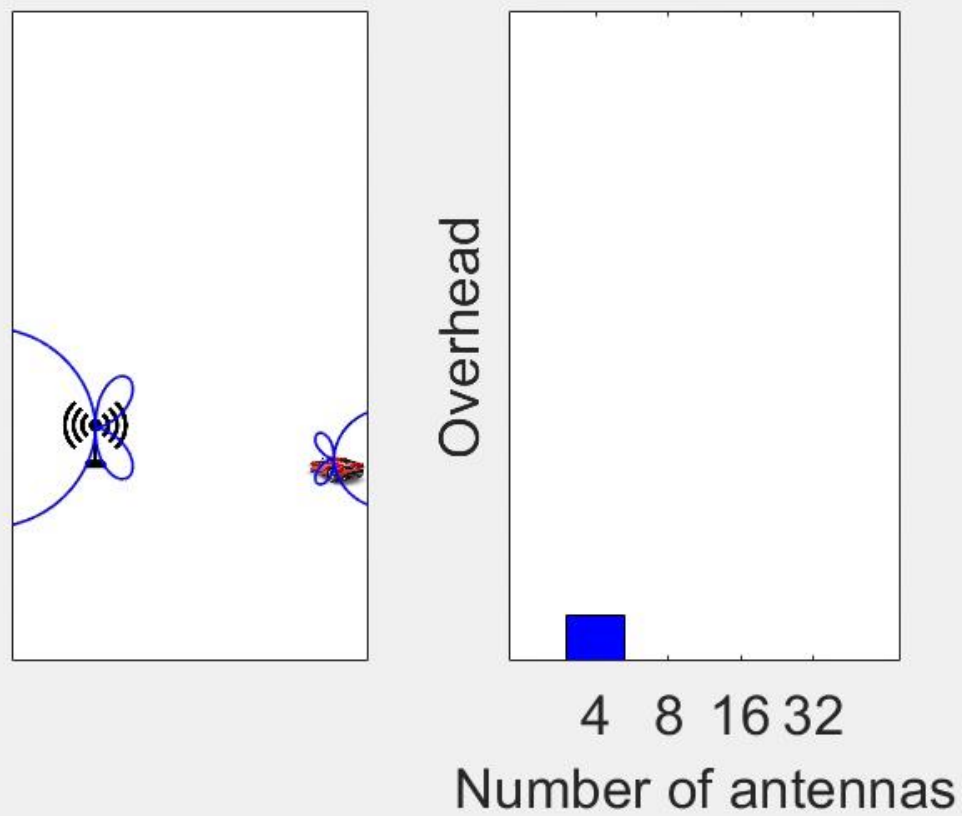
[1] Heath et al, An Overview of Signal Processing Techniques for Millimeter Wave MIMO Systems, 2016

Beam selection in vehicular networks (V2I)

High mobility but predictable trajectories

Communication overhead increases with number of antennas

Machine learning can decrease overhead by choosing subset of candidates



[1] Y. Wang, M. Ribero, M. Narasimha, and R. W. Heath Jr., "MmWave vehicular beam training with situational awareness", 2019
[2] Y. Wang, A. Klautau, M. Narasimha, and R. W. Heath Jr., "MmWave beam selection with situational awareness", 2019

Output of the machine learning module

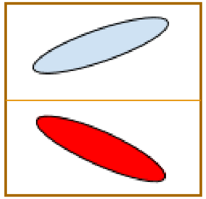
Example with $M_t = M_r = 2$ vectors per codebook

Index

i=1

2

Tx codebook

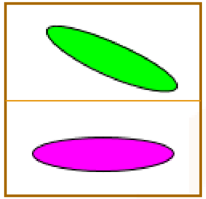


Index

j=1

2

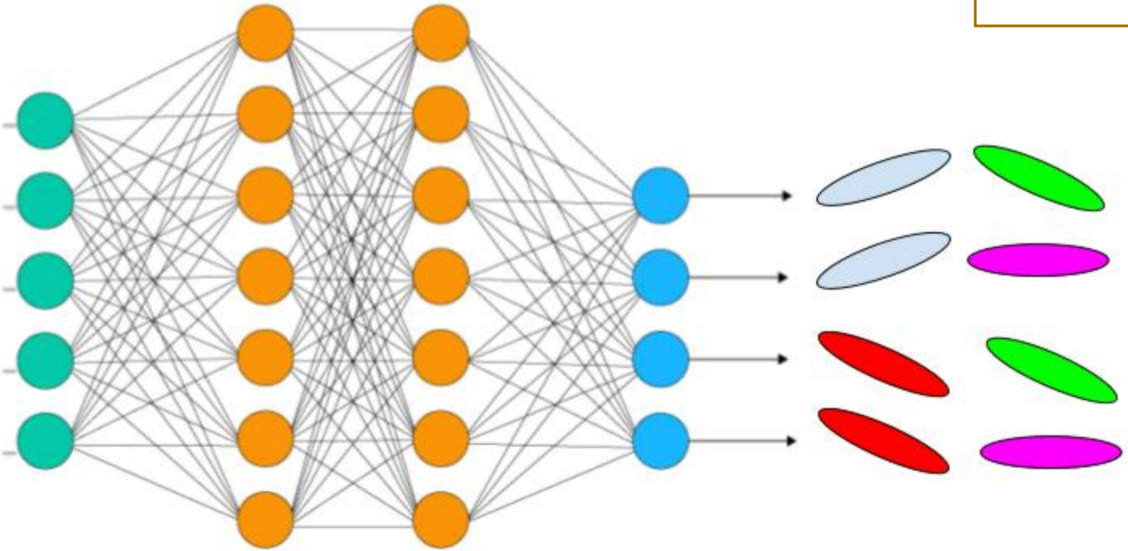
Rx codebook



Goal: maximize magnitude of *combined channel*:

$$y(i,j) = |\boldsymbol{w}_j^H \boldsymbol{H} \boldsymbol{f}_i|$$

Inputs obtained from sensors other than from communication system



Pair or single index	
(1, 1)	0
(1, 2)	1
(2,1)	2
(2,2)	3

High level description of the ITU-ML5G-PS-012 (ML5G-PHY Beam selection)

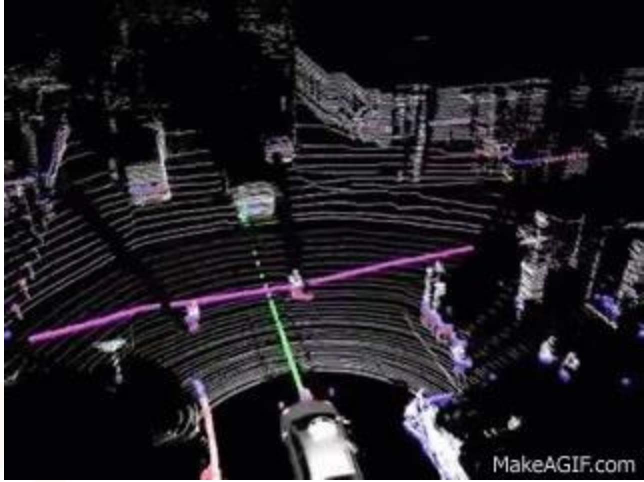
Selection of subset of *k* beams cannot use the communication system

ML can use data from LIDAR, cameras and GNSS (e.g. GPS) for top-*k* classification

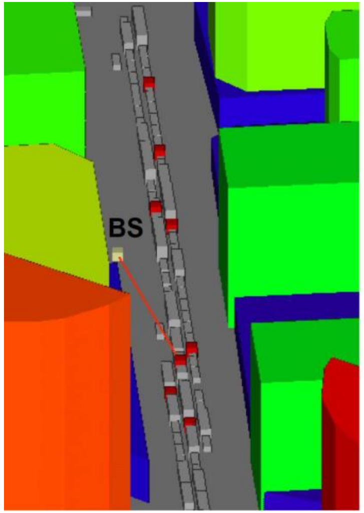
GNSS: Global Navigation Satellite System

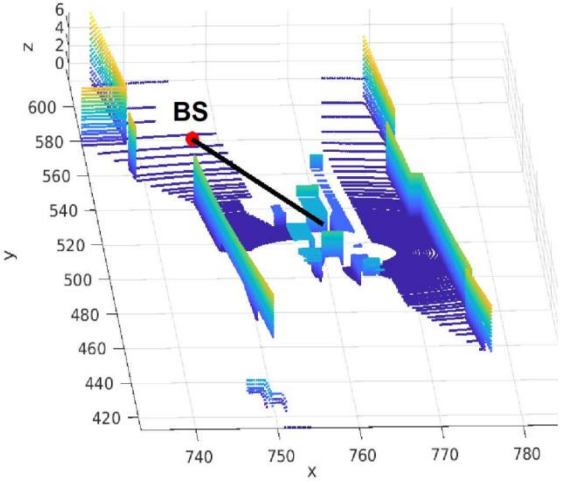
ML-based beam selection using LIDAR

Given the 3D scene of a V2I MIMO communication system and the LIDAR ("light detection and ranging") data from the vehicle's receiver, choose k pairs of beams



A base station (BS) in a urban canyon is the transmitter, while some cars (red) have receivers

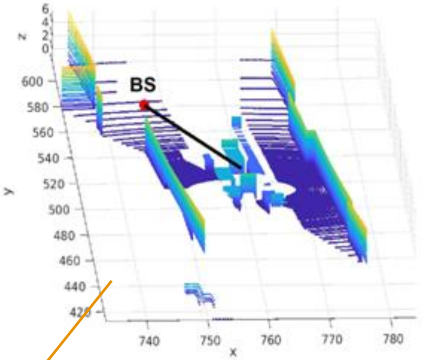




PCD: Point cloud data

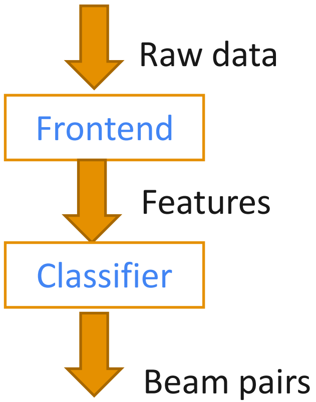
The LIDAR PCD is processed by a frontend and the obtained features are the input to a convolutional neural network

Beam selection using LIDAR data

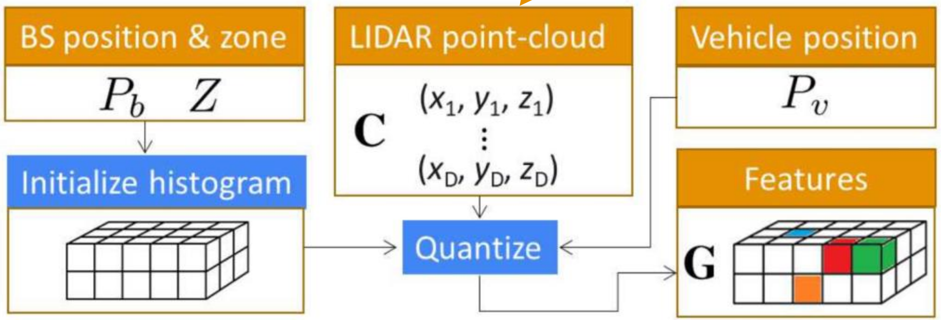


PCD files are large. There are deep networks that can have point clouds as inputs, but here we assume a frontend

.PCD v.7 – PCD file format
VERSION .7
FIELDS x y z rgb
SIZE 4 4 4 4
TYPE F F F F
COUNT 1 1 1 1
WIDTH 213
HEIGHT 1
VIEWPOINT 0 0 0 1 0 0 0
POINTS 213
DATA ascii
0.93773 0.33763 0 4.2108e+06
0.90805 0.35641 0 4.2108e+06
0.81915 0.32 0 4.2108e+06
0.97192 0.278 0 4.2108e+06
0.944 0.29474 0 4.2108e+06
0.98111 0.24247 0 4.2108e+06
0.93655 0.26143 0 4.2108e+06
…



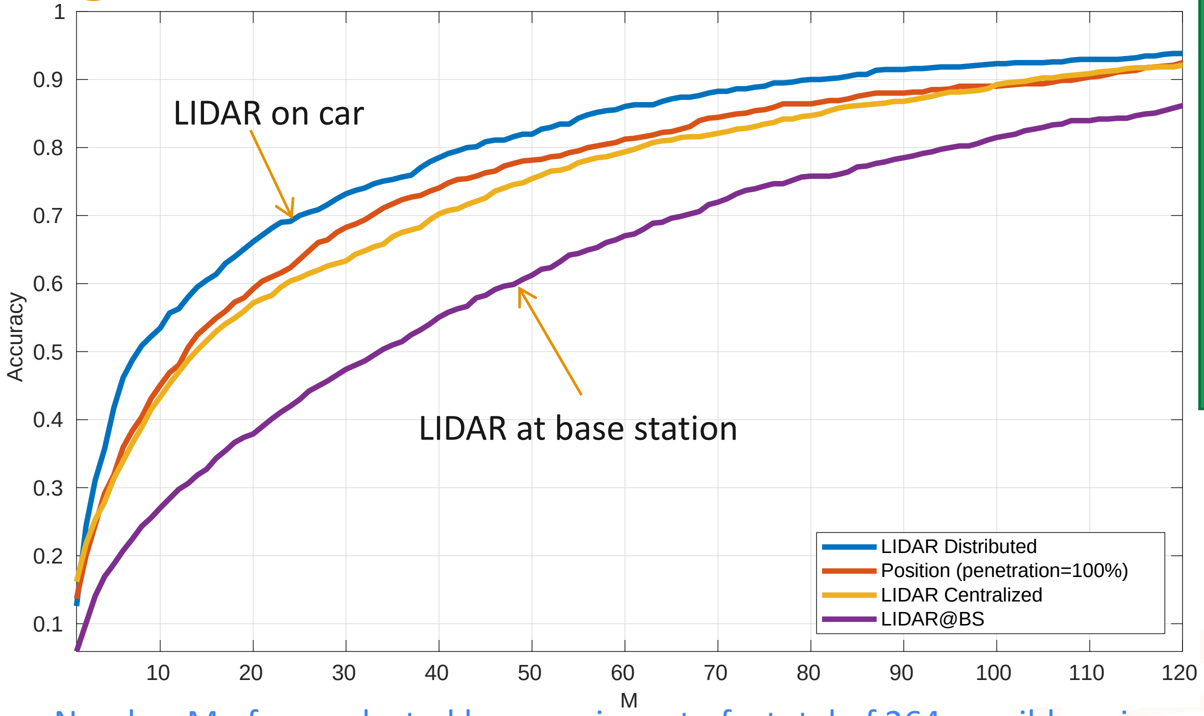
Example of frontend for feature extraction:



For the urban canyon example: **G** can be a 3D grid of dimension 20 x 200 x 10, with elements 1, -1 and -2, indicating obstacle, receiver and BS, or zero otherwise. The **customized frontend** quantizes PCD

[1] http://pointclouds.org/documentation/tutorials/pcd_file_format.htm
[2] A. Klautau, N. González-Prelcic and R. W. Heath, "LIDAR Data for Deep Learning-Based mmWave Beam-Selection", IEEE Wireless Communications Letters, 2019.

Example of results with LIDAR: “Top-M” classification for channels without line-of-sight (harder)



Number M of pre-selected beam pairs out of a total of 264 possible pairs (ITU-ML5G-PS-012 uses 256 pairs)

Deep nets with LIDAR data as inputs enable testing only 40 beam pairs instead of 264 (15% of the original communication overhead)

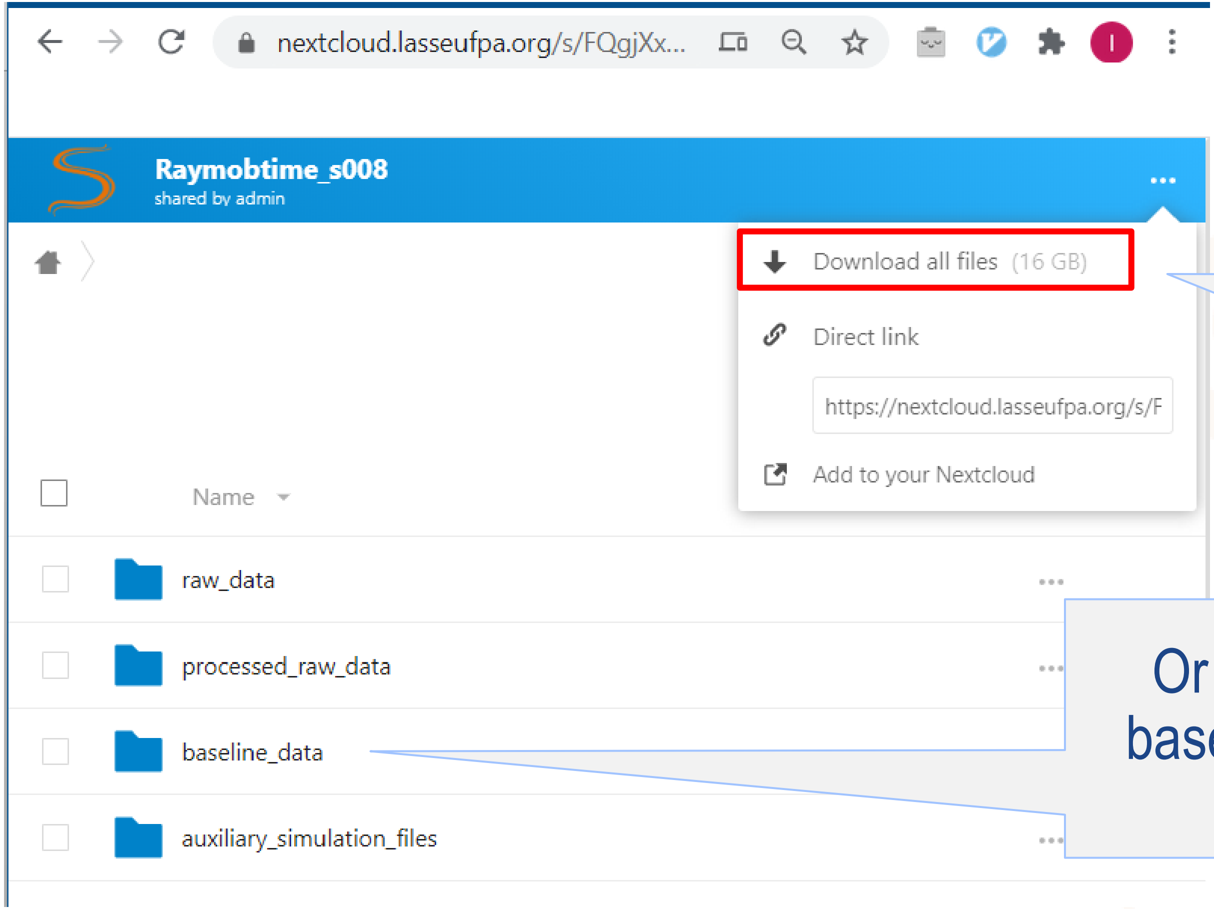
Obs: There figures of merit other than top-M classification, e.g., throughput ration:

$$\mathrm{TR} = \frac{\sum_{i=1}^{N} \log_2(1 + y_{\widetilde{(p,q)}})}{\sum_{i=1}^{N} \log_2(1 + y_{\widehat{(p,q)}})}$$

[1] A. Klautau, N. González-Prelcic and R. W. Heath, “LIDAR Data for Deep Learning-Based mmWave Beam-Selection”, IEEE Wireless Communications Letters, 2019.

Two options: Work with baseline or raw data

Site: http://ai5gchallenge.ufpa.br/ or directly: https://www.lasse.ufpa.br/raymobtime/



Download all data (including raw, ~16 GB)

Or download only baseline data (~512 MB)

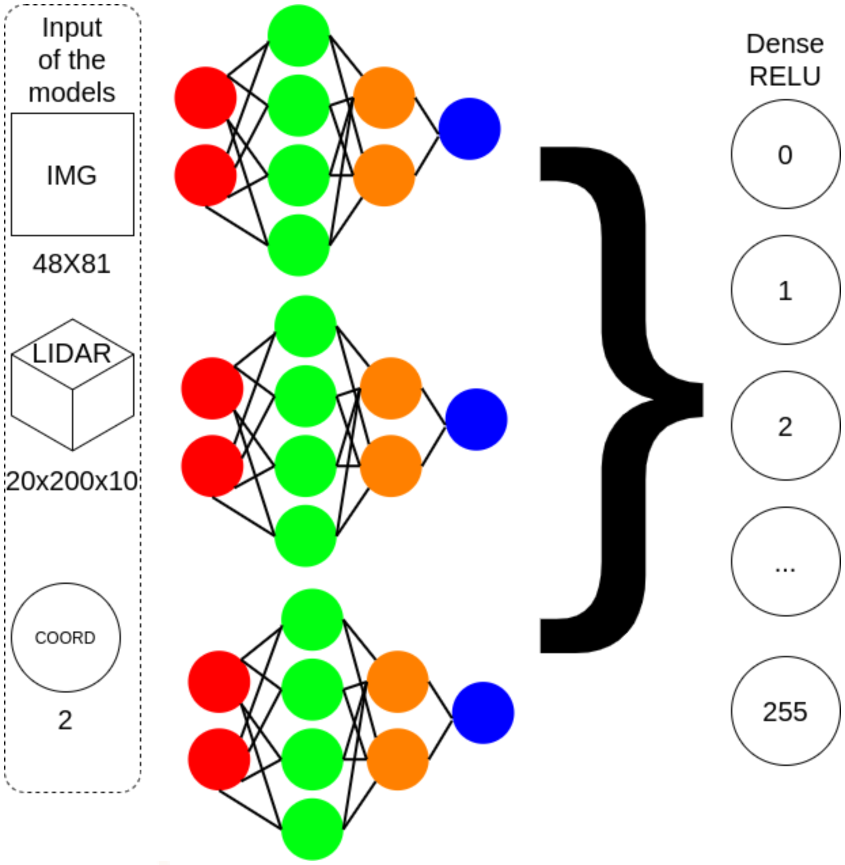
Repository structure:

```
Dataset Home
├──auxiliary_simulation_files
│      s008_Rosslyn_10MobileRx_60GHz_2086episodes_1scenes_Ts1s_InSite3.2.zip
│      s008_SUMO_config_files.zip
│
├──baseline_data          Baseline
│  ├──beam_output
│  │      beams_output_train.npz          Combined channel for each pair of beams
│  │      beams_output_validation.npz
│  │
│  ├──coord_input
│  │      coord_train.npz                 Position features: (x,y) for receiver
│  │      coord_validation.npz
│  │
│  ├──image_input
│  │      img_input_train_20.npz          Resampled images
│  │      img_input_validation_20.npz
│  │
│  └──lidar_input
│         lidar_train.npz                 Quantized LIDAR PCDs
│         lidar_validation.npz
│
├──processed_raw_data
│      lidar_data_s008.zip
│                           Raw
└──raw_data
   │   CoordVehiclesRxPerScene_s008.csv
   │   ray_tracing_data_s008_carrier60GHz.zip
   │   s008_Blensor_rosslyn_scans_lidar.zip
   │
   └──image_data_s008
          camera1.zip
          camera2.zip
          camera3.zip
```

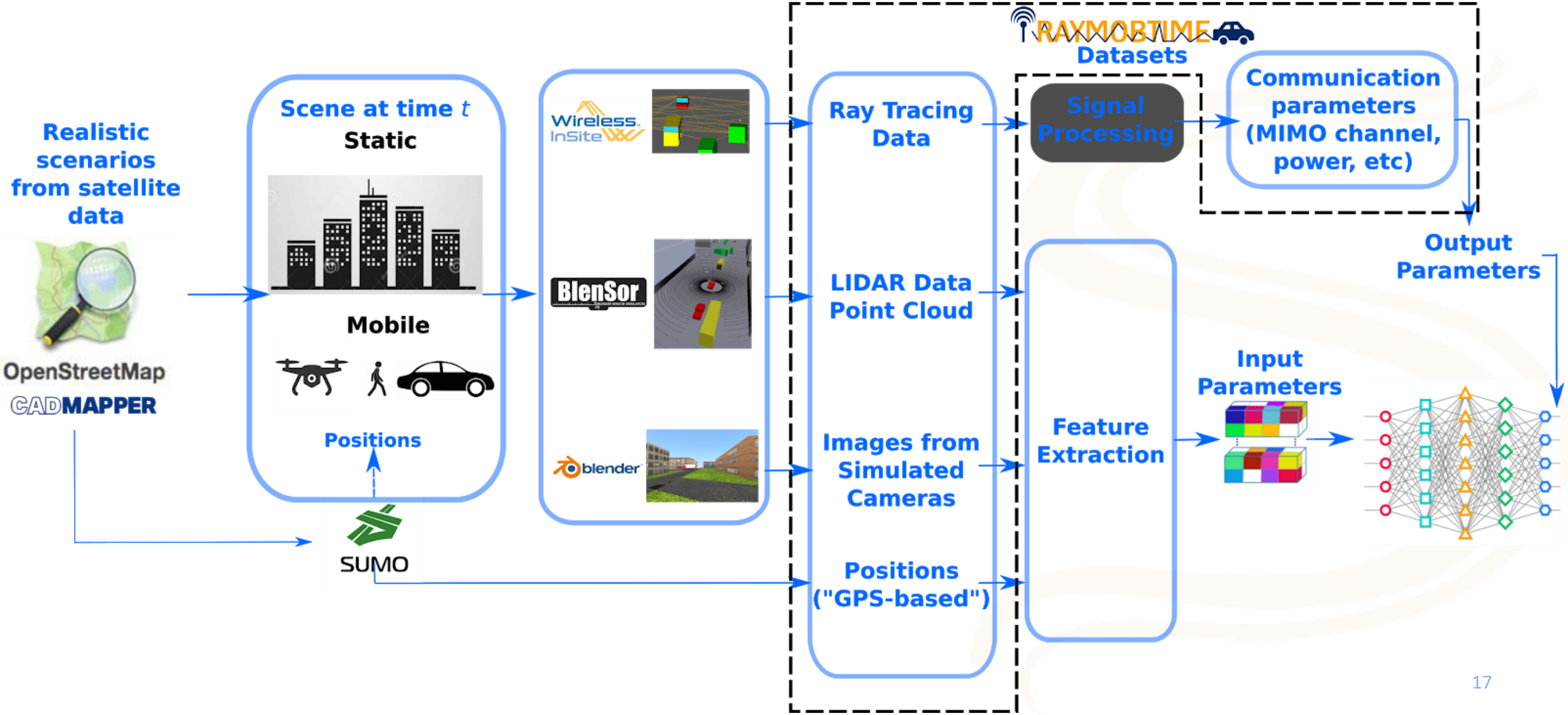
Baseline code

From folder Beam_selection of https://github.com/lasseufpa/ITU-Challenge-ML5G-PHY

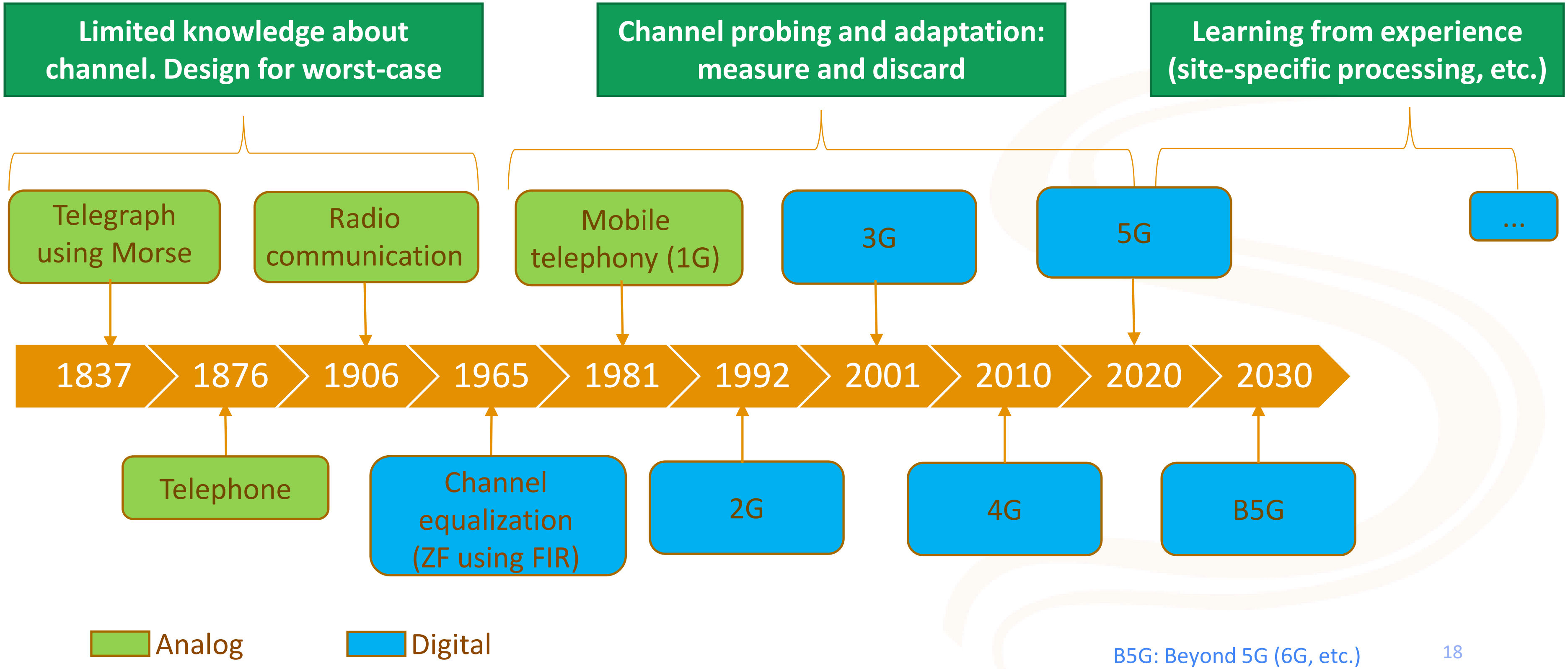
```python
coord_model = modelHand.createArchitecture('coord_mlp',num_classes,
                                           coord_train_input_shape[1],'complete')
img_model = modelHand.createArchitecture('light_image',num_classes,
                                           [img_train_input_shape[1],
                                           img_train_input_shape[2],1],
                                           'complete')
lidar_model = modelHand.createArchitecture('lidar_marcus',num_classes,
                                               [lidar_train_input_shape[1],
                                               lidar_train_input_shape[2],
                                               lidar_train_input_shape[3]],
                                               'complete')

model = Model(inputs=[lidar_model.input,img_model.input,
            coord_model.input],outputs=z)

model.compile(loss=categorical_crossentropy,
            optimizer=opt,
            metrics=[metrics.categorical_accuracy,
                    metrics.top_k_categorical_accuracy,
                    top_50_accuracy])


hist = model.fit([X_lidar_train,X_img_train,X_coord_train],y_train,
        validation_data=([X_lidar_validation, X_img_validation,
                        X_coord_validation], y_validation),
                        epochs=num_epochs,batch_size=batch_size)
```



Raymobtime: understanding the raw data



Motivation for learning Raymobtime (raw data) in ML5G-PHY: timeline



B5G: Beyond 5G (6G, etc.)

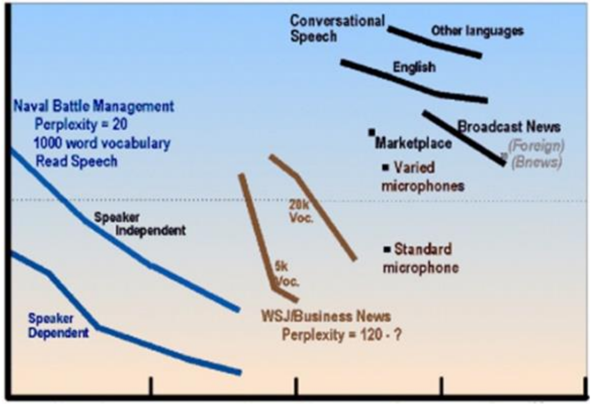
Importance of realistic channel data for 5G and B5G: analogy with speech recognition

Speech recognition

Goal: mimic human brain
Nature gives: spoken language

Datasets by Linguistic Data Consortium and DARPA competitions

log WER

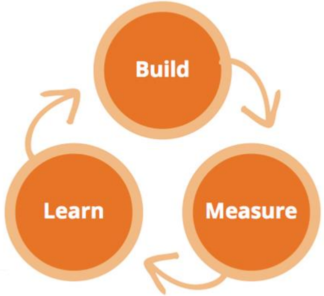


year

Physical layer of mobile communication systems

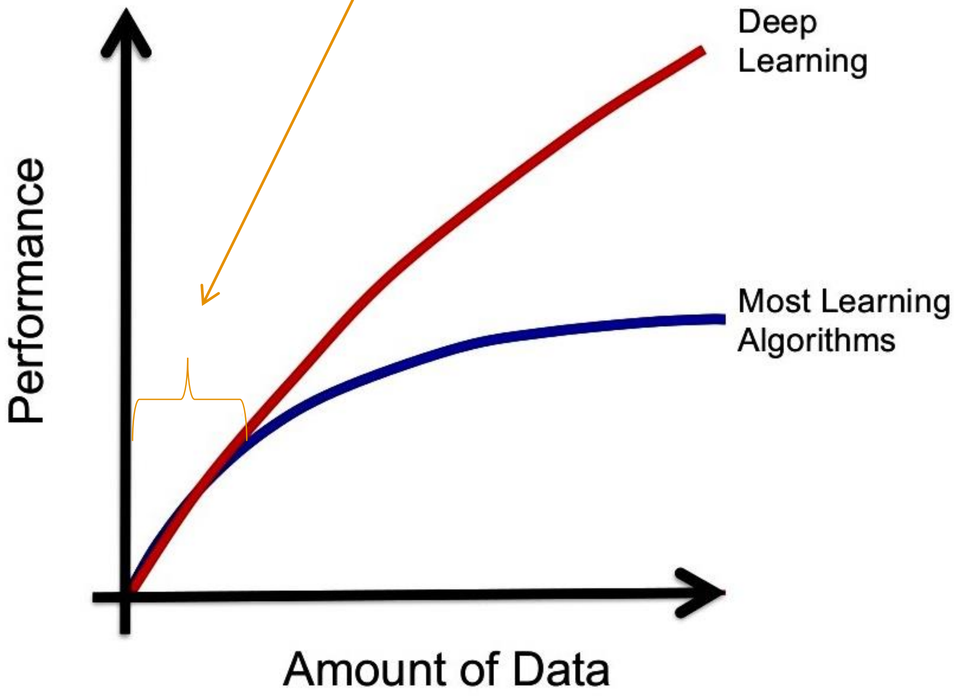
Goal: design comm system
Nature gives: channel

ITU AI/ML in 5G Challenge



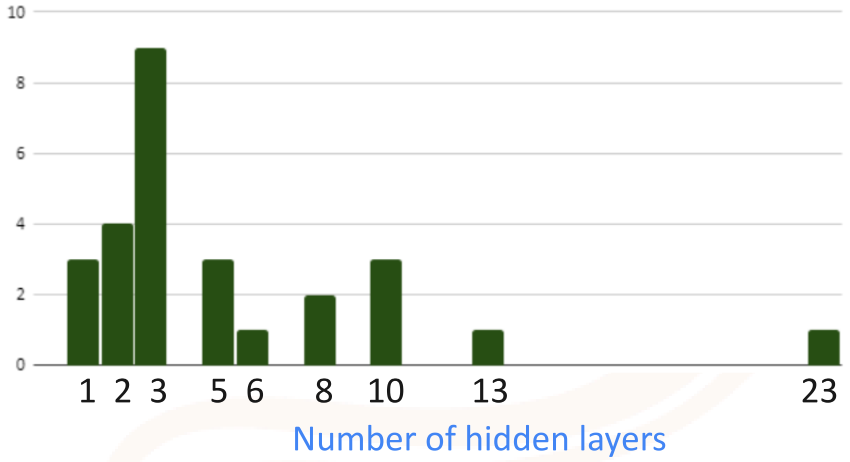
For instance, we do need data for advanced deep learning in 5G

Escape the **small data** regime



Papers (~40) published in 2019 in IEEE ComSoc journals, collected via IEEExplore on Aug 16, 2019, and having deep learning in ther titles

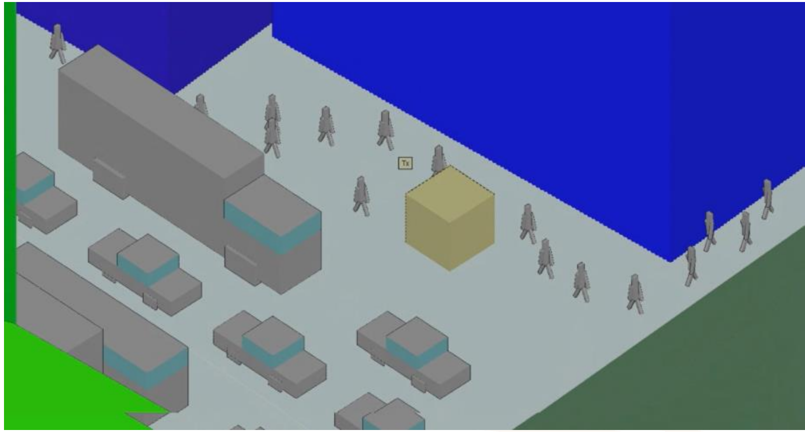
Histogram:



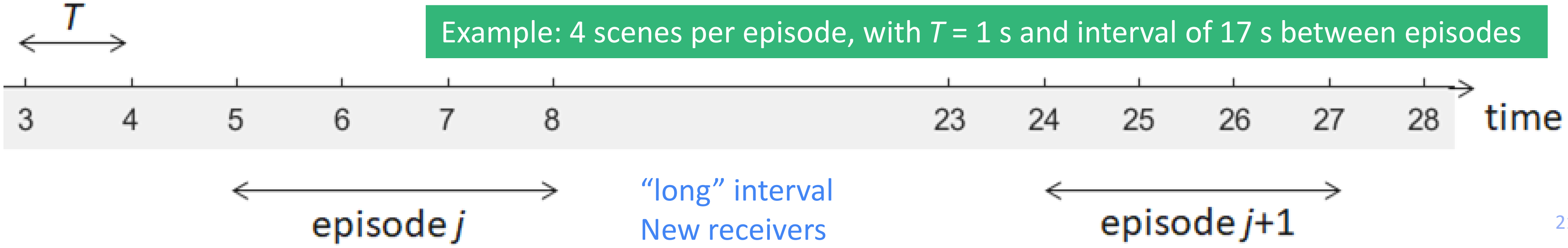
Raymobtime: control mobility and incorporate time dimension

Information periodically stored as snapshots (**scenes**) spaced by *T* seconds

Group of scenes composes an **episode**, which are supposed to be independent (while scenes are correlated)



Example: 4 scenes per episode, with T = 1 s and interval of 17 s between episodes



Invoke ray-tracing software for each "snapshot"

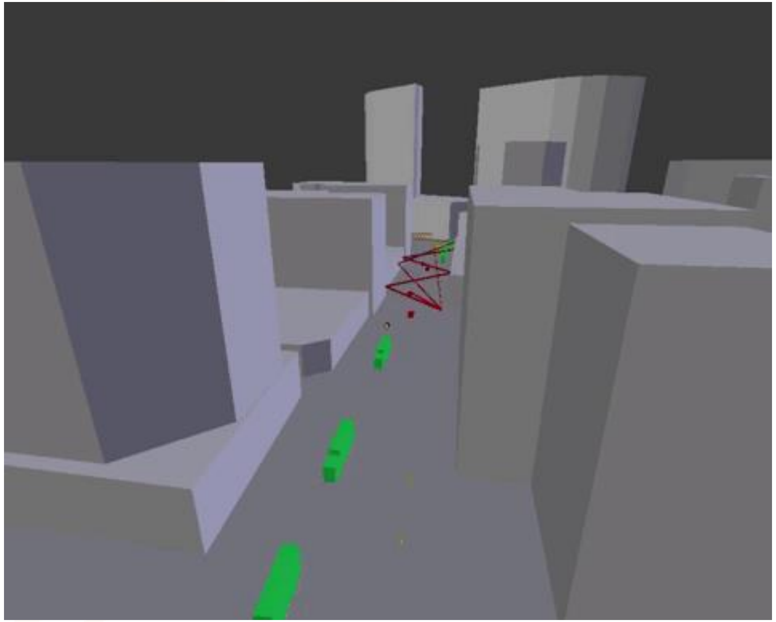
1 **for** *each t ("snapshot")* **do**

2 SUMO: positions of vehicles, pedestrians, etc.;

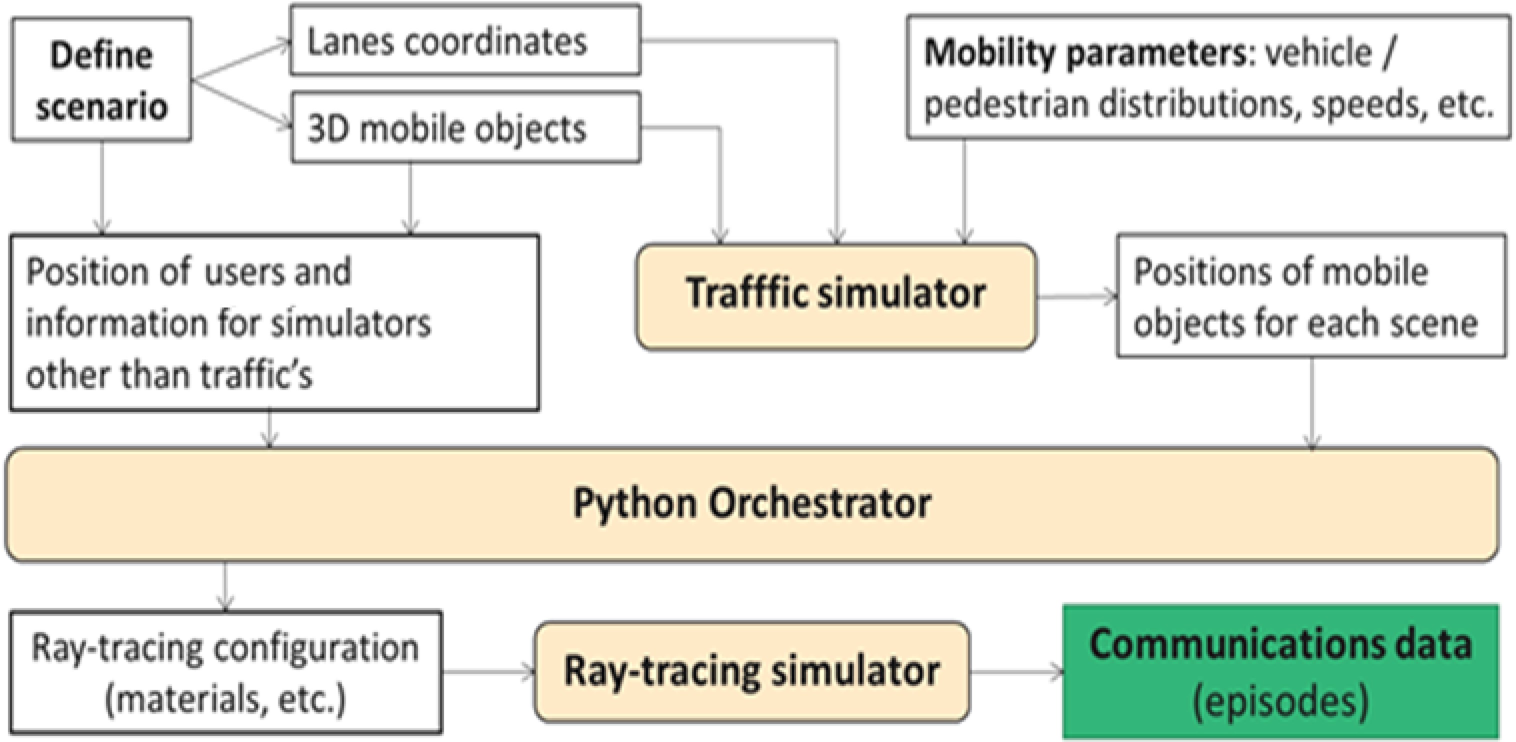
3 Wireless Insite: channels via ray-tracing;

Control mobility via open-source SUMO

Ray-tracing performed by Remcom's Wireless Insite → channels with *consistency* over time, frequency and space



Positioning mobile objects for ray-tracing



Outdoor site-specific data by collecting realistic 3D scenarios from Web

Dongcheng District, Beijing

Rosslyn, Virginia, USA

Kista, Stockholm

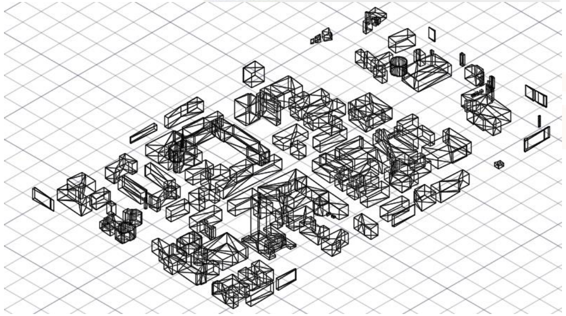
Some of the steps to obtain realistic outdoor 3D Scenario from Web

CADMAPPER

- Export 3D CAD file
- Complemented with terrain (*.ter) profiles from open sources websites as NASA

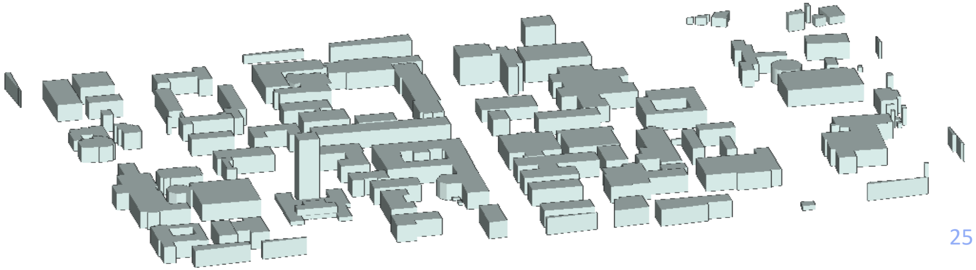
AutoCad

- Organize CAD by layers
- Convert to format supported by InSite (polyface meshes and polylines)



Wireless Insite

- “Building” layer imported as a “feature” called “city”
- Assign electromagnetic properties to each feature



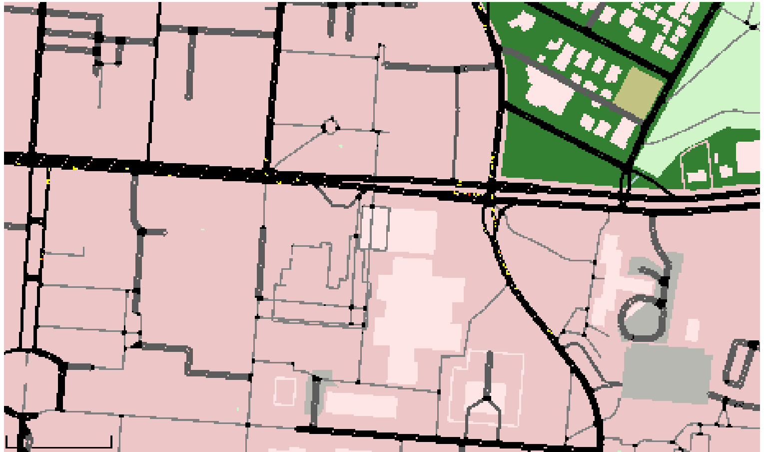
Realistic Mobility Scenario from Web Data

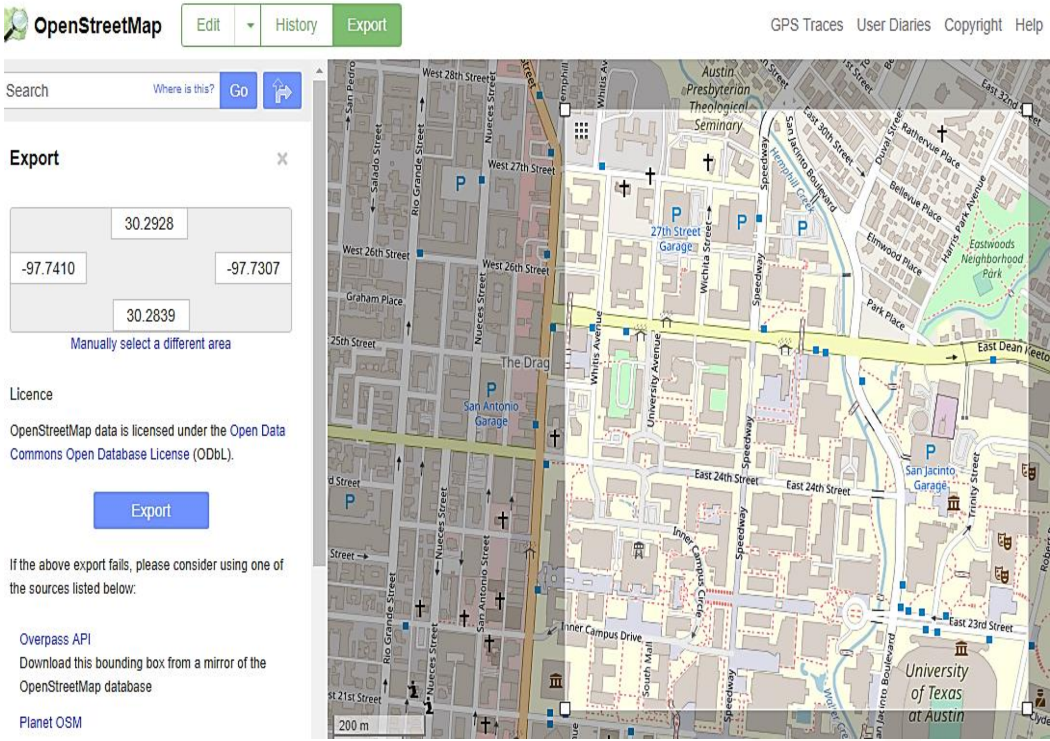
OpenStreetMap

- Export streets
- OSM file

SUMO

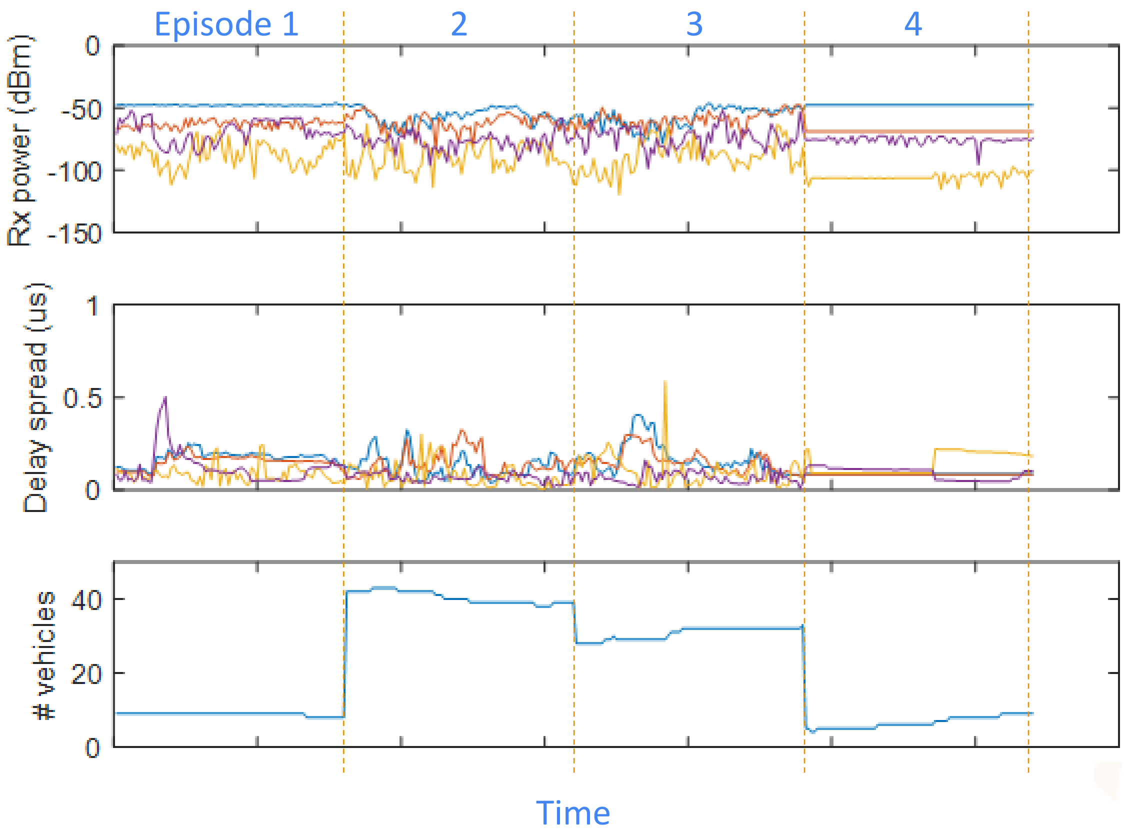
- Import converted streets
- Generate vehicles traffic





26

Example of mobility-controlled generated data



Examples of large scale parameters for 4 receivers using 4 episodes of 80 scenes each, with time sampled at T_s = 50 ms

We can model "seasonal" aspects, mimicking mobility over "day", "night", etc.

Raymobtime datasets

Currently ~316 k channels (multimodal, paired frequency bands, distinct sites, etc.)

Id	Insite Version	3D Scenario	Carrier frequencies (GHz)	# Rx and type	Time between scenes	Time between episodes	# episodes	# scenes per episode	# valid channels
s000	3.2	Rosslyn	60	10 Mob	100 ms	30 s	116	50	41 K
s001	3.2	Rosslyn	2.8; 5	10 Fix	5 ms	37 s	200	10	20 K
s002	3.2	Rosslyn	2.8; 60	10 Fix	1 s	3 s	1800	1	18 K
s003	3.2	Rosslyn	2.8; 5	10 Fix	1 ms	35 s	200	10	20 K
s004	3.2	Rosslyn	60	10 Mob	1 s	30 s	5000	1	35 K
s005	3.2	Rosslyn	2.8; 5	10 Fix	10 ms	35 s	125	80	100 K
s006	3.2	Rosslyn	28; 60	10 Fix	1 ms	35 s	200	10	20 K
Multimodal: s007	3.3	Beijing	2.8; 60	10 Mob	1 s	5 s	50	40	15 K
s008	3.2	Rosslyn	60	10 Mob	100 ms	30 s	2086	1	20 K
s009	3.3	Rosslyn	60	10 Mob	500 ms	10.5 s	2086	42	27 K

Raymobtime data can be used in other ML5G problems related to the physical (PHY) layer such as:
- Channel estimation / channel tracking
- Channel state information (CSI) compression

From s008_Rosslyn_10MobileRx_60GHz_2086episodes_1scenes_Ts1s_InSite3.2.zip

Information from Remcom's ray-tracing and SUMO

- base
- run00000
- run00001
- run00002
- run00003
- run00004
- run00005
- run00006
- run00007
- run00008
- run00009
- run00010
- run00011
- run00012
- run00013
- run00014
- run00015
- run00016
- run00017
- run00018
- run00019

- study
- base.object
- base.Study.xml
- base.txrx
- config.py
- gen.study.diag
- gen.study.xml
- model.setup
- model.study.diag
- model.study.xml
- model.txrx
- model.vw
- random-line.object
- Rosslyn.city
- Rosslyn_DTED.ter
- sumoOutputInfoFileName.txt
- wri-simulation.info

```
"episode_i,scene_i,receiverIndex,veh,veh_i,typeID,xinsite,
1,0,-1,flow1.3,0,Car,821.0478041067474,670.531615620921 5,1
1,0,0,flow1.4,1,Truck,701.569636340602 8,670.610483916960 2,
1,0,-1,flow10.2,2,Truck,825.873151985477 2,376.063572094098
1,0,-1,flow11.2,3,Truck,753.017692519877,660.672923207514 5
1,0,1,flow11.3,4,Truck,755.619990034731 4,623.488703457774 6
1,0,-1,flow11.4,5,Car,798.833763210986 3,397.061843930584 13
1,0,-1,flow12.0,6,Car,749.287051083529 3,666.739242829980 6,
1,0,-1,flow12.1,7,Bus,749.632344178247 1,661.805142896 81,9
1,0,2,flow12.2,8,Car,752.297551298397 7,623.720396047383 7,9
```

Comma-separated values (CSV) text file

Each "run" is a scene with at most 10 "valid" receivers. The rays information is saved as hdf5 files using NaN to indicate invalid receivers

Example of a sumoOutputInfoFileName.txt

episode_i	scene_i	receiverIndex	veh	veh_i	typeID	xinsite	yinsite	x3	y3	z3	lane_id	angle	speed	length	width	height
0	1	0	flowA.0	0	Car	326.7969	221.0916	1450.3269	5454.0116	0.0	L12_1	181.4397	16.0524	4645	1775	1.59
0	1	1	flowA.1	1	Car	327.8680	263.2337	1451.3980	5496.1537	0.0	:734520421_1_0	181.9185	15.1823	4645	1775	1.59
0	1	2	flowA.2	2	Car	329.4418	303.2133	1452.9718	5536.1333	0.0	L13_1	182.4141	14.2401	4645	1775	1.59
0	1	3	flowA.3	3	Truck	337.4475	369.7200	1460.9775	5602.6400	0.0	L14_1	179.7979	13.1329	12.5	2.5	4.3
0	1	4	flowA.4	4	Truck	337.2692	420.2615	1460.7992	5653.1815	0.0	L14_1	179.7979	11.3736	12.5	2.5	4.3
0	1	-1	flowA.5	5	Truck	337.1295	459.8664	1460.6595	5692.7864	0.0	L14_1	179.7979	3.5375	12.5	2.5	4.3
0	1	-1	flowA.6	6	Car	337.0768	474.8257	1460.6068	5707.7457	0.0	L14_1	179.7979	0.0000	4645	1775	1.59
0	1	5	flowB.0	7	Bus	329.4690	200.0077	1452.9990	5432.9277	0.0	L21_1	1.4397	12.7879	9.0	2.4	3.2
0	1	6	flowB.1	8	Truck	328.5713	164.2898	1452.1013	5397.2098	0.0	L21_1	1.4397	14.4235	12.5	2.5	4.3
0	1	7	flowB.2	9	Truck	327.7365	131.0727	1451.2665	5363.9927	0.0	L21_1	1.4397	12.3693	12.5	2.5	4.3
0	1	8	flowB.3	10	Car	327.3777	89.4747	1450.9077	5322.3947	0.0	L11#V_1	356.7047	4.4145	4645	1775	1.59
0	1	9	flowB.4	11	Truck	328.5826	68.5476	1452.1126	5301.4676	0.0	L11#V_1	356.7047	7.0470	12.5	2.5	4.3
0	1	-1	flowB.5	12	Truck	329.7657	47.9995	1453.2957	5280.9195	0.0	L11#V_1	356.7047	0.0000	12.5	2.5	4.3

- “episode_i”: Episode
- “scene_i”: Scene
- “receiverIndex”: Rx 1 to 10, or -1 to invalid
- “veh”: SUMO flow identification
- “veh_i”: ID of vehicle in SUMO
- “typeID”: type of vehicle

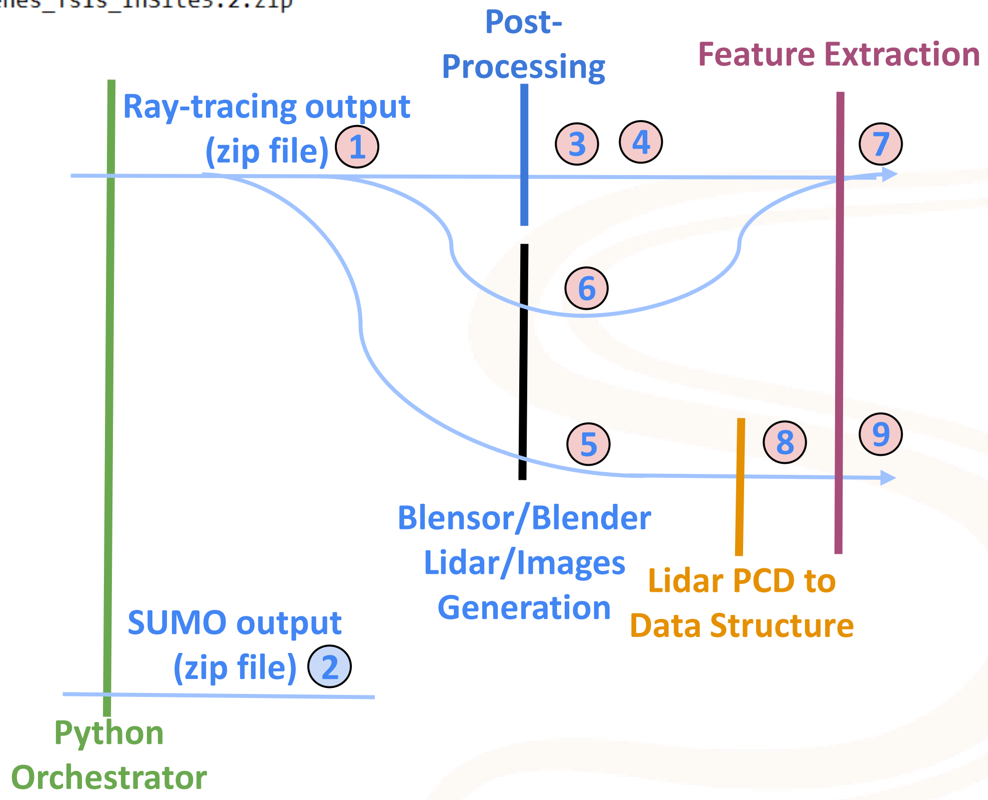
- “xinsite” and “yinsite”: coordinates of vehicles in InSite
- “x3”, “y3” and “z3”: coordinates of vehicles in SUMO
- “lane_id”: SUMO street ID,
- “angle”: Angle of the receiver/vehicle in relation to the North (positive Y)
- “speed”: Speed of the receiver/vehicle
- “length”, “width” and “height”: dimensions of the vehicle,

CoordVehiclesRxPerScene_s008.csv

Concatenates information from all sumoOutputInfoFileName.txt files and shows that the s0008 dataset has 2086 scenes, 11194 (out of 20860) receivers with valid communication channels and 6482 (57.9%) in line-of-sight (LOS)

	A	B	C	D	E	F	G	H	I	J
1	Val	EpisodeID	SceneID	VehicleArrayID	VehicleName	x	y	z	rays	LOS
2	I	0	0	0	flow1.2	704.18681848265	670.58971778045	1.59	25	LOS=0
3	V	0	0	1	flow11.0	753.3833848633	655.44754663915	4.3	25	LOS=0
4	I	0	0	2	flow12.0	783.94351941255	408.6253222917	1.59	25	LOS=1
5	I	0	0	3	flow12.1	832.2173789167	371.13676687655	3.2	25	LOS=0
6	V	0	0	4	flow3.0	756.6564658932	514.19045865405	1.59	25	LOS=1
7	V	0	0	5	flow6.1	746.9595274421	652.75583972565	1.59	25	LOS=1
8	I	0	0	6	flow7.2	748.77667946875	420.2445837159	4.3	20	LOS=0
9	I	0	0	7	flow7.3	702.74528273365	445.1358988597	3.2	25	LOS=0
10	I	0	0	8	flow7.4	688.1055462007	452.2591859074	1.59	25	LOS=0
11	V	0	0	9	flow9.0	765.5127845235	433.64539792815	3.2	25	LOS=1
...										
20858	V	2085	0	6	flow9.2527	753.8278324045	649.0968373349	1.59	25	LOS=1
20859	V	2085	0	7	flow9.2528	760.93045037305	547.60755494775	3.2	25	LOS=1
20860	V	2085	0	8	flow9.2529	765.0030061009	489.41482092565	1.59	25	LOS=1
20861	I	2085	0	9	flow9.2531	669.807737267	451.45323012395	3.2	25	LOS=0

```
Dataset Home
├──auxiliary_simulation_files
│  1 → s008_Rosslyn_10MobileRx_60GHz_2086episodes_1scenes_Ts1s_InSite3.2.zip
│  2 → s008_SUMO_config_files.zip
│
├──baseline_data
│   ├──beam_output                                   ┐
│   │      beams_output_train.npz                    │
│   │      beams_output_validation.npz               │
│   │                                                │
│   ├──coord_input                                   │
│   │      coord_train.npz                           ├ 7
│   │      coord_validation.npz                      │
│   │                                                │
│   ├──image_input                                   │
│   │      img_input_train_20.npz                    │
│   │      img_input_validation_20.npz               ┘
│   │
│   └──lidar_input  ← 9
│          lidar_train.npz
│          lidar_validation.npz
│
├──processed_raw_data  ← 8
│      lidar_data_s008.zip
│
└──raw_data
    │  CoordVehiclesRxPerScene_s008.csv  ← 4
  3 → ray_tracing_data_s008_carrier60GHz.zip
    │  s008_Blensor_rosslyn_scans_lidar.zip  ← 5
    │
    └──image_data_s008  ← 6
           camera1.zip
           camera2.zip
           camera3.zip
```



- **Registration is now open to all participants**
- **Challenge duration** → May 22nd-Oct 21st (≈5-month duration)
- **Registration deadline** → Jun 30th
- **Winners (top 3) official announcement** → Oct 21st
- **ITU final conference and awards** → Nov-Dec 2020

Details at http://ai5gchallenge.ufpa.br/

Thanks to all Raymobtime team

Aldebaro Klautau

Ailton Oliveira

Arthur Nascimento

Brenda Vilas Boas

Ilan Correa

Diego Gomes

Francisco Müller

Robert Heath Jr

Isabela Trindade

Marcus Dias

Virgínia Tavares

Pedro Batista

Walter Tadeu

Nuria González-Prelcic

Yuyang Wang

Jamelly Ferreira

ITUEvents

ITU-ML5G-PS-025: ML5G-PHY: Channel estimation (NC State University, USA)

3 July 2020

ITU

AI/ML in 5G

Challenge

Applying machine learning in communication networks

ai5gchallenge@itu.int

Bronze sponsor:

Organized by: