List of Abbreviations

AE	Antenna Element
BLER	Block Error Rate
BS	Base Station
CSI	Channel State Information
DL	Downlink
GoB	Grid of Beams
GoP	Group of Pictures
HMD	Head Mounted Display
PRB	Physical Resource Block
SINR	Signal-to-Interference-plus-Noise Ratio
TB	Transport Block
TDD	Time Division Duplex
TTI	Transmission Time Interval
UE	User Equipment
UL	Uplink

List of Symbols

Latin alphabet

a_ϕ	azimuth lower limit for GoB
a_θ	elevation lower limit for GoB
b_ϕ	azimuth upper limit for GoB
b_θ	elevation lower limit for GoB
B	bandwidth
$BLER_0$	target BLER
C_t	coordinates of centre of the table
C_m	coordinates of centre of mass of the room
d_f	distance to the front of the user for camera placement
d_F	Fraunhofer distance
d_s	distance to the side of the user for camera placement
d_{out}	distance outwards from the head centre for HMD antenna offset
d_{up}	distance upwards from the head centre for HMD antenna offset
D	largest dimension of radiator to estimate effective area
E_b	energy per bit
$\boldsymbol{H}_{bul}$	channel matrix between BS b and UE u in layer l
I	interference
j	imaginary unit $\left(j = \sqrt{-1}\right)$
k_B	Boltzmann constant
L_{max}	maximum latency for the radio link
N_0	noise power spectral density
N_r	number of receive antennas
N_t	number of transmit antennas
N_{users}	number users or participants
N_{phy}	number of physical users
N_{vir}	number of virtual users
N_{CSI}	number of beams with CSI-RS
N_{bs}	number of BSs
N_{ue}	number of UEs
N_{cam}	number of cameras
N_x	number of antennas along the x-dimension
N_y	number of antennas along the y-dimension
N^{DL}_{slots}	number of DL slots in a transmission period
N^{UL}_{slots}	number of UL slots in a transmission period
N^{TDD}_{slots}	number slots in a transmission period
N^{CSI}_{slots}	number slots between CSI updates
N^{SCH}_{slots}	number slots between scheduling updates
N^{UE}_{ant}	number of antennas on the UE side
N^{BS}_{ant}	number of antennas on the BS side
NF_{BS}	BS noise figure
NF_{UE}	UE noise figure

Greek alphabet

α_P	power compensation factor for UL power control
α_u	angle from the centre of the table to user u
β_x	upper limit on uniform distribution for rotation around x axis
β_y	upper limit on uniform distribution for rotation around y axis
β_z	upper limit on uniform distribution for rotation around z axis
γ	burstiness parameter for application traffic
γ_{OLLA}	step size for OLLA parameter (Δ_{OLLA}) update
Δ_{OLLA}	outer loop link adaptation step
η_{OH}	efficiency due to overhead
η_{slot}	efficiency in bit rate from slot format
λ	wavelength
o	overlap parameter for application traffic
σ_x	standard deviation of normal distribution for position coordinate x
σ_y	standard deviation of normal distribution for position coordinate y
σ_z	standard deviation of normal distribution for position coordinate z
τ_{CSI}	CSIs delay, in number of TTIs
τ_{ACK}	delay before acknowledgement, in number of TTIs

Sets:

$\mathbb{N}$	natural numbers
$\mathbb{N}_0$	natural numbers including zero
$\mathcal{B}$	base station panels
$\mathcal{U}_b$	users served by base station b
$\mathcal{L}_{bu}$	layers scheduled between base station b and user equipment u
$\mathcal{F}$	frequencies to simulate

Other nomenclature

$\boldsymbol{A}$	matrix
$\boldsymbol{a}$	column vector
$\lvert\boldsymbol{a}\rvert$	euclidean norm of vector $\boldsymbol{a}$
$\boldsymbol{A}^{\mathsf{T}}$	transpose of $\boldsymbol{A}$
$\boldsymbol{A}^{\mathsf{H}}$	Hermitian of $\boldsymbol{A}$, also know as the transpose conjugate of $\boldsymbol{A}$
$\lceil a \rceil$	ceil a, i.e. round up a to the nearest integer
$\lfloor a \rfloor$	floor a, i.e. round down a to the nearest integer
$\hat{a}$	estimate of a
$\overline{a}$	average of a

Bibliography

A

On the Implementation of a System-level Simulator

Contents

There are many more parameters than the ones presented here. For instance, it is possible to use different bandwidths and different numbers of PRBs in different frequencies, different antennas across users, place users around rectangular tables or in non-standard places, hybrid architectures for each antenna, change the height across users, put separate UEs in each camera, among plenty of other things. The total number of setting variables, both for channel trace generation and for actual simulation, is ... (COUNT)

As of now, the simulator counts with more than 18000 lines of code, counting empty lines and comments as well.

A.1 SXR Radio Channel Generation

The Radio Channel Generation process uses Quadriga as a channel generator.

It's programmed in Matlab and compiled to an executable. That executable is then run in different subprocesses/threads to parallelise the workload and reducing the computation time of this phase.

In this section, firstly we list the parameters required to generate the channel traces and the input arguments used to execute the each instance. We include a short summary for each parameter and a more detailed description of the arguments.

The arguments are particularly important since they it's through them that Python and Matlab interface. So, they will control the phases of the radio channel simulation.

Subsequently, each section exposes the well-defined phases of the generation process, and we'll understand how these parameters come together.

A.1.1 Parameters and Arguments

With 3 arguments and 98 parameters (as of writing this)

The three arguments are:

- *flow_control* - controls the flow of the program. It is required to execute individual parts of the code, which is needed for the practicality of using a single executable in the while parallelising, as described ahead;
- *instance_info* - used for parallelisation management. It is a 2 element array. The first element is the parallelisation level with values from 0 to 3 representing, respectively, no parallelisation, parallelisation on the frequency level, on the BS-level or on the UE-level. Figure **??**, shows this difference. This influences the number of instances each time division will be further partitioned in and

should be more clear after a look on how Quadriga works. The second element depends on the *flow_control* value. In the setup phase ($flow\_control \in \{1, 2\}$) it tells Matlab how many high-level instances the simulation should be prepared for, this will lead to the generation of those many builders. In the execution phase, it tells Matlab which instance should be executed. As expected, the value in the execution phase should not pass the value given in the setup phase as the maximum;

The table below summarises how the arguments change value with *flow_control*:

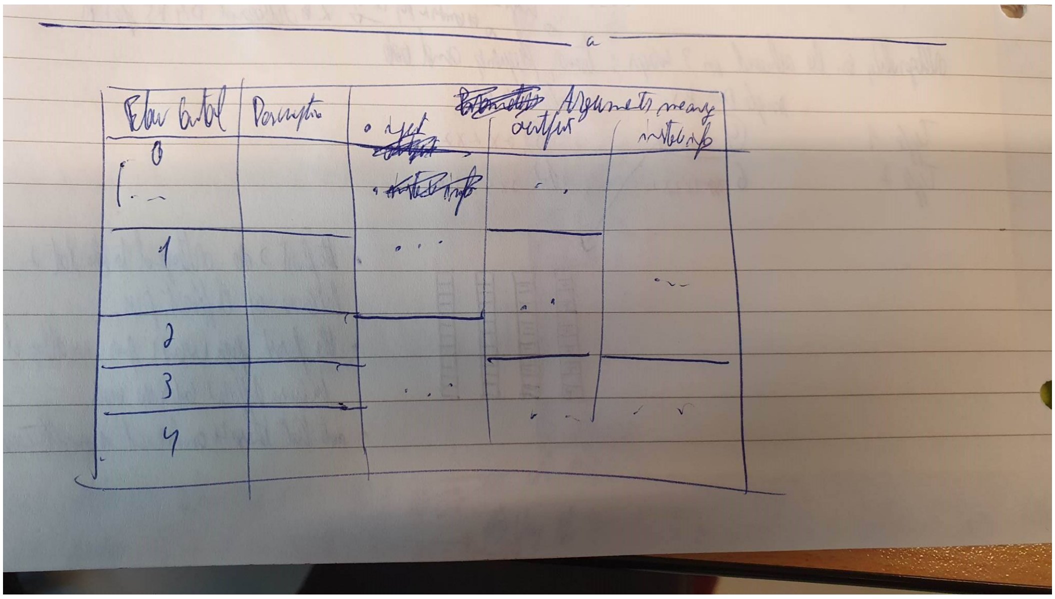

Figure A.1: Awaiting SLS completion.

Channel Generation in the Simulation Flow

Now that we have chosen the Channel Model, back to the simulation flow.

By the end of the simulation, the result is a channel in time domain, with a set of coefficients of the form of expression (A.1). In order, one reads below the UE index, the BS index, the frequency index, the number of Antenna Elements (AEs) in that pair of UE-BS, for all paths in a given time instant, or like Quadriga calls them, snapshots.

$$\begin{aligned} \mathsf{C}(\mathsf{UE}_m, \mathsf{BS}_b, \mathsf{F}_1) = [\mathsf{N}_{\mathsf{AE}_{\mathsf{UE}_m}}, \mathsf{N}_{\mathsf{AE}_{\mathsf{BS}_b}}, \mathsf{N}_{\mathsf{paths}}, \mathsf{snap}]\ , \\ \forall m \in \{1, \ldots, \mathsf{N}_{\mathsf{UE}}\}, b \in \{1, \ldots, \mathsf{N}_{\mathsf{BS}}\} \end{aligned} \tag{A.1}$$

Further note that there'll be a snapshot for each TTI. Since we need information across the carrier frequency, we further convert the time domain channel to frequency domain. It's to the Frequency Response that we call Channel Coefficients. Below, (A.2) shows the dimensions of this matrix.

$$\text{FR}(\text{UE}_m, \text{BS}_b, \text{F}_{\text{f\_idx}}) = [\text{N}_{\text{AE}_{\text{UE}_m}}, \text{N}_{\text{AE}_{\text{BS}_b}}, \text{N}_{\text{PRB}}, \text{snap}] , \tag{A.2}$$
$$\forall m \in \{1, \ldots, \text{N}_{\text{UE}}\}, b \in \{1, \ldots, \text{N}_{\text{BS}}\}, \text{f\_idx} \in \{1, ..., \text{N}_{\text{Freq}}\}$$

This is the end result. How to get to the end result, after the all the setup of the previous section, the next steps are preparing for a parallelisation, if that's the case.

If the instance is running in parallel, this phase first splits the tracks, and then creates a builder for each segment of the track. Effectively, each track is a sequence of positions at certain instants. Hence splitting a track is making a separation in time.

The channel is saved in time domain. This allows the conversion to multiple bandwidths later. Therefore, each simulation only requires the centre of the frequency band (i.e. 3.5 GHz, 26 GHz) and it's possible afterwards to convert to a 25 MHz carrier or to 400 MHz carrier.

Note that the resolution in the frequency domain (spaces PRBs) is inverse to the resolution in the time domain (interval between samples). As a result, we need to simulate the time domain with the interval of the highest numerology we want to use.

As means of simplifying implementation, the same time interval is used for every frequency band even if we want to simulate different numerologies in each band. The time interval used is correspondent to the highest numerology across bands.

A.1.2 Blocking Model

The blocking model applies changes to the channel in time domain such that the received signal is attenuated by a certain amount that should be realistic with the blocking case.

Further note the blocking modifications to the channel are done in Time Domain. As opposed to the frequency channel, where the paths and path delays have used to compute the frequency response and are no longer available, in time domain the per path information is still available, and this individual path information is required as it contains the angles of arrival and departure of each path.

A contribution from a TNO's intern named Sandra. An overview of the model follows.

Overview of the blocking model here.