Simulation of 4G Cellular Communication for Unmanned Air Vehicles (UAVs)

Shafagh Jafer
Stephen Jones
Embry-Riddle Aeronautical University
Daytona Beach, FL
jafers@erau.edu, **s.jonesy72@gmail.com**

ABSTRACT

The data throughput achieved by modern fourth-generation (4G) Long Term Evolution-Advanced (LTE) mobile communications networks meets or exceeds the data-rate requirements for small and medium sized Unmanned Aerial Vehicle (UAV) communication data links. This creates an exciting opportunity to exploit pre-deployed cellular communication network infrastructure for UAV communication. UAVs are highly mobile; thus a challenging aspect in the implementation of a LTE UAV data link is maintaining a reliable network connection and high data rate, particularly at cell edges. This paper explores the utilization of 4G-LTE network as the communication topology among UAVs and ground base-stations. Using OMNeT++ modeling and simulation technology, we have developed a highly configurable and extensible framework providing core models such as UAVs, base stations, and network protocols to simulate various UAV communication scenarios. The preliminary results such as path loss, signal receive power, and data rate reveal promising outcomes demonstrating the capability of 4G-LTE network in serving UAVs with reliable data link.

Author Keywords

Unmanned air vehicle communication; UAV Data Link; OMNET++; 4GLTE; modeling and simulation.

ACM Classification Keywords

I.6.4 SIMULATION AND MODELING (e.g. Model Development).

INTRODUCTION

There have been rapid advances in mobile communications technology globally over the past decade. Mobile communication devices have become inexpensive and are utilized by a large majority of people on a regular basis. As mobile data traffic inevitably escalates, the limited electromagnetic spectrum and associated resources must be utilized more efficiently to sustain the increasing capacity requirements.

The fourth-generation (4G) Long Term Evolution-Advanced (LTE) mobile communication standard has recently been deployed in many parts of the world. This standard delivers ultra-band data throughput of over 100Mbps in a single 20MHz carrier and up to 1Gbps with carrier aggregation [1]. The standard also boasts improved multiple-access schemes that facilitates improved quality of service as well as more efficient spectrum usage compared to previous deployed mobile communication standards.

Unmanned Aerial Vehicles (UAVs) are highly mobile aerial platforms with a wide gamut of capabilities. UAVs are currently used in various domains to perform a multitude of missions. A fundamental subsystem of any UAV is the data link. The data link for a UAV system provides two-way communication, either upon demand or on a continuous basis. An uplink with a data rate of a few kilobits per second (kbps) provides control of the UAV flight path and commands to its payload. The downlink provides both a low data rate channel to acknowledge commands and transmit status information about the UAV and a high data rate channel of 1 to 10 megabits per second (Mbps) for sensor data such as video and radar [2]. The microwave spectrum (1 GHz to 100 GHz) is commonly utilized for UAV data links to facilitate high data rate and minimize antenna size [3]. The data throughput achieved by LTE networks in practice meets or exceeds the data-rate requirements for data links that communicate with current small and medium sized UAVs [1] [2] [3]. This creates an exciting opportunity to exploit pre-deployed cellular communication network infrastructure for UAV communication. LTE networks already exist in many countries throughout the world and coverage is constantly expanding. If UAV communication via LTE proves to be feasible, the cost and complexity of UAV deployment could be greatly reduced, as the communication infrastructure already exists and is permanently and robustly deployed. Thus, the data link infrastructure; particularly the ground control station equipment required to maintain a data link with a UAV; could be simplified substantially. UAVs are highly mobile; therefore a challenging aspect in the implementation of a LTE UAV data link is maintaining a reliable network connection and high data rate, particularly at cell edges (areas served by more than one base station, where network coverage overlaps).

In order to study the feasibility of cellular network in serving UAVs' communication medium, here we utilize the cost-effective and risk-free Modeling and Simulation (M&S) approach. After studying available M&S technologies in the domain of network communication, we selected OMNeT++ to serve our research purpose. Only a small amount of work has been published in the literature on the topic of exploiting existing LTE cellular communication network infrastructure for UAV data links. The aim of this research work is thus to provide a modeling and simulation framework for assessing the feasibility and effectiveness of this prospect in various UAV scenarios, with an initial focus on the LTE physical layer aspects. This paper will tackle this challenge by providing the following insights: a literature review on UAV data links and M&S approaches in this domain, our M&S approach and model definitions in the context of OMNET++ tool, model verification process used in this work, simulations scenarios and statistical analysis, and finally concluding remarks and future enhancement of the research.

LITERATURE REVIEW

Mobile (wireless) data traffic is growing rapidly and 3G networks are quickly reaching capacity. On the other side, the Long Term Evolution-Advanced (LTE) has become the most promising standard for upcoming 4G communication systems. Challenges in providing high-speed data in mobile networks are stated as the low data rate for cell-edge users as well as coverage gaps. There exists a distributed processing approach to solve these issues, known as Coordinated Multi-Point (CoMP). The work in [1] defines and evaluates a model for analysis of performance of LTE mobile network architectures defined by the 3rd Generation Partnership Project (3GPP) [4] [5]. As described in [4], network simulations are often divided into two stages or levels of abstraction known as link-level and system-level. The link-level simulations are used to assess the performance of the physical layer and those higher layer aspects directly related to the radio interface. In link-level simulation, a single-cell radio link is modeled, including some specific features such as synchronization, modulation, channel coding, channel fading, channel estimation, demodulation and multi-antenna processing. On the other hand, a system-level simulator allows evaluation of the performance of a network comprising multiple cells and moving mobile stations. At this level, system modeling encompasses a set of base stations and all their associated mobile terminals. Both the signal level received by each user and other users' interferences are modeled, taking into account the propagation losses and channel fading effects. Signal to interference plus noise ratio (SINR) is calculated for each active user considering the current network configuration and relative random placement of the base stations and mobile stations. These SINR values can be then translated to block error rate (BLER) or effective throughput values using models whose development is based on the results obtained in the link-level simulations.

The simulation framework presented in [6] consists of a link level simulator. The OMNeT++ [16] simulation tool and associated INET framework [14], MONAMI LTE and ChSim [19] channel simulator were used to develop the aerial-terrestrial network scenario. A simple mobility model was used for the tethered Low Altitude Platforms (LAP), which differs significantly to a highly mobile UAV that we are interested in. However, the work presented in [6] is still applicable to this research work as it demonstrates that utilizing LTE for high-speed communication to and from aerial platforms is feasible. It also reveals that the OMNeT++ simulation tool and associated frameworks produce credible, comprehensible results and are thus powerful and capable tools for simulating aerial communications with LTE. This work translated well to this research as the design requirements presented for a LAP are similar to those required of mobile UAVs. The work in [7] proposes a holistic and rapidly deployable mobile network architecture based on a hybrid aerial-terrestrial approach, which is flexible to be adapted to different scenarios based on the characteristics of the aerial platforms and the choice of deployment of the LTE-specific system components. The work claimed to have implemented the LTE logical and physical layers according to 3GPP Release-10 (LTE-Advanced) through extension of the OMNeT++ MONAMI LTE implementation using the ChSim channel models.

Although several open source LTE simulators exist (LTESim [20], ns-3 [11] and OMNeT++ [16]), such simulators and associated models are not entirely applicable to aerial-terrestrial communication scenarios. Numerous aerial modeling and simulation tools were examined in the initial stages of the research work, including those presented in [8] and [9]-[13]. These papers utilized the Objective Modular Network Testbed in C++ (OMNeT++) modeling and simulation tool. Based on our research and the comparisons presented in [8] and [9]-[13], OMNeT++ was considered the most suitable simulation tool and was selected for our project.

MODELING AND SIMULATION APPROACH

This section describes the methodology that was followed to develop the modelling and simulation framework for UAV data links utilizing 4G LTE cellular networks. An overview of the selected tool is discussed followed by the framework design and implementation.

OMNeT++ Overview

The Objective Modular Network Testbed in C++ (OMNeT++) is an object-oriented modular discrete event network simulation framework that includes an integrated development and graphical runtime environment. Its generic architecture boasts great extensibility and allows it to be utilized in various domains. OMNeT++ is most commonly used to model wired and wireless communication networks and associated protocols, however it is useful for modelling and simulating any system where the discrete event approach is suitable. It is important to understand that OMNeT++ is not technically a network simulator; it includes the basic machinery and tools to write simulations, but it does not provide any components for any specific domain. Like other simulation tools, OMNeT++ utilizes a component architecture for simulation models. The building blocks of simulation models are reusable components termed modules. Modules may be connected together via gates (other tools may name them ports) and nested inside other modules to create compound modules. There is no limitation on the depth of module nesting that may be achieved. Modules communicate via message passing and the structure of messages is highly extensible. Messages may represent events, packets, commands, jobs or other entity relevant to the model domain. Communication paths can be defined by connecting gates together; or alternatively OMNeT++ also supports direct-messaging where messages are passed directly from source to destination without needing to follow a predefined path. This is most useful for simulating wireless communication systems. OMNeT++ is released under an academic public license. It is free to use for academic and non-profit purposes. The OMNEST application has been developed by Simulcraft Inc. and is supported for commercial use. OMNeT++ is available on most common operating systems including Mac OS/X 10.7, 10.8, 10.9 and 10.10; Windows XP, 7 and 8; and many Linux distributions. Installation is simple and the process is well described in the installation guide that is distributed with the OMNeT++ package.

Modeling Concepts

As stated above, an OMNeT++ model encompasses modules that communicate with message passing. The most basic form of a module is termed a simple module. Simple modules encapsulate model behavior and are written in the C++ programming language, utilizing the extensible simulation class library. Simple modules may be grouped together to form modules that are termed compound modules. Compound modules may contain multiple levels of other compound modules; the amount of hierarchy levels that can be used is unlimited, but what remains consistent is that simple modules always make up the lowest level of the hierarchy. The entire model is in fact a compound module and is termed a network. As described before, OMNeT++ supports message passing via gates and connections that span modules, or alternatively through direct-messaging from module to module without a simulated physical connection path. Gates represent the input and output interfaces of modules. A gate may be an *input*, *output*, or *inout* gate. Gates may be linked together by a connection. Logically an *input* gate cannot be connected to another *input* gate, and likewise with *output* gates. *Inout* gates may be connected however. Each individual connection must exist within a single level of the module hierarchy, as connections spanning hierarchy levels inhibit model reuse. Compound modules thus act like virtual boxes and provide an interface between their inner entities and the outside world through which messages are transparently relayed. Connections may also be assigned properties such as propagation delay, data rate, bit error rate and packet error rate. Connection types with specific properties may also be defined. These connection types are termed channels and multiple instances of channels may be declared and used throughout a model. The components of an OMNeT++ model are illustrated in Figure 1.

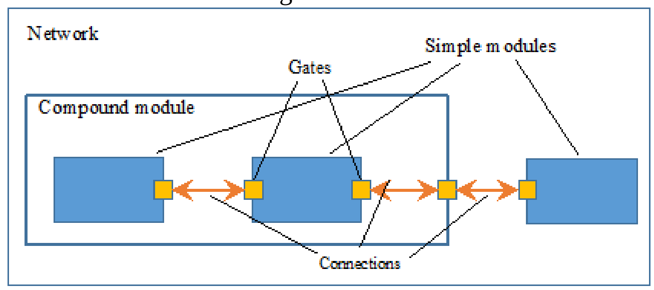

Figure 1: OMNeT++ Model Components [15]

Model Structure

The structure/topology of a simulation model is described in OMNeT++'s Network Description (NED) language. NED facilitates declaring simple modules and connecting and assembling them into compound modules. Compound modules can even be declared as networks, which are self-contained simulation models. Simple module declarations describe the external interface of the module: gates and parameters. Compound module definitions consist of the declaration of the module's external interface (gates and parameters), and the definition of submodules and their interconnection.

Configuration

An *ini* file defines the NED type to be instantiated as the network in the simulation; provides values for model parameters that do not have default values or the default is not suitable; and contains simulation options such as simulation time limit, Random Number Generator (RNG) configuration, names of output files, or the set of statistics to be recorded to files. *ini* files may contain multiple named configurations. Configurations may inherit from one other; adding new settings or overriding existing settings. The default configuration file is named *omnetpp.ini*. The file may be renamed but is typically kept as this default file name.

Message Passing and Handling

Simple modules are implemented as C++ classes, derived from the *cSimple* module library class. The most frequent tasks carried out by simple modules are message sending and processing of received messages. Messages are defined by the *cMessage* class, and can be sent either via output gates, or directly to other modules using direct-messaging. Simulation events are realized through messages in OMNeT++. The simulation kernel delivers messages to the *handleMessage(cMessage*)* function of a module. This function needs to be overridden to add functionality to a module. A module may send messages to itself; these such messages are termed self-messages. Timers and timeouts are typically implemented through self-messages. Self-messages are sent via a call to *cSimpleModule.scheduleAt(simtime_t, cMessage*)*, and are delivered back to the module in the same way as messages arriving from other modules. Self-messages can also be canceled.

Design Process

The process used to design and implement the subject framework of this research work is based on the workflow provided at [16]. In summary, the process is as follows (note that this process is slightly modified and the steps do not perfectly align with those at [16]):

1. Define the modules required to accurately represent the model.
2. Define the model structure/topology.
3. Define the behavior of simple modules.
4. Define the statistics to be gathered.
5. Define the global model parameters and configuration.
6. Build and execute the simulation.
7. Analyze the results.

The simulation framework was developed in two main stages, as follows:

Stage 1: A generic model was developed to incorporate a channel model and verify that it was functioning correctly. This stage was also utilized to ascertain that system modules could communicate wirelessly with each other; and ultimately gain a better working knowledge of modelling and simulation concepts and the OMNeT++ tool.

Stage 2: The initial generic model was extended to incorporate multiple BSs on various communication channels and a simple network-wide handover protocol.

The framework design and implementation process will be described in the remainder of this section.

Defining Required Modules

Recall that the objective of this research work is to develop a framework for analyzing UAV data links utilizing 4G LTE cellular networks, with an initial focus on the physical layer. Thus, models created utilizing this framework would logically encompass UAV and Base Station (BS) entities. A model of the communication channel is also essential.

The most difficult part of this step was to determine how wireless communication is implemented in OMNeT++/INET. The wireless functionality exists in the INET framework, which is not well documented. Through large quantities of experimentation it was determined that in order to properly implement wireless communication and simulate it correctly with OMNeT++ (without modifying the simulation kernel), the following modules must exist in the network:

- *Radio* modules must exist within the network. *Radio* modules model the transmission and reception of wireless packets, and determine if packets were received correctly. *Radio* modules communicate change notifications to the *NotificationBoard* and *ChannelControl* modules (eg. radio state; transmitting, receiving, idle, etc.). When transmitting, *Radio* modules obtain the neighbor list from the *ChannelControl* module (ie. a list of other *Radio* modules tuned to the same channel), and then send a copy of the transmission packet to each neighbor. *Radio* modules should implement the *IRadio* module interface;
- A sole *Mobility* module must exist within each system-level module (even if the module is stationary). A *Mobility* module models the motion of a mobile host and communicates change notifications to the *NotificationBoard* and *ChannelControl* modules. *Mobility* modules should implement the *IMobility* module interface;
- A sole *NotificationBoard* module must exist within each system-level module. *NotificationBoards* make it possible for several modules to communicate in a publish-subscribe manner. A *NotificationBoard* acts as an intermediary between the module where the events occur and modules which are interested in learning about those events, such as the *ChannelControl* module. Notifications include change notifications (eg. coordinate position/location changes) and state changes (eg. radio transmitting; radio idle). The *NotificationBoard* will rarely require modification, and must be named "*notificationBoard*" to function correctly; and
- A sole *ChannelControl* module must exist at the network level. The role of this module is to manage message-passing on the wireless channels that exist in the network. The *ChannelControl* module gets informed about the location and movement of nodes (ie. UAVs and BSs), and determines which nodes are within communication or interference distance, based on the propagation model used. This information is then used by the radio interfaces of nodes when transmitting and receiving packets. The *ChannelControl* module will rarely require modification, and must be named "*channelControl*" to function correctly in the model.
- The *LinearMobility* module models a node moving in a straight line in the X-Y plane (ie. the Z coordinate, or altitude, remains constant). This model was useful for verifying the propagation model. The parameters of the *LinearMobility* model are the initial X, Y and Z coordinates, speed, angle and acceleration. If a node hits the edge of the playground, it reflects off the 'wall' at the same angle that it approached the wall.

Defining Model Topology

UAVs and BSs require a method of communication. As wireless communication is used, a radio module would be a logical building block. The INET framework already contained two implementations of radio models; *GenericRadio* and *Ieee80211Radio*. Subsequent to inspection and experimentation, it was decided to extend the *GenericRadio* module for the purposes of our work and create a simple module type named *BasicRadio*. A new module type was required to be created for this research work as the behavior of *GenericRadio* was not suitable; additional statistics-collection capability needed to be incorporated to facilitate accurate analysis of the model's physical layer. *BasicRadio* contains three gates as described in Table 1.

Table 1: Description of *BasicRadio* module gates

Direction	Gate Name	Purpose
Input	upperLayerIn	Input gate for receiving packets passed down from upper layer
Output	upperLayerOut	Output gate for passing packets up to upper layer
Input	radioIn	Input gate for receiving wireless packets on the radio channel

UAVs and *BSs* are compound modules comprised of the modules described earlier. Their only parameter is a mobility type. The stage 1 system-level compound module types were named *UAV* and *BaseStation* respectively. The stage 2 module types were based on the original modules with the *Source* and *Sink* module types replaced by a *MAC* module type. The block diagrams illustrated in Figure 2 and Figure 3 represent UAV Module and Base Station Module.

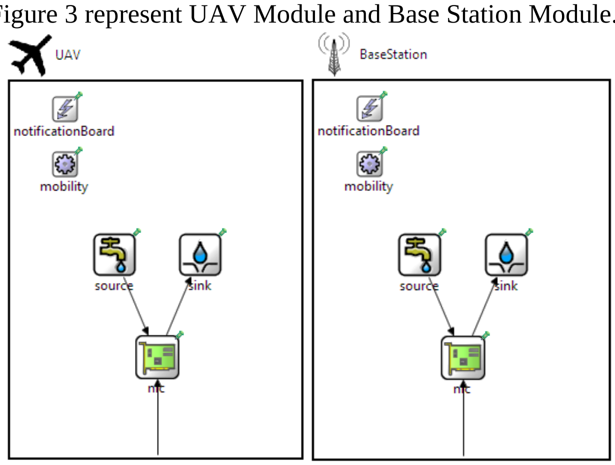

Figure 2: *UAV* and BaseStation Module Type Structure

Defining Module Behavior

As described previously, OMNeT++ module behavior is defined in C++ source. C++ is an efficient, object-oriented programming language which lends itself well to this application of modelling and simulation. Each simple module has associated C++ source code that defines its behavior. Modules details can be found in the applicable source files which are available on request.

The *ChannelControl* module utilizes a propagation model for calculating the transmission time and signal receive power of all wireless transmissions within the network. The propagation model defined in the 3GPP Technical Report 36.942 [17] was implemented in OMNeT++ as part of this research work. This propagation model is useful for calculating path loss and signal receive power. The aerial context of this research application reflects a rural area setting, thus the rural area macro cell propagation model was utilized. The propagation model, L, is represented by the following formula [17]:

$$L(R) = 69.55 + 26.16\log_{10}(f) - 13.82\log_{10}(Hb) + [44.9 - 6.55\log_{10}(Hb)]\log_{10}(R) - 4.78(\log_{10}(f))^2 + 18.33\log_{10}(f) - 40.94$$

where R is the distance between UAV and BS in kilometres, f is the carrier frequency in MHz, and Hb is the base station antenna height above the ground in metres.

The pathloss is given by the following formula [17]:

$$Pathloss = L + LogF$$

where $LogF = 10dB$; representing log-normally distributed shadowing with standard deviation of 10dB. Note that this model describes worst case propagation for non-line of sight; and is designed mainly for distances from a few hundred meters to kilometres. This model is not very accurate for short distances [17].

The received power in downlink and uplink can be expressed as [17]:

$$RX\_PWR = TX\_PWR - Max(Pathloss - G\_TX - G\_RX, MCL)$$

where RX_PWR is the received signal power, TX_PWR is the transmitted signal power, G_TX is the transmitter antenna gain, G_RX is the receiver antenna gain, and MCL is the minimum coupling loss. By considering a continuous-time Additive White Gaussian Noise (AWGN) channel, the data rate capacity of the channel in bits/second may be determined from the following formula [18]:

$$Data\ Rate = B\log_2\left(1 + \frac{RX\_PWR}{N_0\,B}\right)$$

where B is the system bandwidth, and N_0 is the white noise power density (also known as thermal noise). The default simulation parameters are listed in Table 2 [1] [17].

UAVs are highly mobile vehicles, thus a mobility model is a vitally important component of the modeling and simulation framework developed as part of this research work. The INET framework already contained various mobility models, thus some of the models were utilized for this research work. The models are implemented as simple modules with associated C++ behavior implementations. For wireless simulations in OMNeT++, all nodes require a mobility module.

Table 2: Default Simulation Parameters

Parameter	Value
Carrier Frequency, f	900 MHz
BS Antenna Height Above the Ground, Hb	45 m
Log-normal Fade Shadow, $LogF$	10 dB
UAV Transmit Power, TX_PWR_{UAV}	126 mW (21 dBm)
BS Transmit Power, TX_PWR_{BS}	19.95 W (43 dBm)

UAV Antenna Gain, G_TX_{UAV} / G_RX_{UAV}	0 dB
BS Antenna Gain, G_TX_{BS} / G_RX_{BS}	15 dB
Minimum Coupling Loss, *MCL*	80 dB
System Bandwidth, *B*	5 MHz
White Noise Power Density, N_0	-174 dBm/Hz
Radio Sensitivity	-110 dBm

Defining Statistics to be Conducted

OMNeT++ contains built-in support for recording and analysis of simulation results. User-defined data may be recorded to output vectors and output scalars. Output vector files (*.vec*) and output scalar files (*.sca*) are stored in the /simulations/results/ directory by default at the completion of a simulation run. Output vector and output scalar files are textual, line-oriented files. This ensures flexibility, as the text-based format may be accessed and manipulated with a wide range of tools and languages. One such tool that may be utilized to access and display the data residing in output vector and scalar files is the Analysis Tool built into the OMNeT++ Simulation IDE. The tool is most valuable for displaying the data in a format that is simple to read and understand. Further details regarding the result recording and analysis capabilities of OMNeT++ can be found in [15]. The important statistics to be gathered within this simulation framework for subsequent analysis are considered to be the following:

- Signal receive power in dBm;
- Distance travelled by packets in meters;
- Signal path loss in dB;
- Upload and download data rate capability in Mbps;
- The location of UAVs and BSs in X-Y-Z Cartesian coordinates;
- Channel numbers assigned to each BS; and
- Channel numbers that UAVs are tuned to.

MODEL VERIFICATION

In order to verify the propagation model that was developed as part of this research project, a simple scenario was configured utilizing the *BasicNetwork* network where a UAV would begin with 30 km separation from a BS and fly directly towards the BS at 30 meters per second. The UAV was at the same altitude as the BS (45m) with altitude remaining constant. The *LinearMobility* mobility model was utilized for this purpose, and statistics were gathered for every packet that was received by the UAV and BS, regardless of the radio sensitivity level. The simulation ran for 1,002 seconds so that the UAV will be in effectively the same position as the BS at the end of the simulation. Each measurement for receive power in dBm and data rate in Mbps that was collected was compared to the expected value calculated in Microsoft Excel 2013 directly from the system of equations in previous section. This is a fair comparison as a deterministic model is being used. Table 3 provides a snippet of the first five calculations made for the upload direction. For the upload direction, the receive power in dBm was accurate to within 0.0036% of the expected value on average, with a standard deviation of $6.6 * 10^{-4}$ %, while the data rate in Mbps was accurate to within 0.059% of the expected value on average, with a standard deviation of 0.0246%. For the download direction, the receive power in dBm was accurate to within 0.00072% of the expected value on average, with a standard deviation of $2.01 * 10^{-6}$, while the data rate in Mbps was accurate to within 0.0030% of the expected value on average, with a standard deviation of $1.28 * 10^{-5}$. The small margin of error is presumably due to loss of precision when data types are converted in the simulation and/or in Microsoft Excel. Regardless, the model has been verified to be implemented correctly.

Table 3: Propagation Model Verification for Upload Direction

Simulation Time (s)	Distance (m)	Expected Receive Power (dBm)	Actual Receive Power (dBm)	%accuracy dBm	Expected Data Rate (Mbps)	Actual Data Rate (Mbps)	% Accuracy Mbps
0.0001	30000	-119.807	-119.80	99.996792	0.369256	0.369563	100.08326
0.6281	29982	-119.798	-119.79	99.996692	0.369992	0.3703	100.08315
1.0897	29970	-119.792	-119.78	99.996628	0.370485	0.370793	100.08324
1.4833	29958	-119.786	-119.78	99.996566	0.370978	0.371286	100.08311
2.4228	29928	-119.771	-119.76	99.997255	0.372214	0.372524	100.08327

The results of this scenario reveal that the maximum achievable data rate in the upload direction is 79.75 Mbps, and the maximum achievable data rate in the download direction is 116.28 Mbps. Figure 3 plots transmission distance vs upload data rate for valid packets received with a receive power greater than -110 dBm. Figure 4 plots transmission distance vs download data rate for distances up to 30 km. The maximum transmission range of UAV and BS modules was also verified. By substituting a receive power of -110 dBm into the system of equations presented in the previous section of this paper and solving for R, it can be determined that the maximum distance that a UAV can transmit a valid packet to a BS is 15.46 km. Similarly, the maximum distance that a BS can transmit a valid packet to a UAV is 68.39 km. This is accurately modeled within the developed framework.

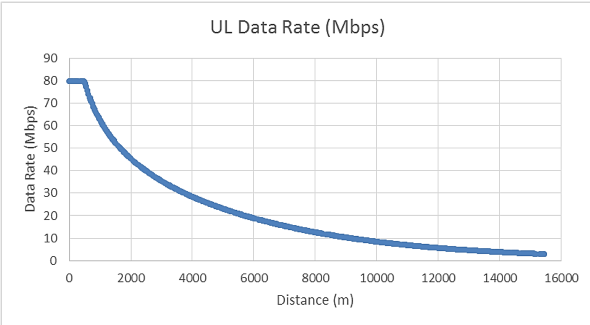

Figure 3: Distance vs Upload Data Rate

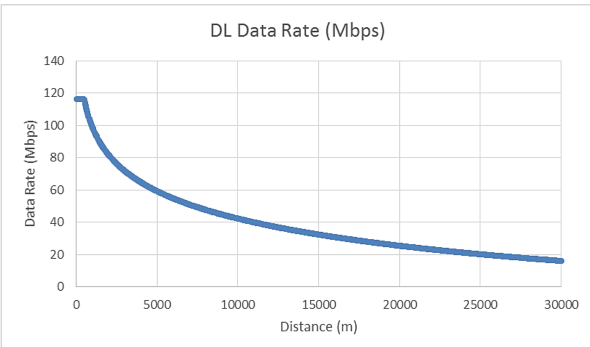

Figure 4: Distance vs Download Data Rate

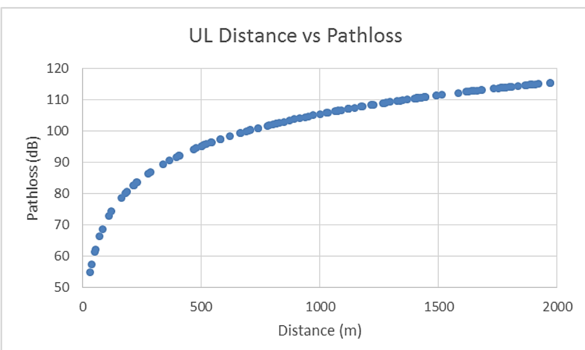

Figure 5: Upload Distance vs Pathloss Verification

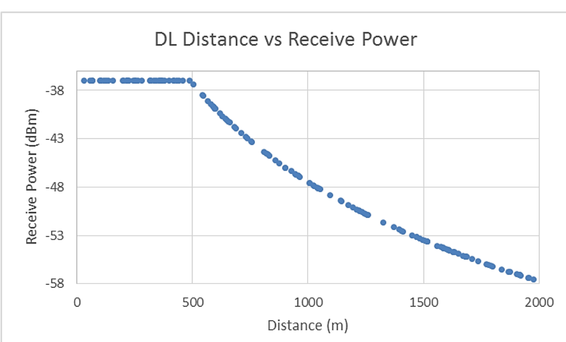

Figure 6: Download Distance vs Receive Power Verification

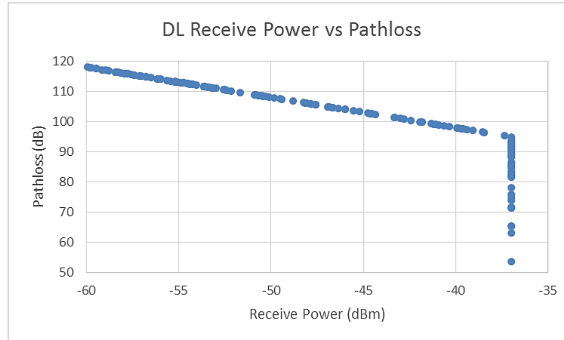

Figure 7: Download Receive Power vs Pathloss Verification

The model was also verified against the model published in [1]. The charts illustrated in Figure 5 to Figure 7 represent the data gathered from the verification simulation run. These charts accurately reflect Figures 7 to 9 in [1] for rural settings. This further confirms that the model has been correctly implemented.

RESULTS AND ANALYSIS

This section will analyze the results of various scenarios that were executed to further verify the developed framework. Some results and analysis have already been documented in the Model Verification section of this research report. In order to properly test the developed framework for a range of different inputs, various generic scenarios were developed and executed. The generic scenario parameters are summarized in Table 4. The value of N was varied to verify scalability of the respective parameters. The actual values used for each scenario are presented in Table 5. Scenario 1 was discussed earlier in the model verification section of this report which involved one UAV and one base station. Scenario 2 was executed correctly for values of N up to and including 20, while there was only one base station in each case. 20 UAVs is considered to be an extreme case for these scenarios, thus values of N above 20 were not tested (the quantity of UAVs and BSs that can be utilized in a simulation is limited only by the 16-bit integer size and system memory size. UAV/BS collisions are not simulated).

Table 4: Description of Generic Scenarios used for Analysis

Scenario ID	Scenario Description		Network Type	Rationale
	# UAVs	# BSs		
1	1	1	BasicNetwork	Proof of Concept
2	N	1	BasicNetwork	UAV Scalability

Table 5: Actual Parameters Used for Scenarios

Scenario ID	Actual Parameters Used	
	# UAVs	# BSs
1	1	1
2	≤ 20	1

All scenarios shared some common parameters. For instance, the playground area was 30 km by 30 km by 20 km in size by default. Other parameters are those presented in [17] and Table 2 of this paper. Given the limited space of this paper, in this section we can only show upload execution results for the scenario with 5 UAVs. Our overall scenarios conducted the following statistics: UL/DL Distance, UL/DL receive Power, and UL/DL Data Rate. Similar behavior was observed for various numbers of UAVs ($2 < N \leq 20$).

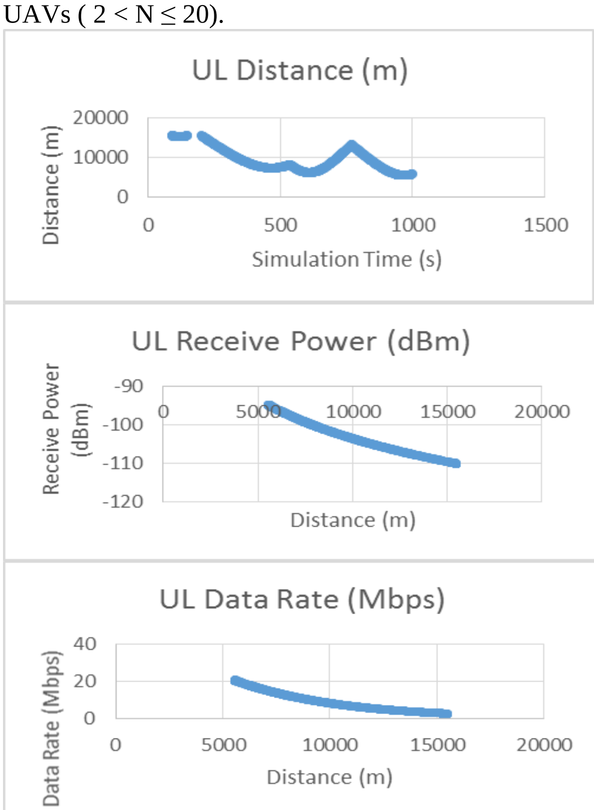

Figure 8. Sample execution results for 5 UAVs scenario.

CONCLUSION

As the data throughput achieved by LTE networks in practice meets or exceeds the data rate requirements for data links that communicate with current small and medium sized UAVs, an exciting opportunity exists to exploit pre-deployed cellular communication network infrastructure for UAV communication. This work documents the development of modeling and simulation modules for UAV data links utilizing 4G LTE cellular networks. The powerful OMNeT++ modeling and simulation tool was selected as the most suitable tool for achieving the objectives of the project, and it proved to be very effective in combination with the associated INET framework. We have implemented core simulation models including UAV, base station, and basic network. The developed M&S baseline is highly configurable and extensible allowing to create much more complex scenarios including N to N UAV and BaseStation with other network protocols. Path loss, signal receive power and data rate capacity may be easily modeled for an endless array of scenarios. The simulation results reveal that download data rates in excess of 110 Mbps and upload data rates in excess of 75 Mbps are theoretically achievable at short ranges up to 500 meters with standard LTE-compatible hardware. We are currently implementing HandOver Network Protocol to investigate vast number of UAVs scenarios with more complicated network configurations. This work may also be further refined by academia, as all source code developed as part of this project has been published under an open-source license and is available on request.

REFERENCES

[1] M. Moallemi, G. Wainer, S. Jafer, G. Boudreau and R. Casselman. “Simulation of mobile networks using Discrete Event System Specification theory.” 2013. Available: <http://cell-devs.sce.carleton.ca/publications/2013/MWJBC13/p7-MWJBC13.pdf>.

[2] P. Fahlstrom and T. Gleason. “Aerospace Series: Introduction to UAV Systems.” 4th Edition. Wiley, NJ. 2012.

[3] W.W. Rochus. “UAV Data-Links: Tasks, Types, Technologies and Examples.” RTO Applied Vehicle Panel (AVT) Special Course on Development and Operation of UAVs for Military and Civil Applications, RTO EN-9. 1999.

[4] S. Ahmadi. “LTE-Advanced: a practical systems approach to understanding 3GPP LTE releases 10 and 11 radio access technologies.” Academic Press. ISBN: 9780124051621. 2014.

[5] 3GPP LTE-Advanced Home Page. [Online]. Accessed 20 Nov. 2015. Available: <http://www.3gpp.org/LTE-Advanced/>.

[6] K. Gomez, T. Rasheed, L. Reynaud and I. Bucaille. “Realistic deployments of LTE-based hybrid aerial-terrestrial networks for public safety.” 2013 IEEE 18th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD), pp.233-237. 2013.

[7] K. Gomez, T. Rasheed, L. Reynaud and S. Kandeepan. “On the performance of aerial LTE base-stations for public safety and emergency recovery.” Globecom Workshops (GC Wkshps), 2013 IEEE, pp.1391-1396. 2013.

[8] N.I. Sarkar and S.A. Halim. “A review of simulation of telecommunication networks: simulators, classification, comparison, methodologies, and recommendations.” 2011.

[9] D. Martin-Sacristan, J.F. Monserrat, V. Osa and J. Cabrejas. “LTE-Advanced System Level Simulation Platform for IMT-Advanced Evaluation.” 2011.

[10] G. Piro, L. A. Grieco, G. Boggia, F. Capozzi, and P. Camarda. “Simulating LTE cellular systems: an open source framework.” IEEE Trans. Veh. Technol., Vol: 60 ,2 Pp: 498 – 513. 2010.

[11] N. Baldo, M. Miozzo, M. Requena-Esteso and J. Nin-Guerrero. “An open source product-oriented LTE network simulator based on ns-3.” Proc. ACM MSWiM, 2011.

[12] M.A. Rahman, A. Pakštas and F.Z. Wang. “Network modelling and simulation tools.” Simulation Modelling Practice and Theory, Volume 17, Issue 6, Pages 1011-1031. 2009.

[13] W. Kasch, J. Ward and J. Andrusenko. “Wireless network modeling and simulation tools for designers and developers.” Communications Magazine, IEEE, vol.47, no.3, pp.120,127. 2009.

[14] INET Framework Home Page. [Online]. Accessed 03 Apr. 2015. Available: <http://inet.omnetpp.org/>.

[15] A. Varga. “OMNeT++ User Manual, Version 4.6.” OpenSim Ltd. 2014.

[16] "Simulation with OMNeT++." OpenSim Ltd. [Online]. Accessed 03 Apr. 2015. Available: <http://www.omnetpp.org/documentation/3632>.

[17] 3rd Generation Partnership Project, “Technical Report 36.942, V11.0.0; 3rd Generation Partnership Project; Technical Specification Group Radio Access Network; Evolved Universal Terrestrial Radio Access (E-UTRA); Radio Frequency (RF) system scenarios (Release 11)”, 2012.

[18] D. Tse and P. Viswanath. “Fundamentals of Wireless Communication.” Cambridge University Press, New York. 2005.

[19] S. Valentin, ChSim - A Wireless Channel Simulator for OMNeT++, in TKN TU Berlin Simulation Workshop, September 2006. Software and documentation available at http://wwwcs.upb.de/cs/chsim.

[20] G. Piro, “LTE-Sim - the LTE simulator,” [OnLine] Available: http://telematics.poliba.it/LTE-Sim.