A Trace-based ns-3 Simulation Approach for Perpetuating Real-World Experiments

Helder Fontes
INESC TEC and Faculdade de Engenharia, Universidade do Porto
Porto, Portugal
hfontes@inesctec.pt

Rui Campos
INESC TEC and Faculdade de Engenharia, Universidade do Porto
Porto, Portugal
rcampos@inesctec.pt

Manuel Ricardo
INESC TEC and Faculdade de Engenharia, Universidade do Porto
Porto, Portugal
mricardo@inesctec.pt

ABSTRACT

A common problem in mobile networking research and development is the cost related to deploying and running real-world mobile testbeds. Due to cost and operational constraints, these testbeds usually run for short time periods but generate very unique and relevant results that are hard to reproduce.

We propose the use of ns-3 as a solution to successfully reproduce real-world mobile testbed experiments. This is accomplished by feeding ns-3 with real testbed traces including node positions and radio link quality only. In order to validate our approach, the network throughput between a fixed Base Station and a Unmanned Aerial Vehicle (UAV) was measured in a real-world testbed. The experimental results were compared to the network throughput achieved using the ns-3 trace-based simulation and a plain ns-3 simulation. The obtained results show the high accuracy of the trace-based simulation, thus validating our approach.

CCS CONCEPTS

•**Networks** →**Network simulations; Network experimentation; Network measurement; Mobile networks; Mobile ad hoc networks;** *Protocol testing and verification;* Network protocol design;

KEYWORDS

ns-3, Mobile Network Simulation, Trace Based Simulations, Reproducibility of Experimental Conditions, Perpetuation of Real-World Mobile Testbeds, Offline Experimentation, Concurrent Testbed User Access

ACM Reference format:
Helder Fontes, Rui Campos, and Manuel Ricardo. 2017. A Trace-based ns-3 Simulation Approach for Perpetuating Real-World Experiments. In *Proceedings of the 2017 Workshop on ns-3, Porto, Portugal, June 2017 (WNS3 2017),* 7 pages.
DOI: http://dx.doi.org/10.1145/3067665.3067681

WNS3 2017, June 2017, Porto, Portugal

DOI: http://dx.doi.org/10.1145/3067665.3067681

1 INTRODUCTION

The development of new protocols for communication systems generally involves four phases: design, evaluation in a network simulator, evaluation in a real testbed, and deployment (Figure 1).

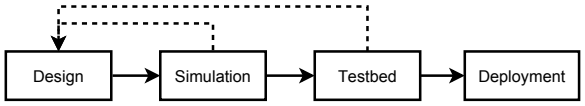

Figure 1: Development process of protocols for communication systems[6].

In the past years, we have been optimizing this development process, mostly relying on Network Simulator 3 (ns-3) [7]. In the beginning we only used ns-3 to create a simulator implementation of the solution, which was tested in multiple simulation scenarios. With the introduction of the ns-3 Emulation [8], we started using the Fast Prototyping of Network Protocols [3] methodology, which consists in reusing the same ns-3 code as a shared protocol model implementation between the simulation and the testbed (prototyping) environments, as illustrated by the green block in Figure 2. Later, we extended the compatibility of the ns-3 Emulation to support broader types of real-world networking scenarios [5], and improved its performance [6] with the goal of optimizing the fast prototyping of network protocols, represented by the blue arrow in Figure 2. In the work presented herein we explore how the simulation can benefit from the data gathered from the prototype (green arrow in Figure 2).

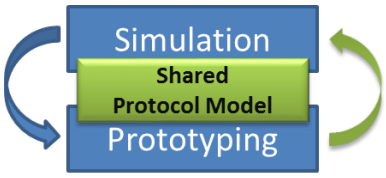

Figure 2: Simulation and Prototyping mutually beneficial relation via ns-3 Shared Protocol Model.

Based on our experience in past and current research projects, mobile testbeds are increasingly complex and costly to maintain. For instance, emerging testbeds include Unmanned Aerial Vehicles (UAVs) [10], Underwater Autonomous Vehicles (AUVs), and balloons [2]. Their use brings up difficult logistic operations and reduces the duration of real-world experiments, due to limited battery or fuel autonomy and the high costs associated with renting

boats and reserving airbase time slots. On top of this, the radio link quality – herein defined as the Signal to Noise Ratio (SNR) at the receiver – is highly unstable and the mobility of the nodes is difficult to reproduce in real-world testbeds, which makes experiments non repeatable.

To address the problem, the concept of trace based network simulation has been proposed in the state of the art. Two main approaches are found: 1) Packet Based Replay systems such as the one proposed in [9], where the authors capture traffic of real networks and try to reproduce the same experimental condition in simulation down to the per packet resolution including the same throughput and packet rate; 2) Application Layer Replay systems, as the one presented in [1], where the authors try to abstract all low level variables and reproduce the traffic delays and performance bottlenecks experienced in the real network at the application layer.

Our approach is simpler. We combine the ns-3 TCP/IP and MAC simulation capabilities with the physical characteristics of the real-world scenario, provided by real traces of node positions and radio link quality. This results in reproducible experimental conditions inside the simulator, effectively perpetuating a real-world experiment. The reproducible scenario can then be used to continue improving a solution under evaluation in simulation environment, which enables: 1) concurrent user access to the same exact experimental setup; 2) running simulations in faster than real time; 3) running multiple simulation instances at the same time, exploring different variants of the solution under evaluation.

The original contributions of this work are two-fold: 1) the approach of using ns-3 as a platform for perpetuating real-world experiments based on real traces of node positions and radio link quality; 2) a new ns-3 *TraceBasedPropagationLossModel*, which is able to reproduce the radio link quality in ns-3 from the real traces, including the ability to represent asymmetric radio link qualities.

The paper is structured as follows. In Section 2 we define our requirements, discuss the variables to be captured in the real traces, analyze the support and limitations of ns-3 to implement our approach and, finally, we present the proposed *TraceBasedPropagationLossModel*, including its implementation details. In Section 3 we validate our approach using a real-world scenario. Finally, in Section 4 we draw the conclusions and refer the future work to further improve the approach presented herein.

2 PROPOSED TRACE-BASED SIMULATION APPROACH

Based on our experience with UAVs operating over long distance TV White Spaces (TVWS) wireless links, we find the expected radio link quality very hard to model due to multiple factors, including: 1) the constant movement of the UAV that affects the position of the antenna; 2) the high noise floor that is often present on these sub-GHz frequencies, due to Electromagnetic Interference (EMI) and/or Radio-Frequency Interference (RFI) caused by close-by hardware and digital TV emissions in the order of kW in neighboring frequency channels. On the other hand, operating a testbed with one or more mobile nodes (e.g., UAVs) has high costs and logistics involved; as such, experiments are limited in number and time duration.

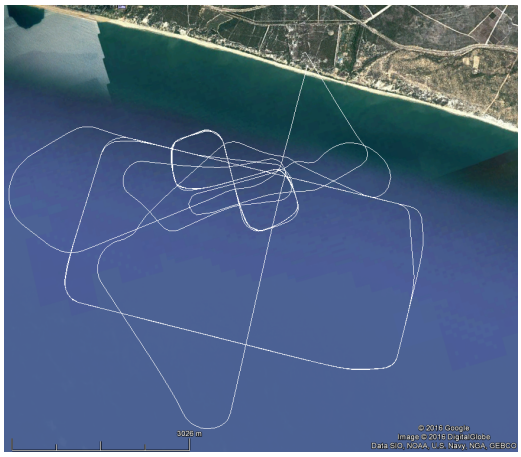

Figure 3: UAV path recorded in a real-world experiment.

An example of such real-world experiments are those being carried out in the FP7 SUNNY project [10]. In the SUNNY testbed there is a static node that acts as a gateway to the ground control operations – the Base Station (BS) – and UAVs that fly in trajectories such as the one presented in Figure 3. These UAVs can fly at altitudes ranging between 100 and 600 m and reach speeds up to 400 km/h. Such conditions result in very unstable radio link quality which are very hard to reproduce as we will show in Section 4.

Our goal is to capture the conditions of the real-world experiments and enable their perpetuation in simulation environment. This will allow reproducing the experiments offline and the evaluation of an unlimited number of solutions in the exact same conditions. To attain this goal our approach combines all built-in ns-3 capabilities for simulating the TCP/IP protocol stack with traces characterizing relevant physical parameters, such as the position and the radio link quality per node. We argue that in this way it is possible to reproduce real-world experiments without the complexity associated to the state of the art approaches.

2.1 Variables to be Collected

Our approach assumes the physical conditions experienced by real nodes can be characterized by means of two variables: 1) the node position along the time; 2) the quality of the radio links established with its peers. This information is used to feed the trace-based ns-3 simulation. In what follows, we refer to each of these variables and point out the way they can be collected in a real testbed.

- **Node GPS Coordinates.** In a real-world mobile testbed, the position of the nodes is frequently changing throughout the experiment. As such, its value shall be collected periodically. Using a GPS receiver and *gpsd* in any Linux based Operating System (OS) it is possible to obtain a 3D GPS coordinate (latitude, longitude, altitude) of the node per second, along with the UTC timestamp. This is enough to move the nodes in ns-3, according to the real waypoints.
- **Radio Link Characteristics.** In a real-world mobile testbed, the radio link quality is constantly changing throughout the experiment; so it shall be collected periodically.

The wireless interfaces installed in the real nodes are capable of reporting: 1) the Received Signal Strength Indicator (RSSI) (in dBm) for each of the neighboring nodes in radio range; 2) the total Noise Floor power (also in dBm) that accounts for the white noise, noise figure of the RX circuit, EMI, and RFI. Based on the Noise Floor and the RSSI, the expected Signal to Noise Ratio (SNR) for receiving packets from its peers can be calculated. It is important to note that this information should be collected at both ends of the radio link, because very often radio links are asymmetric. For instance, UAVs tend to have higher noise floor than Ground Station nodes, due to the proximity of the antennas to other electronic equipment in the UAV and the lack of physical obstacles to block interfering signals originated from distant sources on the ground.

Using a *bash* script it is possible to compile the output of the command *"date"* (outputs the time that can be synchronized with *gpsd* using *ntp*), the command *"iwinfo wlan0 info"* (outputs the noise floor) and the command *"iw wlan0 assocli"* (outputs the present Received Signal Strength Indicator (RSSI) per associated peer) to obtain the radio link quality once per second. If more resolution is needed, a packet capture program such as *horst* can be used to collect the RSSI per packet.

The Channel Frequency (MHz), Channel Bandwidth (MHz), TX-Power (dBm), Physical Rate (Fixed/Auto), and Wi-Fi standard (a/b/g/n) remain constant throughout a real-world experiment. These variables must be recorded once per experiment, as they are essential to make the ns-3 simulation scenario consistent with the real-world experiment.

2.2 Reproducing Real Node Positions in ns-3

ns-3 already implements the *MobilityModel*, which defines the concept of node position and enables node movement. The *WaypointMobilityModel* is specifically suitable for reproducing the positions of real nodes in ns-3. In the *WaypointMobilityModel* the node can be fed with a list of *Waypoints*, composed by a *Cartesian Coordinate Vector* associated with the respective simulation time that corresponds to each coordinate; ns-3 has a function that can convert the GPS coordinates of the real nodes into a *Cartesian Coordinate Vector*. Between each *Waypoint*, ns-3 moves the node at a constant speed such that the node arrives at the next *Waypoint* in the defined time.

Because the *Simulation Time* starts at 0 s, an *offset* between the *WallClockTime* and the *Simulation Time* needs to be defined so that everything is correctly scheduled in *Simulation Time*.

2.3 Reproducing Radio Link Quality in ns-3

ns-3 Wi-Fi nodes communicate between each other using a *Channel* abstraction that is a shared medium between the nodes. Two nodes need to be in the same *Channel* to communicate via a wireless link. To account for the propagation loss, ns-3 has several *PropagationLossModels* that can be associated to a *Channel*.

When a node is sending a frame, ns-3 calculates its Received Signal Strength (RSS) – equivalent of the RSSI in the real wireless cards – for the destination node, considering: 1) the TX-Power of the source node; 2) the TX gain of the source node (antenna gain); 3) the attenuation calculated by the *PropagationLossModel* associated to that *Channel* considering the distance between the nodes; 4) the RX gain of the receiver.

There is a *PropagationLossModel* named *FixedRssLossModel* which allows setting an RSS for a given channel, and makes the simulator ignore the attenuation calculated for the distance between the nodes; note that the RX gain is still considered after the *PropagationLossModel*, so it should be set to 0 when using the *FixedRssLossModel*, as recommended by the ns-3 manual. When using *FixedRssLossModel* the RSS value can be changed during the simulation execution. The problem is that it is only defined per *Channel*, and not per link between two nodes nor per link direction. This means that all the nodes communicating in the same Wi-Fi *Channel* have the same RSS. In order to reproduce the real radio link behavior, the RSS shall depend on 1) the peers involved in the communication and 2) the link direction. For this purpose, a new *PropagationLossModel* was developed, called *TraceBasedPropagationLossModel*. The new model is described bellow, in Section 2.4.

2.4 *TraceBasedPropagationLossModel*

The new *TraceBasedPropagationLossModel*, depicted in Figure 4, is a subclass of the existing *PropagationLossModel*. The new class includes the new attributes and operations necessary to implement the functionality required to reproduce asymmetric radio link quality based on real-world radio link quality traces. Throughout this section we present its implementation details. The ns-3 version 3.26 was used for implementing the model. The source code for the *TraceBasedPropagationLossModel* implementation can be found in [4].

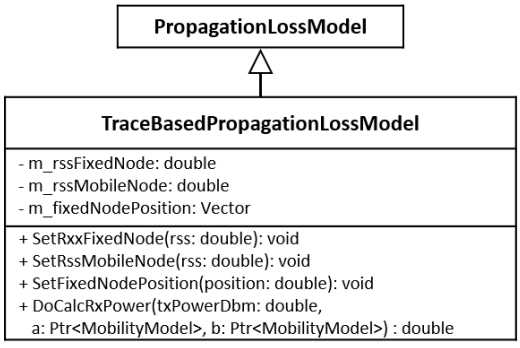

Figure 4: Class diagram for the proposed *TraceBasedPropagationLossModel*.

While in a real node the wireless card reports the RSSI – an indicator that greatly depends on the Auto-Gain Control (AGC) being applied on the RX circuit – ns-3 uses the actual (theoretically) calculated Received Signal Strength (RSS). Note that the AGC problem that affects the RSSI also affects the Noise Floor reported by the real wireless cards. Due to this, when reproducing the radio link quality in ns-3, **the proposed *TraceBasedPropagationLossModel* considers the SNR of the real system**. We can reproduce such real-world SNR by setting a user defined RSS in ns-3 above the ns-3 noise floor.

The SNR is used in the *ErrorRateModel* to calculate the probability of receiving a frame with an error and dropping it. The *ErrorRateModel* used is the *NistErrorRateModel* which is considered in the ns-3 documentation to be a more realistic one for OFDM modulations, compared to the "YansErrorRateModel".

The new *TraceBasedPropagationLossModel* considers the source and destination of the frame to define an RSS that will translate into an SNR equal to the captured in the real traces. To test this concept, the *TraceBasedPropagationLossModel* was developed to support a wireless connection between two nodes – a UAV and a Ground Fixed Node – and to be capable of considering different RSSs depending on the source of the frames. To make this possible, three *Attributes* were defined: 1) *RssFixedNode* – representing the current RSS (in dBm) for the frames received in the Ground FixedNode from the UAV; 2) *RssMobileNode* – representing the current RSS (in dBm) for the frames received in the UAV from the Ground Fixed Node; 3) *FixedNodePosition* – representing the coordinates of the node considered to be the Ground Fixed Node. These three Attributes can be set during the simulation execution via *Config::set* or via each related "set" method (see Figure 4).

The current implementation addresses only air-ground communication scenarios, i.e., one of the nodes must have a static position. Note that, from ns-3 execution context, when a *PropagationLossModel* is called to calculate the RSS for a link (see *doCalcRxPower* in Figure 4), only two *MobilityModels* pointers are received. These are used to obtain the sender and destination nodes positions (link distance), but do not have references for the nodes themselves. Therefore, the *FixedNodePosition* is a workaround employed to identify which of the two nodes is sending the frame – by matching the FixedNode position to the sender or receiver – and define the correct RSS (*RssFixedNode* or *RssMobileNode*) accordingly. The next version of the *TraceBasedPropagationLossModel* will be able to set the correct RSS values for scenarios with more than two nodes, without the workaround currently used.

2.5 Trace-based Simulation Settings

Before running the ns-3 simulation, the user should consider the following settings to better represent the real-world experiment:

- **TX Power End and TX Power Start** – No need to alter the default values for minimum and maximum TX Power. This is not considered when using the *TraceBasedPropagationLossModel*.
- **TX Gain** – No need to alter. It will not be considered as the resulting TX Power will be overridden by the *TraceBasedPropagationLossModel*.
- **RX Gain** – Should be kept as "0". If not, it will add to the RSS from the real trace, as explained previously.
- **WiFi Standard** – Set the WiFi Standard used in the real-world experiment, for instance, "802.11g".
- **Frequency** – Set the channel frequency used in the real-world experiment, for instance, "739MHz", if we use TVWS.
- **Channel BW** – Set the channel bandwidth used in the real-world experiment, for instance, "5MHz".
- **Propagation Delay** – The "ConstantSpeedPropagationDelayModel" should be used.
- **Propagation Loss** – Select the new "TraceBasedPropagationLossModel".
- **Error Rate Model** – The "NistErrorRateModel" should be used.
- **Remote Station Manager** – For UAV networks, we often use a constant rate "ConstantRateWifiManager".
- **Data Mode** – The constant rate used for data packets, for instance, "ErpOfdmRate6Mbps".
- **Control Mode** – The constant rate used for control packets, for instance, "DsssRate1Mbps".
- **WiFi Mac** – Set wifiMac type, usually "AdhocWifiMac".
- **Mobility Model** – Configure the Ground FixedNode with a "ConstantPositionMobilityModel" and create the WayPoints and use the "WayPointMobilityModel" for the MobileNode.
- **FixedNodePosition attribute** – Set the position of the Ground Node. It is accessible by *Config::Set*.
- **RssFixedNode** – Events should be scheduled throughout the simulation to reflect the correct value over the simulation time, according to the real trace values. It is accessible by *Config::Set*.
- **RssMobileNode** – Events should be scheduled throughout the simulation to reflect the correct value over the simulation time, according to the real trace values. It is accessible by *Config::Set*.

3 PROPOSED APPROACH VALIDATION

To validate our approach, two independent tests were performed. The first test aimed at validating the new *TraceBasedPropagationLossModel*; the second test was considered to assess whether a trace based ns-3 simulation is more accurate than a plain ns-3 simulation when compared with the results obtained experimentally.

3.1 Validation of the *TraceBasedPropagationLossModel*

For validating the new *TraceBasedPropagationLossModel*, a simple simulation scenario was created containing two nodes that stayed in static positions throughout the simulation. Each node had a Wi-Fi 802.11b/g interface configured in AdHoc mode, and operating at 739 MHz with a channel bandwidth of 5 MHz. Node 0 is set as the GroundNode and Node 1 acts as the MobileNode. The *TraceBasedPropagationLossModel* was configured with the *FixedNodePosition* equal to the Node 0. The initial simulation RSS values were set to -50 dBm for both nodes. The MobileNode (Node 1) ran an UdpEchoServerApplication and the GroundNode (Node 0) ran an UdpEchoClientApplication. The GroundNode sends UDP packets with 1400 bytes to the MobileNode at a rate of 1 packet per second. The packet capture option was enabled in both nodes. The RSS values were Scheduled to change 3 times, for both nodes, at different time periods during the same simulation, as follows:

1 Node 0 RSS = -50.0 dBm and Node 1 RSS = -80.0 dBm;
2 Node 0 RSS = -60.0 dBm and Node 1 RSS = -81.0 dBm;
3 Node 0 RSS = -70.0 dBm and Node 1 RSS = -82.0 dBm.

This simulation scenario is defined in file "first_scenario.cc", available for download in [4].

After running the simulation, two output *.pcap* files generated by ns-3 were obtained – one for each node – with *RadioTap* header included. By opening those files in *Wireshark* we got the data represented in Figure 5, Figure 6, and Figure 7. In the three selected time periods, the RSSI values reported for each frame had the same values that were defined by the trace. Therefore, it is proven that the new *TraceBasedPropagationLossModel* is working as expected.

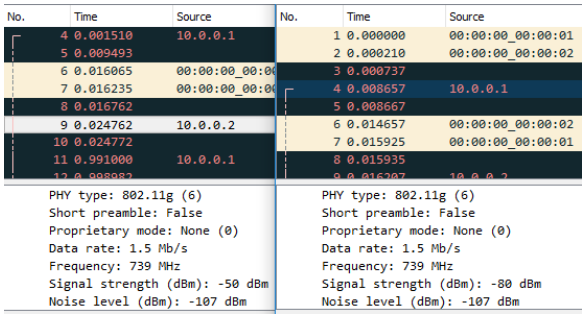

Figure 5: Wireshark screenshot #1 showing RSSI for Node 0 and Node 1 after the first Scheduled update.

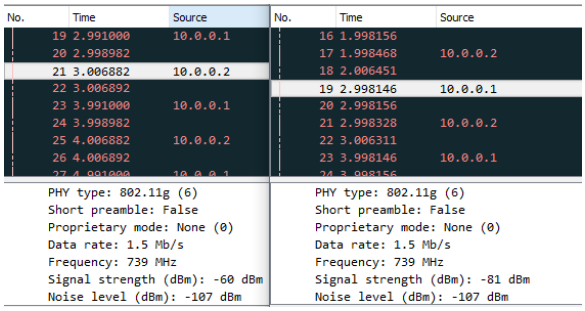

Figure 6: Wireshark screenshot #2 showing RSSI for Node 0 and Node 1 after the second Scheduled update.

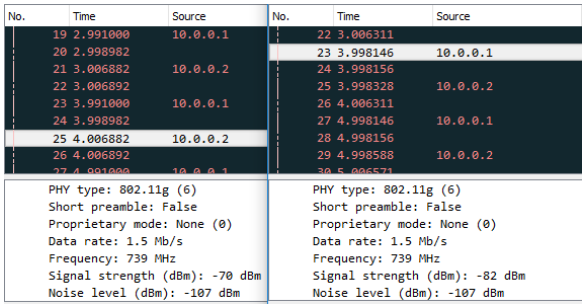

Figure 7: Wireshark screenshot #3 showing RSSI for Node 0 and Node 1 after the third Scheduled update.

3.2 Validation of the Trace-based ns-3 Simulation

We gathered traces of the positions and radio link quality obtained in a real-world experiment performed within the context of FP7 SUNNY project [10], which ran approximately for 2000 s. In this experiment there was a static node on the ground – the Base Station (BS) – and a UAV. The BS was positioned near shore in a elevated control room (50 m above the sea level + 5 m antenna mast) that assured good radio propagation conditions with a clear Fresnel ellipsoid for the radio frequency used (739 MHz). The UAV altitudes ranged from near 30 m during launching operations, and 100 to 600 m during flight operations. Figure 3 shows the path of the UAV during flight, while Figure 8 shows the radio link distances between the UAV and the BS along the real-world experiment. Both nodes were transmitting at 25 dBm. In the UAV a 2 dBi dipole antenna was used; the BS had a 6 dBi dipole antenna. The network link was loaded primarily with TCP traffic, but also with some residual UDP (Real-Time Protocol used by some systems) traffic generated by UAV on-board equipment communicating with the control systems installed at the BS. During the real-world experiment the UAV position and radio link quality (Figure 9) were collected, in order to reproduce the experiment in ns-3. The network throughput was measured along the experiment.

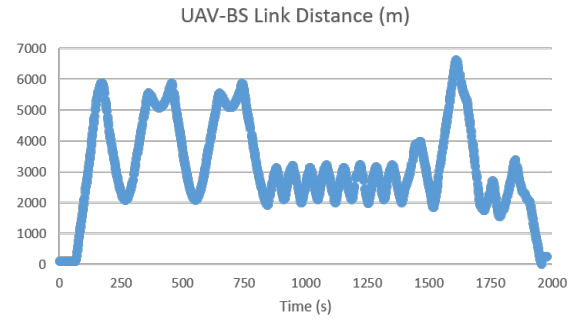

Figure 8: Radio link distance between UAV and BS.

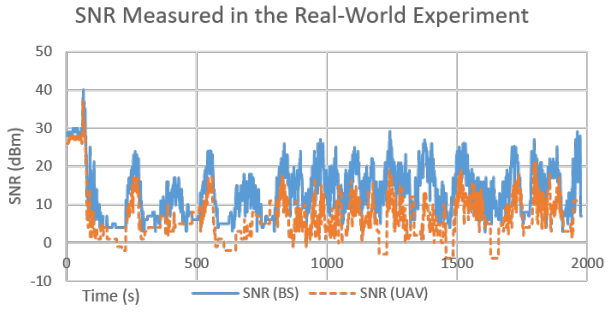

Figure 9: SNR recorded in the real-world experiment for the UAV and BS.

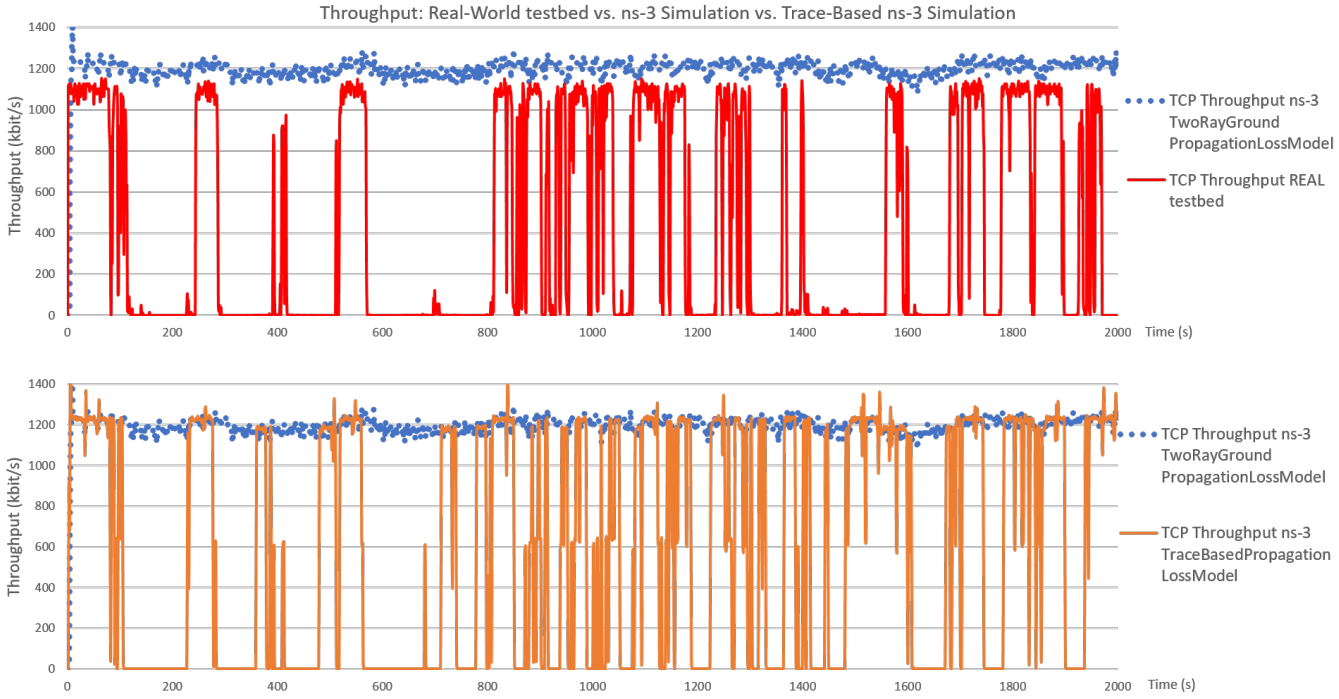

Figure 10: Real Throughput vs. 2-Ray Ground Model Simulation vs. Trace-Based Simulation.

Firstly, a plain ns-3 simulation was executed considering the node positions, channel frequency, bandwidth, transmission power, physical rate, and Wi-Fi standard used in the real-world experiment. This simulation scenario is described in file "second_scenario.cc" available for download in [4]. The BS node (Node 0) was configured with a *ConstantPositionMobilityModel* using the real coordinate, and the UAV node (Node 1) was configured with the *WaypointMobilityModel* that replicated the real positions of the UAV. The *PropagationLossModel* used was the *TwoRayGroundPropagationLossModel*. A TCP traffic flow was generated with the *OnOffApplication* from the UAV to a sink installed in the BS node. The TCP packet Maximum Segment Size (MSS) was set to 1400 bytes – in order to replicate the traffic observed in the real-experiment, where most of the packets were large and only limited by an MTU of 1500 bytes – and the Constant Bit-Rate (CBR) value was set to 2000 kbit/s, which is enough to keep the link fully loaded. For each simulated second, the throughput at the sink application was measured in 5 different simulation runs and the average value was calculated. The results are shown in Figure 10. The dotted line corresponds to a moving average using 2 consecutive samples of the simulated results, while the solid line of the upper plot represents the real-world values. As we can see, **the results are far from being realistic, as in ns-3 simulation not even once the link was broken between the two nodes**. This clearly shows the problem that we are addressing in this paper.

Secondly, the same ns-3 scenario based on the real UAV positions was executed, but this time using the real-world SNR traces via the proposed *TraceBasedPropagationLossModel*. This simulation scenario is described in file "third_scenario.cc" available for download in [4]. For each simulated second, the throughput at the sink application was measured in 5 different simulation runs and the average value was calculated. The results are presented in Figure 10 in the solid line of the bottom plot, using a moving average of 2 consecutive samples. We can see that the trace-based approach achieves a network throughput much closer to the one measured during the real-world experiment. Yet, the maximum real-world throughput does not reach the maximum value measured in simulation. This may be related to the fact that CSMA/CA contention can be triggered by noise, which is abundant in this real-world experiment and reached -68 dBm on the UAV side, lowering the MAC efficiency.

On the other hand, in some time-slots in the trace-based ns-3 simulation, the nodes were able to communicate while in the real scenario they were not (e.g., $t = 1500\ s$), which can be related to three other factors: 1) while the RSSI varies per packet, the noise floor takes longer to be updated in a real network interface, which could result in an outdated reported SNR during some time periods; 2) the TCP Retransmission Timeout (RTO) of the real-world applications could be out of sync with ns-3 RTO at that moment. If a TCP ACK repeatedly fails to be delivered to the sender, the amount of time to wait to retransmit it again keeps increasing, and it is difficult to reproduce in ns-3 the same TCP state that occurred in the real world TCP application; 3) the TCP Data Retries is other configurable parameter that can be different between ns-3 and the real-world TCP applications. In the real trace, there were moments where a TCP connection was closed near the $t = 1500\ s$, and it had to be manually restarted later on at $t = 1550\ s$, hence the big difference

between the real trace and the trace based simulation plots in that specific time interval. These aspects will be part of our future work, in order to improve the accuracy of the proposed trace-based ns-3 simulation approach.

Even though the trace-based simulation did not attain a perfect match with the real-world experiment, its realism is much better when compared to a plain ns-3 simulation, and it was able to reproduce in simulation the link instability problems encountered in the real-world experiment.

4 CONCLUSIONS AND FUTURE WORK

Real-world experiments based on emerging mobile testbeds are increasingly complex, costly to maintain, and hard do replicate. Our goal was capturing the real-world testbed conditions and perpetuate such experiments using trace-based ns-3 simulations. Our approach combines the ns-3 TCP/IP and MAC simulation capabilities with the physical characteristics of the real-world scenario provided by real traces of node positions and radio link quality.

We implemented the *TraceBasedPropagationLossModel* which helped validating our approach using traces of a real-world mobile testbed, producing results much closer to the real scenario than using plain ns-3 simulation. The trace-based ns-3 simulation approach enables: 1) concurrent user access to the real testbed conditions based on past traces; 2) running simulations in faster than real time; 3) running multiple simulation instances at the same time, exploring different variants of the solution under evaluation.

As future work, we will improve the *TraceBasedPropagationModel* to support scenarios with more than two nodes communicating in the same *Channel*. A helper will also be implemented to enable the automatic importation of the traces and scheduling of all necessary events in the simulator.

ACKNOWLEDGMENTS

The authors would like to thank the support from the Portuguese Foundation for Science and Technology (FCT) under the fellowship SFRH/BD/69051/2010.

This work is financed by the ERDF – European Regional Development Fund through the Operational Programme for Competitiveness and Internationalisation - COMPETE 2020 Programme and by National Funds through the Portuguese funding agency, FCT – Fundacao para a Ciencia e a Tecnologia within project POCI-01-0145-FEDER-016744.

This work is also part of the FP7 SUNNY project (id: 313243).

REFERENCES

[1] P. Agrawal and M. Vutukuru. 2016. Trace Based Application Layer Modeling in ns-3. In *2016 Twenty Second National Conference on Communication (NCC)*. 1–6. DOI: http://dx.doi.org/10.1109/NCC.2016.7561126

[2] BLUECOM+. 2017. BLUECOM+ – Connecting Humans and Systems at Remote Ocean Areas using Cost-effective Broadband Communications. (Feb. 2017). http://bluecomplus.inesctec.pt/

[3] G. Carneiro, H. Fontes, and M. Ricardo. 2011. Fast Prototyping of Network Protocols Through ns-3 Simulation Model Reuse. *Simulation Modelling Practice and Theory* 19, 9 (Oct. 2011), 2063–2075. DOI: http://dx.doi.org/10.1016/j.simpat.2011.06.002

[4] H. Fontes. 2017. Source Code for the TraceBasedPropagationModel – use in ns-3.26. (Feb. 2017). http://telecom.inesctec.pt/~hfontes/trace_based_propagation_loss_model2017.zip

[5] H. Fontes, R. Campos, and M. Ricardo. 2015. Improving ns-3 Emulation Support in Real-world Networking Scenarios. In *Proceedings of the 8th International Conference on Simulation Tools and Techniques (SIMUTools '15)*. ICST, Athens, Greece, 261–266. DOI: http://dx.doi.org/10.4108/eai.24-8-2015.2261074

[6] H. Fontes, T. Cardoso, and M. Ricardo. 2016. Improving ns-3 Emulation Performance for Fast Prototyping of Network Protocols. In *Proceedings of the Workshop on ns-3 (WNS3 '16)*. ACM, Seattle, WA, USA, 108–115. DOI: http://dx.doi.org/10.1145/2915371.2915374

[7] ns-3. 2017. ns-3 Home Page. (Feb. 2017). http://www.nsnam.org

[8] ns-3. 2017. ns-3.26 Emulation Overview – Model Library. (Feb. 2017). https://www.nsnam.org/docs/release/3.26/models/html/emulation-overview.html

[9] P. Owezarski and N. Larrieu. 2004. A Trace Based Method for Realistic Simulation. In *2004 IEEE International Conference on Communications (IEEE Cat. No.04CH37577)*, Vol. 4. 2236–2239. DOI: http://dx.doi.org/10.1109/ICC.2004.1312915

[10] SUNNY. 2017. SUNNY - Smart UNattended airborne sensor Network for detection of vessels used for cross border crime and irregular entrY. (Feb. 2017). http://www.sunnyproject.eu/