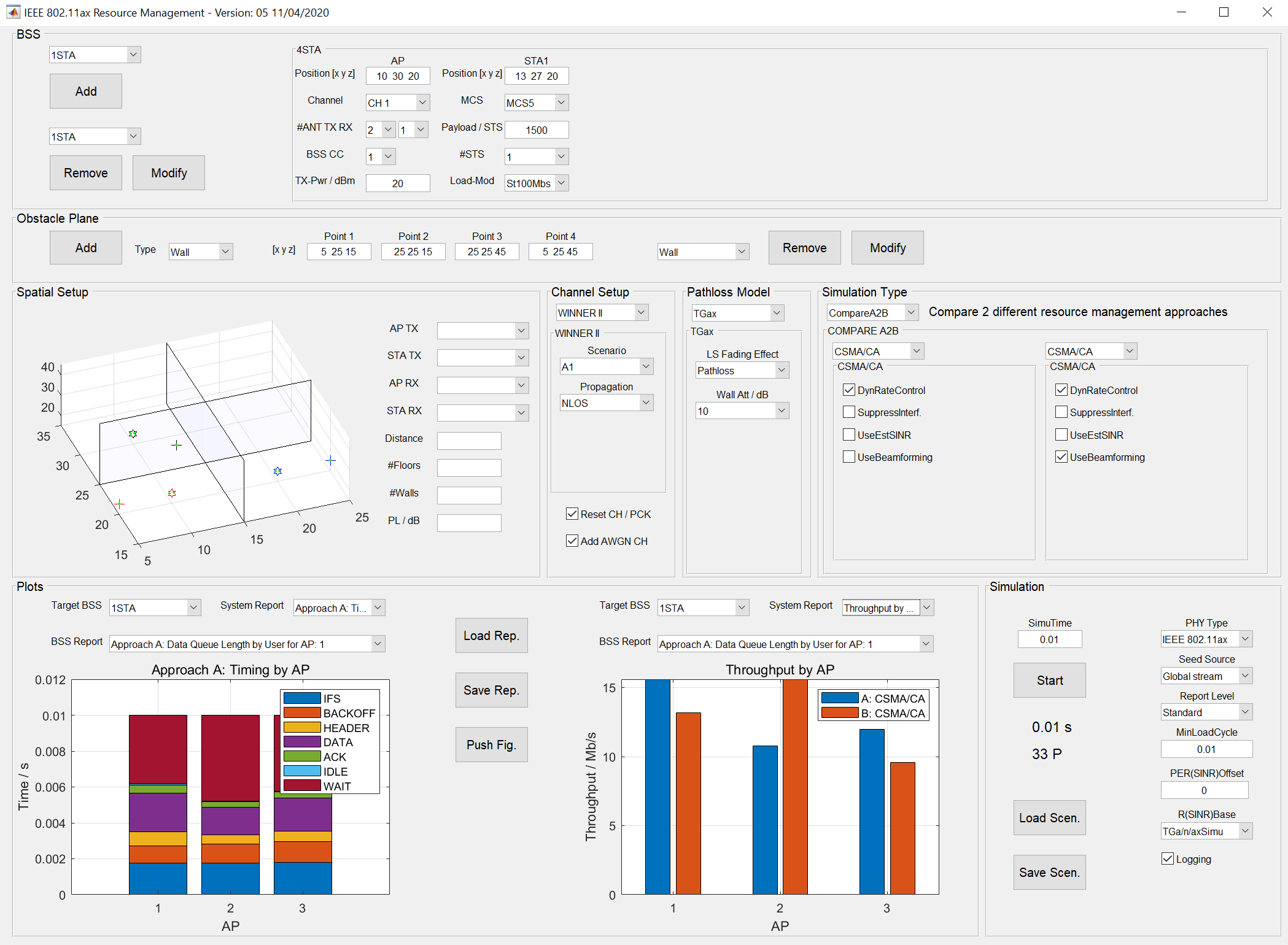
IEEE 802.11 Resource Management Simulation Platform



Version: 6, 03.01.2022

# Introduction

The IEEE 802.11ax resource management simulation platform is based on Matlab WLAN-Toolbox 2.0 (R2018b) PHY to simulate downlink WLAN packet traffic for different scenarios and parameter variations.

A discrete event simulation model combined with a state machine design is used to implement a CSMA/CA MAC layer and other medium access and resource management methodologies.

Current key features are:

* Free BSS and STA scenario definition with unlimited spatial placement of stations.
* Full implementation of TGax and WINNER II fading channel model and AWGN noise channel.
* IEEE 802.11ax PHY simulation down to bit level with MCS 0 – MCS 11, MIMO, multiple STS, all with and without bit level interference.
* IEEE 802.11n/a PHY for reference with MCS 0 – MCS 7.
* Multiple channels (1, 6, 11) supported for CSMA/CA.
* TGax pathloss model or free configurable eta power law pathloss model.
* CSMA/CA MAC with DIFS/EDIFS, SRC, variable BACKOFF, CCA, ACK & RESEND etc. along IEEE 802.11a.
* Flexible data load model with stream 1 – 100 Mbit/s or packets 1 – 10 Mbit/s or random packets by station.
* A2B comparison mode for algorithms under investigation or CSMA/CA.
* Loop Over mode for Payload Length, MCS, TP, OBSS\_PD, SNR to evaluate impact of changing parameters for CSMA/CA.
* Reporting on PER, throughput, queue status, SINR, MCS selection and others.
* Flexible design to implement other parameters and centralized approaches.

Warning:

Using the platform requires some knowledge about WLAN, channel models and the IEEE 802.11 standard. The platform is not at all protected against entry of wrong parameters which may lead to an exception of the Matlab program or unexpected results. Also an entry of valid parameters but wrong combination may lead to errors due to a mismatch with the IEEE 802.11 standard. It is good practice to monitor the command window of Matlab for any error messages.

# Usage

The flow to run simulations is defining BSS and obstacles, selecting and running a simulation scenario and evaluating results. Many parameters to enter are directly coupled to the PHY layer of the high level functions of the MATLAB WLAN-Toolbox. Entering wrong or non-standard conform parameters will normally result in MATLAB exceptions, which are visible in the command window of MATLAB.

To start the simulation the script in file ‘start\_here\_platform\_GUI.m’ should be started with ‘Run’ in MATLAB editor mode.

Please note that not all simulation types support all features. For reference figure 1 provides an overview. More details are discussed in later sections.

|  |  |  |  |
| --- | --- | --- | --- |
|  | CSMA/CA | CSMA/SR | CSMA/SDMSR |
| IEEE802.11ax/n/a PHY | x |  |  |
| TGax fading channel | x | x | x |
| WINNER II fading channel | x | x | x |
| TGax / Eta PowerLaw pathloss model | x | x | x |
| AWGN channel | x | x | x |
| Loop simulation | x |  |  |
| A2B simulation | x | x | x |
| Multi STA support | x |  |  |
| MIMO support | x |  |  |
| Diversity support | x |  |  |
| Multi-channel support | x |  |  |
| Beamforming (only WINNER II and IEEE802.11ax PHY) | x |  |  |

Figure 1: Supported features by simulation type

## Defining BSS

Figure 2 shows the setup for BSSs. The choice is to define BSS with one, two, four or nine STAs. The current limit is to acknowledge limitations for future implementations. IEEE 802.11ax defines OFDMA up to nine stations only.

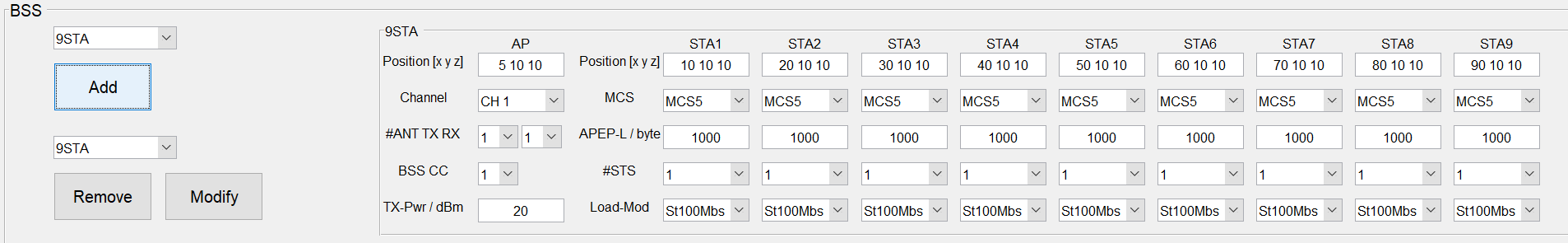


Figure 2: BSS section of GUI

With the top left pulldown the BSS type can be selected. Next step is to define the spatial setup of the AP and each STA, the MCS and payload length, the number of space time streams (STS) and the load model for each station. MCS and payload length are used as default values, except the selected simulation type will overwrite these values.

The position of each station as x, y, z coordinates in meter is free to define, but all entered spatial values in a simulation should not directly overlap. However, very small differences are fine enabling simulating very close stations.

The load model allows defining either a stream of data for 1, 5, 10, 100 Mbit/s or packets with a net rate of 1, 5, 10 Mbit/s. or random length and time packets. For stream of data, the queue load cycles may be adjusted (see ‘Simulation’ section), for packets, the load cycle is fixed to 100 ms. Details are given in the function DataLoadGetTimeSize in the sim\_A2B.m or sim\_Loop.m files. It should be realized, that very short simulation times will lead to singular loads of the data queues. It is good practice to start with a saturated load model (100 Mbit/s) to evaluate the behaviour of a network. For latency measurements there are four load models for packets resulting in a data stream. For time to first byte (TTFB) 80 b packets lead to a stream of 0.1 Mb/s. For time to deliver (TTD) 8 kb packets lead to a stream of 1 / 10 / 100 Mb/s.

The number of space time streams is currently one to one coupled to number of transmit and receive antennas as explained before. All STAs share the same number of receive antennas within one BSS.

For the AP, the default transmit power should be selected in dBm. Again, this value may be overwritten according to the selected simulation type.

For CSMA/CA, available channels for a BSS can be selected from non-overlapping channels 1, 6, 11. All other simulation types need to have all BSS on one common channel.

The mentioned BSS colour code (BSS CC) is currently not used.

After choosing the type of BSS (number of STAs) and changing any parameters, ‘Add’ will add the BSS to the simulation. The defined BSS will enumerate in the pulldown menu below the ‘Add’ button and will be shown in the spatial setup figure.

More BSS may be added to the simulation as described. There is no limit to the number of stations. However, it is good practice to start with very few BSS and STAs as simulation time is related to the number of BSS and STAs.

BSS may be removed from the simulation by selecting the BSS in the pulldown below the ‘Add’ button and pressing the ‘Remove’ button. A selection flashes the corresponding BSS members in the spatial setup view to make selection easier.

BSS parameters may be changed by reloading the parameters. The pulldown below the ‘Add’ button allows selection of the BSS. The selection will load the parameters into the corresponding template. After changing some parameters the ‘Modify’ button saves the modifications into the simulation.

If a IEEE 802.11n/a PHY is used there are limitations in using MCS and number of antennas. Using not allowed values generates a warning or will lead to exceptions of the program, visible in the command window.

## Defining Obstacles

The simulation allows to define obstacle planes, which will affect the pathloss. Figure 3 shows the part of the GUI. Two types of obstacles are available: Walls and floors.

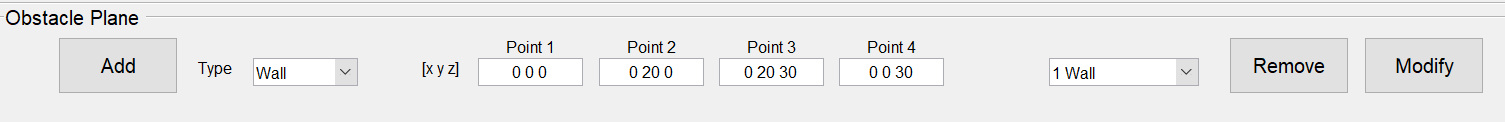


Figure 3: Obstacle section of GUI

The methodology to add, modify or remove an obstacle is pretty much the same as with BSS. However, each obstacle is defined as a section of a plane with 4 coordinates for the rectangle. The coordinates should be chosen to define a rectangle clockwise or counter clockwise.

## Spatial Setup View

The Spatial Setup view visualizes the defined BSSs and obstacles and allows querying a pathloss between any kind of STA or AP. Figure 4 shows an example of 6 BSS with multiple wall obstacles and a selected pathloss.

Hovering over the plot with the mouse pointer shows up a tool menu. Usual MATLAB features like rotating the plot are available.

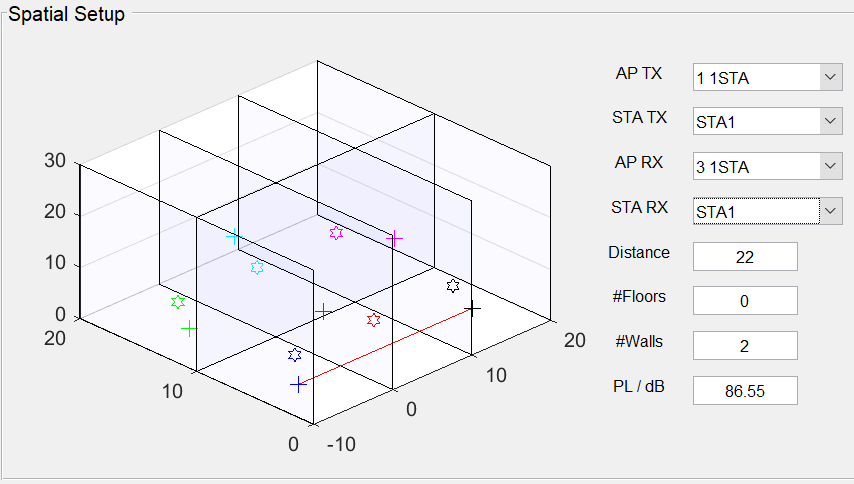


Figure 4: Spatial Setup section of GUI

It is possible to show details of a path between any network element. The pulldown menus should be used to select the path via a ‘From AP / BSS’ and ‘From STA within BSS’ to ‘To AP / BSS’ to ‘To STA within BSS’ logic. Any combination may be used to show pathlosses between AP to AP, AP to STA or STA to STA etc. The distance as well as the number of penetrated floors and walls and resulting pathloss do show up. The pathloss is calculated as selected, either along IEEE 802.11ax standard or along eta-Power-Law.

If the WINNER II channel mode is used, antennas are modelled as uniform linear arrays (ULA) oriented along the x-axis. Green dots with enhanced spacing will show the orientation of the ULAs in the Spatial Setup view.

## Channel Setup Section

The channel setup section allows to define the fading and noise channel to be used during the simulation.

The first pulldown menu allows to select the fading channel model ‘TGax’, ‘WINNER II’ or ‘None’.

For ‘TGax’ ‘Delay Profile’ enables selection of the channel delay profile as defined in the TGax documents. ‘Model-A’ is frequency flat fading and ‘Model-B’ is commonly used for indoor simulations.

For ‘WINNER II’ ‘Scenario’ enables selection of the scenario model as defined in the WINNER II documents. Currently only A1 is implemented which reflects indoor simulations. ‘Propagation’ is used to select LOS or NLOS propagation. In contrast to TGax which includes a switch between LOS and NLOS based on distance the WINNER II model uses a general LOS vs. NLOS switch. In fact, using LOS switches of scattering paths by geometric oriented reflection clusters. For MIMO simulation this switch is quiet important. For indoor simulations the recommendation is to stay with NLOS.

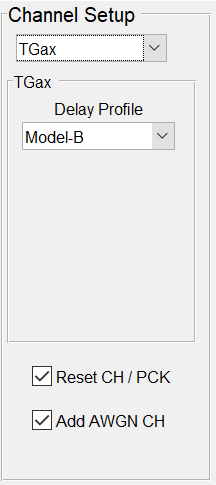
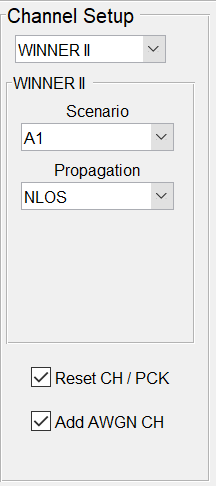
 …

Figure 5: Channel Setup section of GUI

The checkbox ‘Reset CH/PCK’ (reset channel per packet) is closely related to the ‘Seed Source’ selection discussed in the simulation section. With ‘Reset CH/PCK’ selected, the fading channel model (and the AWGN channel model) will be reset to the starting state for every packet transmission. The combination ‘Global Stream’ and ‘Reset CH/PCK will lead to independent random realizations of the process for every usage. For the fading channel it means jumping with random steps along the changing transfer function of the channel. Even with few realizations, many different versions of a channel state will flow into the simulation.

If ‘Global Stream’ and ‘Reset CH/PCK’ deselected is combined, for the fading channel, the simulation will walk step by step along the evolving channel transfer function (but random start of the transfer function). This may be helpful in evaluating adaptive algorithms based on changes in time.

‘Global Stream’ and ‘Reset CH/PCK’ selected is the normal way of simulation. Experience shows that simulations with 10000 packets and more do normally clearly converge towards expected statistical behaviours of the network. However, this may differ depending on the size of a defined network and other parameters.

## Pathloss Model Section

The pathloss model section allows selecting one of two implemented pathloss models and defining corresponding parameters.

### TGax Model

‘Large Scale Fading Effect’ details the TGax pathloss model of the simulation, again as defined in the standard. It should be noted that even with ‘None’ selected there is still a fading component of the channel which shows up as statistical phase and amplitude variations of the channel transfer function based on Reighley fading. With delay profiles selected other than ‘Model A’ there is also a frequency dependent changing component. ‘Pathloss’ and in addition ‘Shadowing’ adds a pathloss and a fixed attenuation to this model. There is also a third component, the Doppler change, which results in periodic transfer function changes with a cycle of 10 ms – 100 ms. As discussed before, that component is switched off.

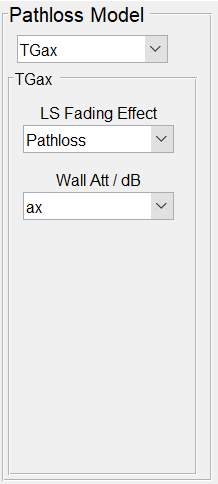
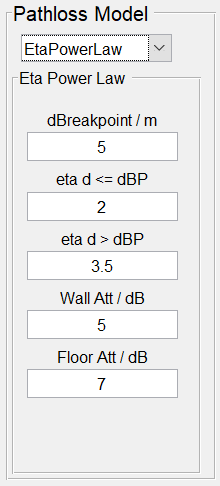
 

Figure 6: Pathloss Setup section of GUI

‘Wall Att / dB’ allows to change the pathloss attenuation of walls to values other than defined in the TGax documents. This is helpful to evaluate the effect of a higher spatial isolation between BSSs.

### Eta-Power-Law Model

This model uses free space attenuation with 2 different exponents, separated at a breakpoint distance.

The first three parameters ‘dBreakpoint / m’, ‘eta d <= dBP’ and ‘eta d > dBP’ define both exponents and the breakpoint. It is common to stay with an exponent of two for the first exponent and just vary the second one along with breakpoint variations. If compared to the TGax model, it should be realized that the breakpoint distance in the TGax model differs by selected delay profile.

‘Wall Att / dB’ and ‘Floor Att / dB’ define the wall and floor attenuation. In contrast to the TGax model, the pathloss of multiple walls or floors are simply added up.

## Simulation Type Section

The simulation type section of the GUI defines which simulation should be run and what parameters should be used. The type of simulation is selected top left with a pulldown menu. Currently ‘CompareA2B’ and ‘LoopOver’ is implemented.

### CompareA2B

Figure 7 shows an example of the simulation setup of selected A2B types. The goal is to do a one to one comparison of two different algorithms. It is also possible to compare just one algorithm but using different parameter settings.

#### CSMA/CA

CSMA/CA may serve as a reference to be compared to other algorithms.

The checkbox ‘DynRateControl’ enables a dynamic selection of MCS based on the available SINR. The SINR is calculated as a moving average over multiple transmissions. The clipping levels are defined in the function SNR2RMCS in the WorkOnEvent\_CSMA\_CA.m file. If the checkbox is deselected, the MCS from the BSS setup is used for all transmissions.

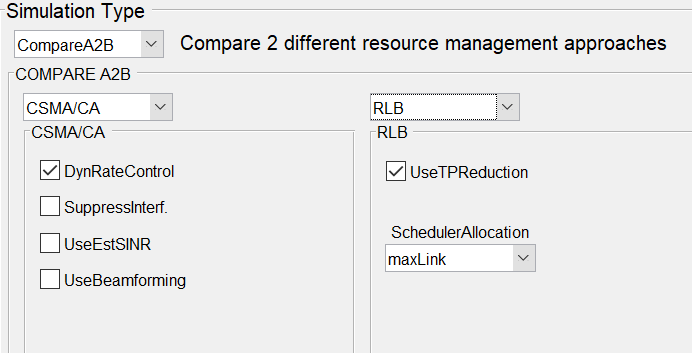


Figure 7: A2B Simulation Setup section of GUI for CSMA/CA and RLB

‘UseEstSINR’ selects SINR from packet detect as a source for MCS decision. If unselected, calculated SINR is used. There is a difference between both methods about 2 dB – 3 dB (est. SINR lower). This is due to introduction of additional noise during noise estimation of packet receive.

The ‘SuppressInterf.’ checkbox allows to run the simulation without any interference between the BSSs. This is helpful to evaluate the effect of interference on network performance.

‘UseBeamforming’ may be used together with the WINNER II channel model and an IEEE 802.11ax-PHY. It implements an ideal singular value decomposition (svd) based beamforming approach. For every packet, the channel between transmitter and receiver is sounded by a NDP-packet. The resulting channel state information (CSI) is used to calculate a steering matrix to be used for the following data packet. The seed mechanism of the channel model is implemented in such a way that NDP and data packet use the same channel realization.

#### CSMA/SR

CSMA/SR is a spatial reuse (SR) channel access protocol which includes parallel tranmissions of links due to optimal TP and MCS resource settings per link. A brute force MCS driven approach (MCSDA) is used to arrive at TP and MCS settings.

For more information refer to presentations of ‘Netzwerkrunde’ at …\info\interne\_berichte\Netze.

#### CSMA/SDMSR

CSMA/SDMSR combines the approaches CSMA/SDM and CSMA/SR towards spatial reuse with spatial division multiplexing using MCSDA equal allocation methodologies but also for cross link SDM clusters. The selection of allocation sets is quiet flexible based on link and antenna scenarios. Calculation and simulation may take significant time. The recommendation is to start with #TXAnt/#RXAnt/#STS 1/2/1 or 2/2/2 configurations with very few links.

For more information refer to presentations of ‘Netzwerkrunde’ at …\info\interne\_berichte\Netze.

### LoopOver

The ‘LoopOver’ simulation type allows to evaluate the impact of single parameter changes to network performance.

Figure 10 shows an example. Currently there are 5 parameters defined which may be evaluated. The payload length, the MCS, the transmit power, the spatial reuse level OBSS\_PD and SNR. For each parameter a start, step, stop set of values can be defined. However, just one parameter is evaluated as selected by the pulldown menu. The other parameters are taken from the BSS definition unless overwritten by a selected algorithm. For OBSS\_PD the default value defined by the standard is -82 dBm.

The ‘SNR’ simulation supports single link simulations and cancels out any pathloss between transmitter and receiver. Instead, noise is increased in such a way that the corresponding SNR-value is achieved. This simulation type is helpful to show PER performance by SNR for specific configurations.

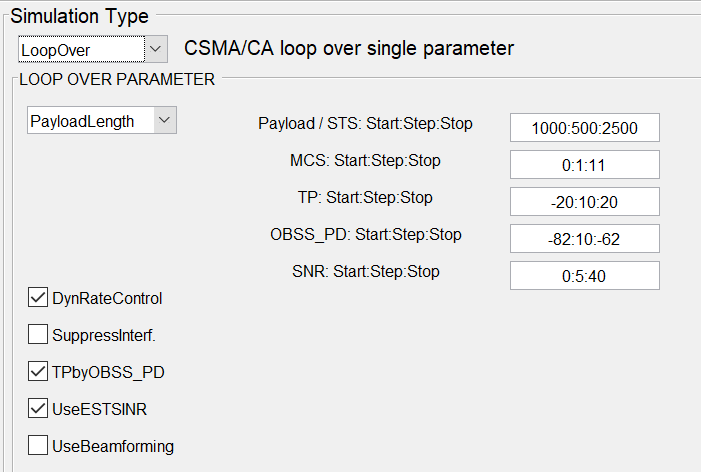


Figure 10: LoopOver Simulation Setup section of GUI

‘DynRateControl’ overwrites the MCS from the BSS setup and selects MCSs based on given SINR as described before. This affects all simulation subtypes, except the MCS one.

‘TPbyOBSS\_PD’ implements the proportional invers transmit power adjustment along OBSS\_PD increase as required by the standard. This parameter only affects a ‘OBSS\_PDLevel’ simulation.

‘UseEstSINR’ selects SINR from packet detect as a source for MCS decision. If unselected, calculated SINR is used. There is a difference between both methods about 2 dB – 3 dB (est. SINR lower). This is due to introduction of additional noise during noise estimation of packet receive.

‘UseBeamforming’ establishes a svd-based beamforming as discussed in a previous section.

## Simulation Section

In the simulation section a simulation may be started by pressing the ‘Start’ button. The progress of simulated time and number of sent packets may be monitored below the ‘Start’ button.

The ‘Start’ button changes to a ‘Cancel’ button to stop simulation. Entering ‘CTRL-C’ in MATLAB command window has the same effect.

It is also possible to save and load whole BSS, obstacle and simulation setups with the save and load scenario buttons. The platform folder does have a ‘scenario’ folder but it is possible to load and save scenarios from any path. The procedure follows the usual Windows style.

‘SimuTime’ defines the overall simulated time in seconds. It is good practice to start with 10 ms and check the general setup and flow. Common times of simulations used in publications are in the area of ten seconds, which will take quite a long time to simulate. For example, simulating four BSS with two stations each A2B for CSMA/CA for five seconds simulated time leads to 7 h simulation time resulting in 35000 packets sent.

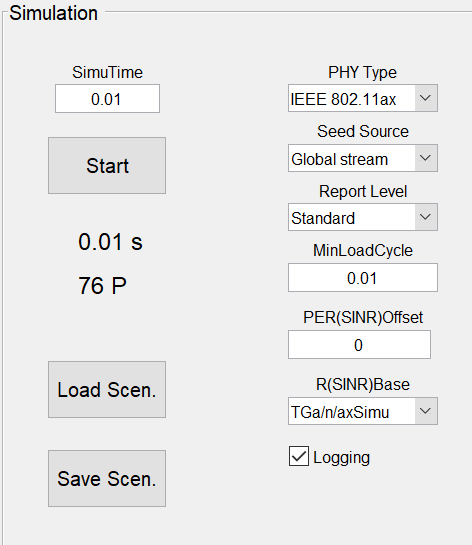


Figure 11: Simulation Setup section of GUI

‘PHY Type’ selects the to be used PHY of the simulation. IEEE 802.11ax/n/a is implemented, which will result in different packet formats and a different receive processing. As discussed before, there are limitations for the different PHYs in terms of allowed parameters. In case of errors, first step is checking the allowed range in the standard.

‘Seed Source’ defines the source for random processes within the simulation. Random processes used are random payload generation, random allocation factors within the AWGN channel and the TGAX fading channel. Using ‘Global Stream’ will result in random starting states for all these processes. This means that a second simulation will be different vs. a first simulation, even with unchanged parameters. Using ‘mt19937 with seed’ will start all processes with a defined state. Running two simulations should show exactly the same results.

‘Report Level’ defines what kind of plots are generated. The mapping to Basic, Standard or Extended (B/S/E) is done in the next section. A higher Level does always include the lower levels.

‘MinLoadCycle’ defines the cycle time in seconds for loading the data payload queues. This effects all stream load modes only and results in different bytes by cycle to be added to the data queues.

‘PER(SINR)Offset’ in dB affects all algorithms and all simulations. The MCS selection is based on MCS(SINR) curves from simulation at a PER of 0.1 to 0.01 or by calculation from minimum receiver sensitivity from the IEEE802.11 standard. Moving to very low signal energies close to minimum receiver sensitivity as defined in the standard requires an additional link budget margin. This margin may be addressed by a shift of the PER(SINR) curves. A value of three to six dB proofed to be a good compromise not to run into too many packet errors.

‘R(SINR)Base’ selects the source of SNR clipping levels. Either the clipping levels come from simulations ‘TGax/n/aSimu’ or are derived from the minimum receiver sensitivity levels defined in the standard ‘IEEE802.11’. All levels are grouped and defined at the beginning of the simulation functions ‘simA2B.m’ and ‘simLoop.m’.

‘Logging’ turns on or off the logging function as described earlier.

## Evaluating Plots

After the end of a selected simulation, plots are generated and enumerated in the pulldown menus of the plot section of the GUI (figure 12). The kind of plots do depend on the selected simulation and report level. For multiple BSSs, the plot generation may take some seconds.

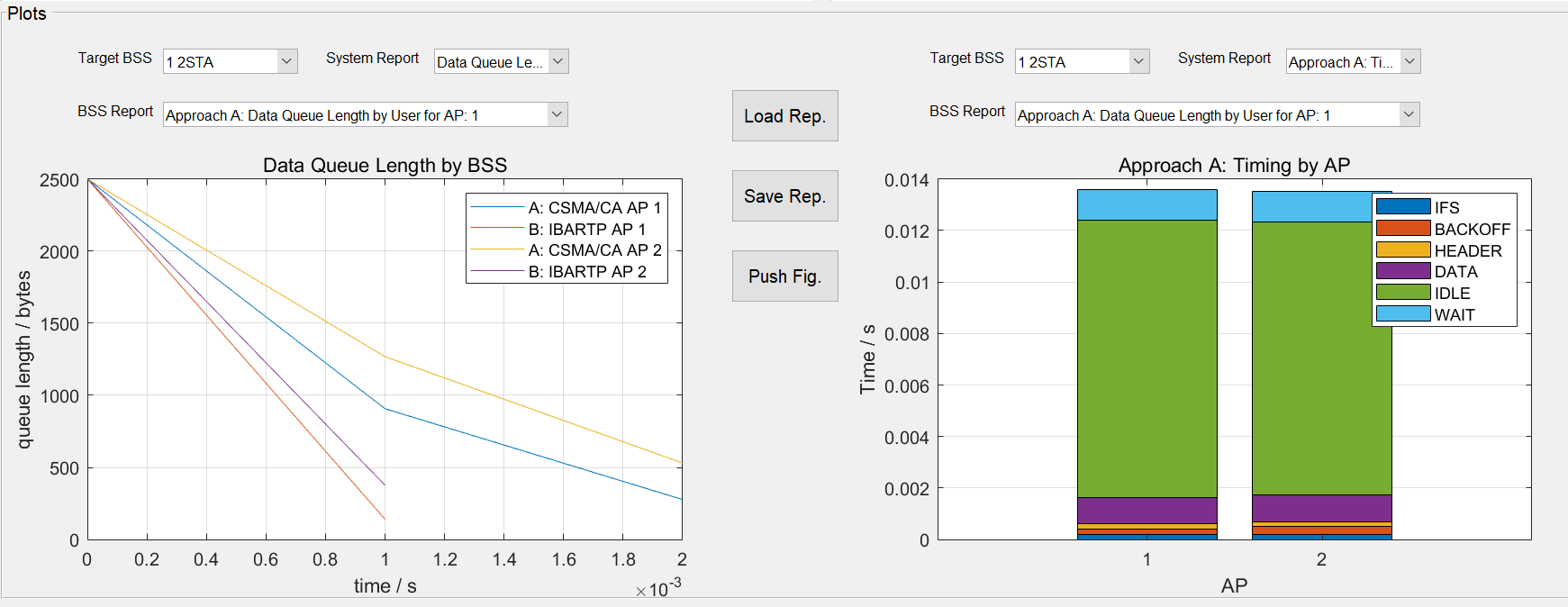


Figure 12: Plots section of GUI

Plots may be selected either on a system or AP level with ‘System Report’ or may be selected by BSS as STA / User reports. First step is to select a target BSS with pulldown ‘Target BSS’ and second step is to select an available report for that BSS with ‘BSS report’.

The left and right plot areas have the same functionality. The two areas allow to compare different plots one to one. Both plot areas have MATLAB toolbar menus (hovering over with mouse pointer) to rotate plots etc.

There is a save and load functionality for a complete set of plots. The platform folder does have a ‘reports’ folder but it is possible to load and save plot sets from any path. The procedure follows the usual Windows style.

The ‘Push Fig.’ button generates singular figures for every plot. This helps to better change and copy plots into other documentation.

Some of the plots do use a histogram function from MATLAB. This function requires a minimum set of data to plot meaningful results. If this minimum set of data is not generated during simulation the plot will be empty and a message is generated in the command window of MATLAB.

The plots are pretty much self-explaining. A short overview of current reports does follow:

### A2B – System Reports

B = Basic, S = Standard, E = Extended Report

S:‘Data Queue Length by BSS’: Shows the data queue length by AP for simulation A and B.

B:‘Packet Error Rate by AP: Shows the packet error rate by AP for simulation A and B.

B:‘Throughput by AP’: Shows the throughput by AP for simulation A and B. Throughput is defined as MPDU length (PPDU payload) in bytes.

B:’Latency by AP’: Shows the average latency between data queue loaded and successful reception at the receiver in seconds per system. This report only shows up, if the latency load generation load models are used.

S:‘Approach A/B Timing by AP’: Shows the time spent of the AP state machine in different states.

B:‘Packet Error Rate for System: Shows the packet error rate for the total system for simulation A and B.

B:‘Throughput for System’: Shows the throughput for the total system for simulation A and B. Throughput is defined as MPDU length (PPDU payload) in bytes.

B:’Latency for System’: Shows the average latency between data queue loaded and successful reception at the receiver in seconds per system. This report only shows up if the latency load generation load models are used.

E: ‘Jain’s Fairness Index’: Shows the fairness index for simulation A and B. For more details, please refer to literature.

### A2B – BSS Reports

B = Basic, S = Standard, E = Extended Report

S:‘Approach A/B Data Queue Length by User for AP n’: Shows the data queue length by user for simulation A or B and AP n.

B:‘Approach A/B Packet Error Rate by User for AP n’: Shows the packet error rate by user for simulation A or B and AP n.

B:‘Approach A/B Throughput by User for AP n’: Shows the throughput by user for simulation A or B and AP n.

S:‘Approach A/B MCS Usage by User for AP n’: Shows the MCS usage by user for simulation A or B and AP n.

E:‘Approach A/B Estimated SINR by User for AP n’: Shows the estimated SINR by user for simulation A or B and AP n. Estimated SINR is taken from channel estimation during receive of packet.

S:‘Approach A/B Calculated SINR by User for AP n’: Shows the calculated SINR by user for simulation A or B and AP n. Calculated SINR is the calculated ratio of signal power after pathloss to all interferer power after pathloss plus noise.

### LoopOver – System Reports

B:‘Packet Error Rate by *Parameter*: Shows the packet error rate by a selected parameter of loop by BSS.

B:‘Throughput by *Parameter*: Shows the throughput by a selected parameter of loop by BSS.

B:‘Packet Error Rate System by *Parameter*: Shows the packet error rate of the total system by a selected parameter of loop by BSS.

B:‘Throughput System by *Parameter*: Shows the throughput of the total system by a selected parameter of loop by BSS.

### LoopOver – BSS Reports

B = Basic, S = Standard, E = Extended Report

B:‘Packet Error Rate by *Parameter* for AP n: Shows the packet error rate by a selected parameter of loop by station for AP n.

B:‘Throughput by *Parameter* for AP n: Shows the throughput by a selected parameter of loop by station for AP n.

S:‘Timing by *Parameter* for AP n: Shows the time spent of the AP state machine in different states.

S:‘MCS Usage by *Parameter* for AP n: Shows the MCS Usage by a selected parameter of loop by station for AP n.

E:‘Estimated SINR by *Parameter* for AP n: Shows the Estimated SINR by a selected parameter of loop by station for AP n.

S:‘Calculated SINR by *Parameter* for AP n: Shows the Calculated SINR by a selected parameter of loop by station for AP n.

# Simulation Model

The simulation platform simulates transmission to reception of WLAN packets according to the high efficiency (HE) IEEE 802.11ax standard through a fading channel. The current implementation is limited to single user (SU) packet formats. For reference, the IEEE 802.11 n/a PHY with corresponding packet format may be selected for simulation.

Figure 13 shows the implemented general signal chain. MATLAB WLAN-Toolbox, as a PHY level simulator, allows a simulation down to a bit level at complex baseband (see figure 14).

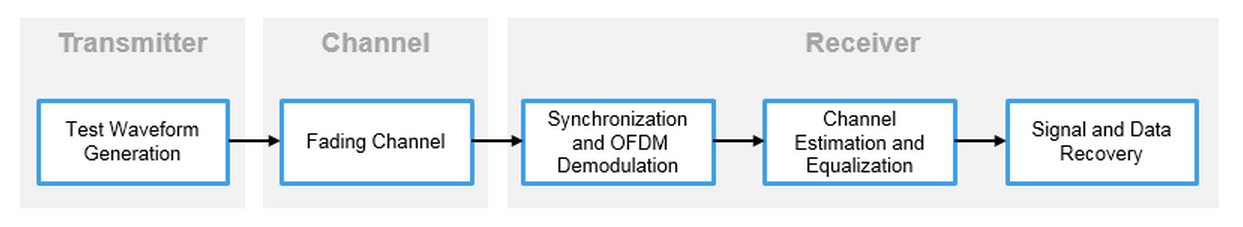


Figure 13: Signal chain.

A packet header (PHY preamble) is combined with a MAC payload (MPDU) to build a WLAN packet (PPDU). This packet is a modulated and multiplexed (MCS / OFDM) complex baseband signal including FEC BCC/LDPC. The current implementation uses a bandwidth of 20 MHz, which maps to a sample frequency of 20 MHz. All used parameters are configured for a simulation in the 2.4 GHz band. However, these parameters can be changed easily.

The packet passes a fading channel which is implemented as a clustered tapped FIR filter as defined in the IEEE 802.11 task group or WINNER II documents. Different delay profiles, Doppler-change, pathloss and other factors define how the signal is affected.

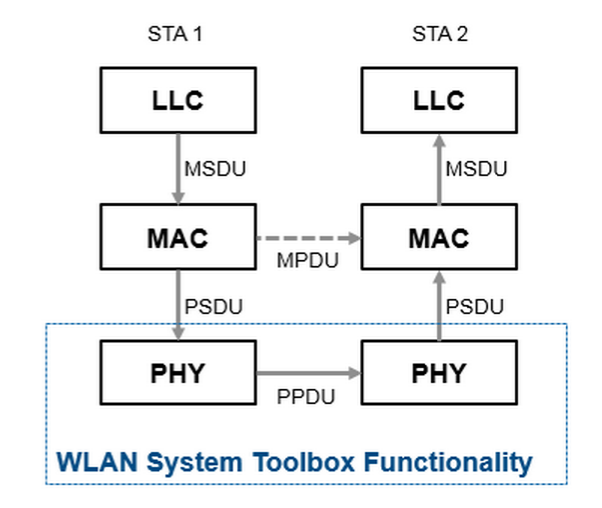
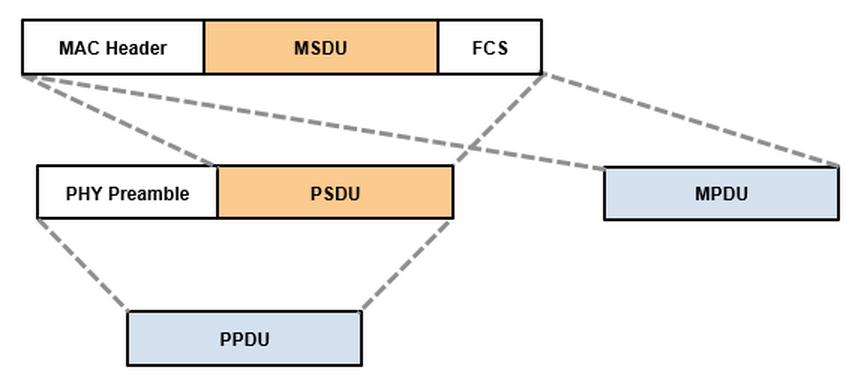
 

Figure 14: MATLAB WLAN-Toolbox packet implementation.

In the receiver, the baseband signal is demodulated and the MAC payload is recovered. Packet and frequency offset correction as well as channel estimation is used for OFDM symbol equalization.

For all simulations, the comparison of the recovered payload to the sent payload defines the presence of a packet error. This payload (MPDU) is also the base for all throughput calculations. The channel estimation also includes a noise estimation which allows to show a SNR (SINR) for a received packet.

Interference between different packets is implemented as superposition of complex baseband signals as shown in figure 15.

As discussed, a packet passes a fading channel with pathloss to reach the target station. Noise is added before receive processing. The same packet passes a different realization of the fading channel towards another station. As this second station is located elsewhere, the pathloss is different. At each receiver (station) all signals are overlait to build the to be decoded signal at complex baseband.

The standard (IEEE 802.11ax) allows to use up to eight transmit and receive antennas. This is reflected in WLAN-Toolbox with up to eight baseband signal streams. These streams may be used for same, correlated or uncorrelated data which means implementing diversity to improve SNR or space time streams to improve communication capacity (or a mix of it).

Using a TGax-fading-channel the current implementation is limited to a one to one direct mapping of transmit signals to transmit antennas. This leads to a simple rule that number of transmit antennas is equal to number of receive antennas is equal to number of space time streams. Also transmit power is equally spread to the number of transmit antennas.

Using a WINNER II geometric fading channel model provides more freedom in selection of number of transmit antennas, receive antennas and space time streams. Besides the direct mapping the following numTX/numSTS combinations are supported: 1/1, 2/1, 4/1, 8/1, 2/2, 4/2, 8/2, 4/4, 8/4. Any numRX can be used. However, the numRX has to be greater or equal to numSTS. The WINNER II channel model also allows to implement different antenna type configurations and orientations. For simplicity, the current implementation always uses uniform linear arrays, oriented along the x-axis.

For the WINNER II channel model, svd-based beamforming is implemented. The up to eight baseband signal streams at the transmitter are modified by a steering matrix. Therefore the steering matrix influences both the target transmission but also any interference transmission. The steering matrix is calculated based on channel sounding by a NDP-packet per data packet. The same channel model realization is used for NDP- and data-packet, by using temporarily the same channel seed.

More complex implementations like space time block coding (STBC) etc. are topics for future work.

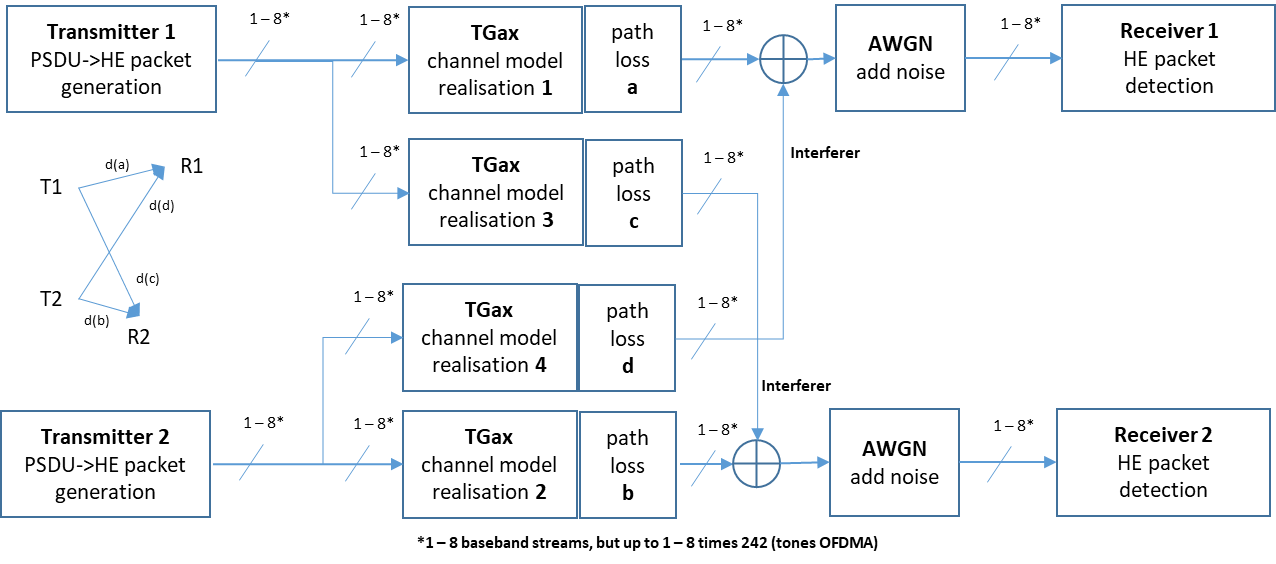


Figure 15: Interference model.

More information is available at MATLAB WLAN-Toolbox help with a good cross reference to documents from IEEE 802.11 standard and research.

# Discrete Event Simulation

The platform is a discrete event simulator along time. In contrast to other simulation concepts, simulation work is only done at discrete times when the state of the overall system changes. For example, if a packet is transmitted, the first sample at complex baseband is at time one, the last sample at time two. There is no need to simulate every single sample. The whole receive processing can be done at time two. The discrete event simulator therefore just takes 2 simulation steps, at time one to transmit and jumping to time two to do the receive processing. This saves a lot of time and simulation resources.

The current platform uses the following key components of a discrete event simulator:

## Resources

Every station has its own receive channel as an array of complex baseband samples (variables CHSTA and CHAP). For the receiving stations, this is the point of merit for all traffic and interference. Each access point needs a channel as well for clear channel assessment (CCA).

Each access point holds an array for user data by station (variable DATAQ). Currently, only the length of data is covered, as all payload is random. There is also a pending data queue by station (variable DATAPENQ) to represent the traffic ‘on air’ during transmit.

There are a couple of counters to cover station retry counts etc. (variables APCNT, APSTACNT, SCHEDCNT and more).

The channels of every station will become large arrays over simulation time. This will significantly degrade the program performance. To avoid such issues a ‘flying channel concept’ is implemented. A garbage collector shortens the channel arrays periodically to the needed amount based on maximal transmission time. To stay consistent, the arrays are addressed with an offset (variables OSCHAP, OSCHSTA). Therefore, all time and sample indexes operate on the absolute time and sample since start of the simulation.

An uplink channel concept is implemented for ACKs. As complex baseband signals, the ACK are transmitted with a separate TGax channel realization towards APs and other STAs.

## Events

DataOnQueue: Fills the data queues of the station according to the selected load model.

ChannelGarbageCollection: Shortens the channel arrays as explained above.

APCycle: Any event by AP; depends on content of event, the selected simulation and state of the AP state machine. For CSMA/CA and other methodologies, details are discussed later.

## Global Event Queue

The global event queue (table variable GEQ) is the anchor of the simulation. Essentially, the simulation does some work by event along the event queue. Depending on the result of an event, new events may enter. Figure 16 shows an GEQ example for a simulation of a two BSS with two STA each scenario, just after start up.

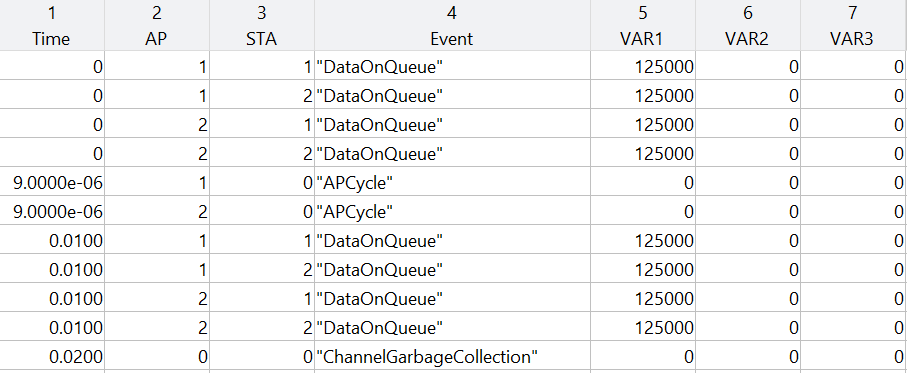


Figure 16: Global Event Queue example

Data is allocated for all 4 stations, each 125 kByte. This is repeated at 0.01 s along the selected load model. In between, two AP cycles start the CSMA/CA protocol for each AP. At 0.02 s the garbage collector shortens the channel arrays. The minimum discrete time interval between events is either zero or a SLOTTIME (here 9μs).

## Logging

To evaluate the system behaviour in detail and to better understand the different algorithms, a logging is included (folder logs). The logs are LogApproachA.txt and LogApproachB.txt for A2B comparisons or LogApproachn.txt for each of the n loop simulations in loop mode.

## Statistics

During simulation, data is collected (variable statsA, statsB for A2B comparison and statsPAR for loop mode). The content is used to generate plots to show throughput, PER etc.

# CSMA/CA Implementation

A basic CSMA/CA is implemented along IEEE 802.11a to serve as a reference for all simulations. The DCF functionality is implemented with DIFS (which includes acknowledge time), EDIFS in case of previous error, station retry count (SRT) with resulting variable, random BACKOFF time, ACK and simulated RESEND in case of error, as the data will be put back to the data queues.

## State Machine

The CSMA/CA state machine is implemented as APCycle events. The normal flow of states is from IDLE to CCA to IFS to BACKOFF to TRANSMIT to RECEIVE to ACK to IDLE. The standard requires CCA / IFS / BO to be implemented on a SLOTTIME base, as channel free needs to be checked every SLOTTIME. This slows down the simulation to some extent. The following states for each AP do apply:

APIdle: Checks for data on the data queues. If data is present, it checks for a free channel. If the channel is free, it starts the APIFS state with an IFS counter, if not free, it stays in CCA mode with APCCA.

APCCA: Checks if the channel is free. On a busy channel it stays in APCCA. On a free channel it moves to APIFS with an IFS counter.

APIFS: Checks for a free channel. If busy, it moves back to APCCA state. If free, it decrements the IFS counter. If the IFS counter is zero, it moves to the APBO state, allocating a backoff counter based on last retry counts and standard parameters.

APBO: Checks for a free channel. If busy, stays in APBO, if free, decrements the backoff counter. If the backoff counter is zero, it moves to APTX state.

APTX: Selects data from a station round robin, puts the data on the pending queue, transmits the packet. Moves the state machine to APRX state.

APRX: Receives the packet. If the transmission is successful, ACK state is entered. In case of error, the data from the pending queue moves back to the data queue for retransmission. The state machine moves to IDLE.

ACK: An uplink ACK is generated to confirm successful packet reception. After ACK, the state machine enters IDLE for a new cycle.

During all states detailed timings are recorded. This allows to evaluate the simulation time by state in detail for each AP.

## Example

Figure 17 shows a simulation for two BSS with 2 STA each using CSMA/CA:

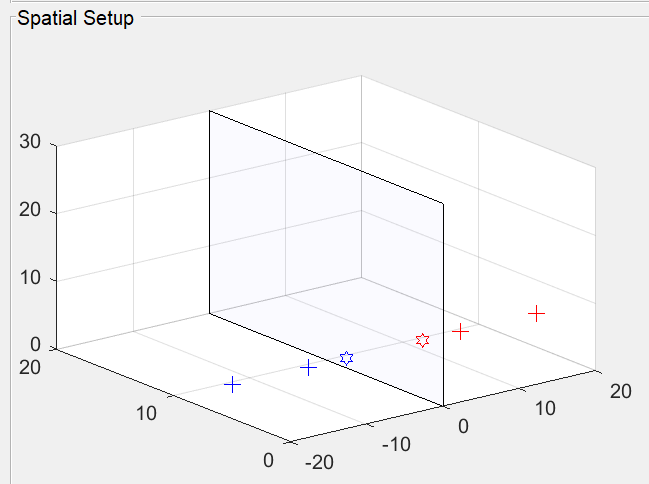


Figure 17: Spatial setup for CSMA/CA example.

The log (figure 18) shows how the CSMA/CA implementation is working. Each record has a time stamp for the actual date and time, followed by the simulation time idxT and the simulation sample idxS. The AP state and other information follows, if applicable. Next are recorded times for IFS, BO, HEADER, DATA, IDLE and WAIT. Other information follows by state.

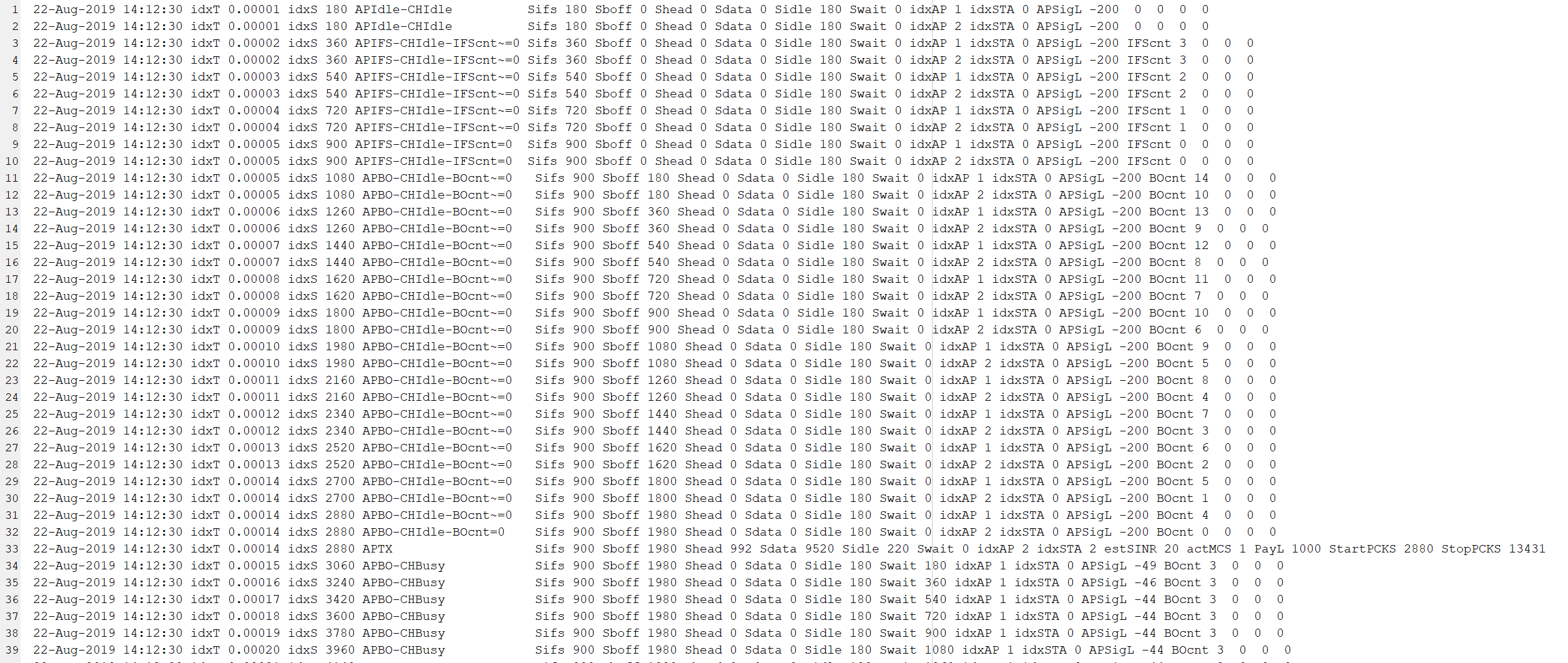


Figure 18: Log file part one for CSMA/CA example.

Records one and two show each AP in APIdle. The additional columns are the affected AP idxAP and STA idxSTA. For idle, there is no affected station which is recorded as station zero. The APSigL is the signal power at the AP, needed for CCA. Below minimum sensitivity level is recorded as -200 dBm.

Records three to ten show both APs counting down the IFS slots. Both APs reach APBO state in records 11 and 12, but with different backoff times.

At record 33, AP two wins the contention and enters the APTX state. It transmits to STA two with an estimated SINR of 20 dB using MCS 1 with a payload of 1000 bytes. STARTPCKS and STOPPCKS do show the samples (as an array index) where the packet is placed on the channel. The SINR is generated by a moving average filter of last transmissions. The 20 dB is a placeholder and will move to the current value after some transmissions.

Starting record 34, the backoff counter of AP one becomes zero. But the AP cannot transmit, as the channel is busy. The APSigL shows the assessed signal strength at the AP with -49 dBm. Taking again a moving average filter into account, the level moves quickly to -44 dBm.

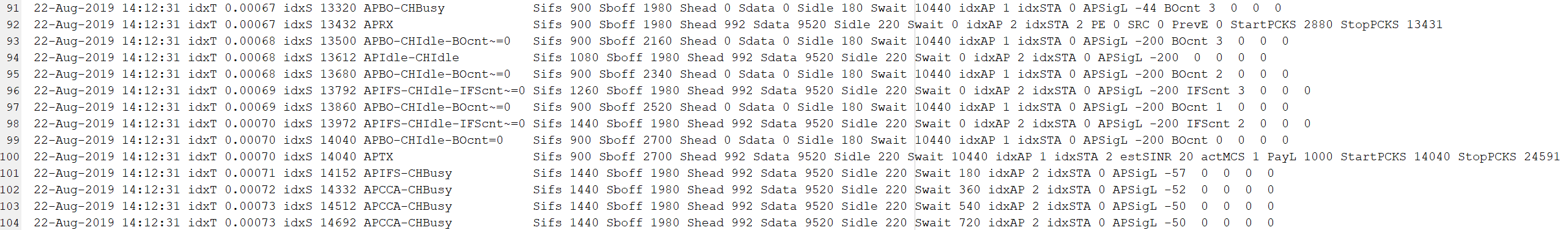


Figure 19: Log file part two for CSMA/CA example.

The channel stays busy until the packet from AP two is received. Figure 19 shows this situation. Record 92 shows a packet received from AP two to STA two with no packet error (PE=0), system retry count zero and previous error zero. The exact samples of the packet are also recorded.

Starting record 93, AP one continues with decrementing the backoff counter as the channel is free now. The same time, AP two starts a new IFS cycle.

Record 100 shows the transmission of AP one as the backoff counter reaches zero.

Note: ACKs are not shown in this example as ACK was implemented at a later time.

Figure 20 is one plot from the used example. 42 packets were send during that 10 ms simulated time. The plot shows clearly the time spend in the different states of the APs. Obviously, a good part of time is waiting for the channel to become free as the APs do share the medium.



Figure 20: Timing plot for CSMA/CA example

# Simulation Software Architecture

The implementation of the platform uses a three level approach, as outlined in figure 21. This allows to add new simulation types and medium access approaches with a fair amount of work.

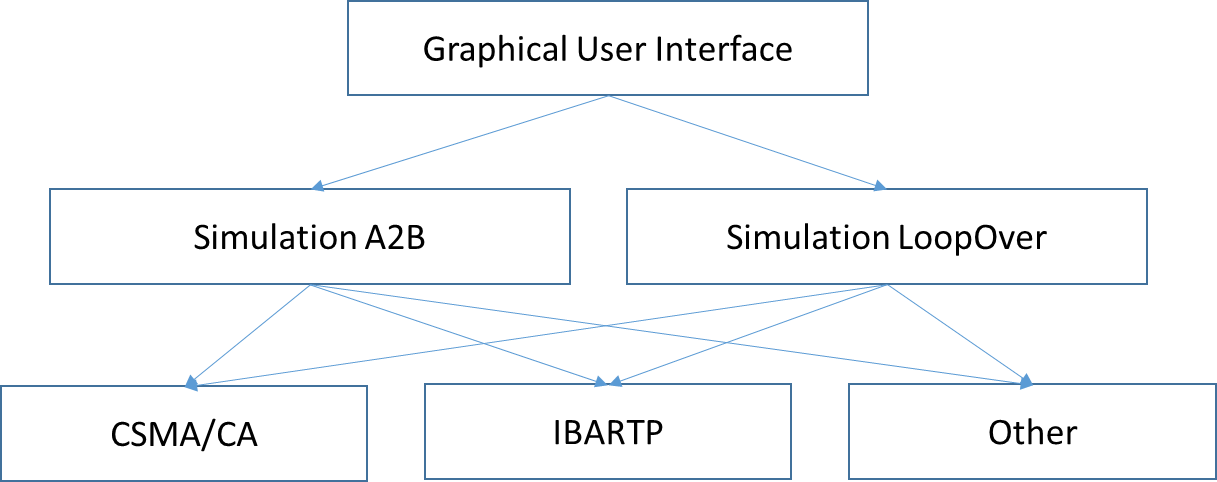


Figure 21: Simulation Software Architecture.

Level one is the graphical user interface (file start\_here\_platform\_GUI.m). This script sets up the GUI to define BSS, obstacles and the simulation. It calculates some key values like pathloss, if any relevant parameters are changed, and passes to the next level according to the selected simulation type A2B or LoopOver. The whole setup is mostly captured in cell arrays like BSS (variable all\_bss), obstacles (variable all\_obstacles), paths between stations (variables all\_path\_STA, all\_path\_AP), pathloss between stations (varaibles all\_pathloss\_STA, all\_pathloss\_AP) and simulation parameters (variable simulation). Results of the simulations are given back to the GUI via figure and axes objects by STA(variable resultU) and by AP and system (variable resultB).

Level two defines the simulation type. Implemented are an A2B comparison (file sim\_A2B.m) and a loop approach to test parameter changes (file sim\_loop.m).

For the A2B approach the overview flow is:

* Define common parameters for both simulations.
* Build tgax channel and simulation objects.
* Setup parameters and run initialization for approach A.
* Prepare data load model on GEQ for approach A.
* Prepare garbage collection model on GEQ for approach A.
* Built MATLAB configuration objects for approach A.
* Run main loop / simulation, calling level three until time is over for approach A.
* Setup parameters and run initialization for approach B.
* Prepare data load model on GEQ for approach B.
* Prepare garbage collection model on GEQ for approach B.
* Built MATLAB configuration objects for approach B.
* Run main loop / simulation, calling level three until time is over for approach B.
* Build plots as defined in the simulation model for approaches A and B.
* Handle back the plots to the GUI.

For the loop approach the overview flow is:

* Define common parameters for all loop steps (outside the loop).
* Build tgax channel and simulation objects (outside the loop).
* Setup parameters and run initialization (inside the loop).
* Prepare data load model on GEQ (inside the loop).
* Prepare garbage collection model on GEQ (inside the loop).
* Built MATLAB configuration objects (inside the loop).
* Run main loop / simulation, calling level three until time is over (inside the loop).
* Build plots as defined in the simulation model for all loop steps.
* Handle back the plots to the GUI.

The main loop from level two calls level three for the different access methods and approaches for every single event. Level three actually implements a state machine model, as discussed in paragraph ‘CSMA/CA’.

There are a lot of functions, either in the named files or also in separate files. For more details see the comments in the code.

## Important parameters

Important parameters, if not set by the GUI, are defined on level 2 at the beginning of the sim-files:

* simulation.N\_dBm: noise floor including noise figure.
* simulation.SLOTTIME: etc.: relevant times from the standard for high efficiency.
* simulation.CCA\_ED: the clipping level used for CCA free / busy for both packets and energy.
* simulation.fs: sampling frequency; this is also defined in MATLAB configuration objects later.
* simulation.CCATime: this defines the moving average window for CCA.

There are also some important parameters defined during tgax channel object definition:

* tgax.ChannelBandwidth: fixed to 20 MHz.
* tgax.CarrierFrequency: fixed to 2.4 GHz.
* tgax.EnvironmentalSpeed: disables the Doppler change for the fading channel.

Important parameters defined during simulation object definition are:

* cfgSimu{idxBSS,idxSTA}.IdleTime: shortest idle time, needed for signal expansion during channel filtering.
* cfgSimu{idxBSS,idxSTA}.GuardInterval: fixed to shortest guard interval.

More parameters are discussed in the Usage section.

# Other

Please report any errors or suggestions to michael.knitter@tu-dortmund.de