AERONAUTICAL CHANNEL MODELING FOR PACKET NETWORK SIMULATORS

Author: Sandarva Khanal
Advisor: Dr. Richard A. Dean
Department of Electrical and Computer Engineering
Morgan State University

ABSTRACT

The introduction of network elements into telemetry systems brings a level of complexity that makes performance analysis difficult, if not impossible. Packet simulation is a well understood tool that enables performance prediction for network designs or for operational forecasting. Packet simulators must however be customized to incorporate aeronautical radio channels and other effects unique to the telemetry application. This paper presents a method for developing a Markov Model simulation for aeronautical channels for use in packet network simulators such as OPNET modeler. It shows how the Hidden Markov Model (HMM) and the Markov Model (MM) can be used together to first extract the channel behavior of an OFDM transmission for an aeronautical channel, and then effortlessly replicate the statistical behavior during simulations in OPENT Modeler. Results demonstrate how a simple Markov Model can capture the behavior of very complex combinations of channel and modulation conditions.

KEY WORDS

iNET, Aeronautical Channel, Markov Model, Hidden Markov Model, OPNET Modeler.

I. INTRODUCTION

The Aeronautical channel poses its own set of unique challenges during simulation. Aircrafts moving at the speed of up to 2 Mach are exposed to environmental conditions that can change dynamically and rapidly. Since ground and aircraft antennas are non-directional, multipath signal distortion is the main source of disturbances for iNET system. Also, under such dynamic conditions, other factors such as Doppler Shift and fading/shadowing may simultaneously contribute to the signal degradation at times, while randomly remaining passive at other instances [1]. Therefore, depending upon the channel condition, terrain, weather and other environmental aspects, aeronautical channel can bring abrupt and drastic changes in the signal level. For this reason, any developmental progress made in modeling iNET protocols and nodes cannot be truly verified unless the aeronautical channel, that these nodes will be tested under, has been successfully implemented. The success of iNET simulation thus depends on the successful

implementation of the unique effects introduced by aeronautical channel, hence emphasizing aeronautical channel simulation as a critical part of iNET simulation. However, this component is not present in commercial simulators such as OPNET Modeler, thus highlighting the significance of the work proposed in this paper in the overall success of iNET.

In packet simulators such as OPNET actual physical layer features are replaced by statistical models such as the Markov Model. OPNET does not include aeronautical channel models and so it is important to create this feature for our simulations. Previous work, done in modeling the aeronautical channel for iNET, included incorporating MM driven calculations to randomize SNR output during simulations [1]. This was an initial step to prove that any default Radio Pipeline Stages for wireless communication could be modified to add custom features as needed. The problem in this simulation however, was that the input parameters that were fed into the MM were random and required verifications quantitatively and qualitatively. In order to solve this problem, and to be able to recreate more realistic channel behavior during simulation, this paper proposes using the HMM to learn from real channel data and extract the features necessary to feed into MM within the customized OPNET SNR pipeline stage. This paper begins with an overview on MM and HMM, followed by the approach adopted in this project. The HMM algorithms were validated before using them for feature extraction. This paper ends with the discussion of the simulation scenario as well as the results used to conclude that using HMM and MM, we could effortlessly replicate the statistical behavior of an aeronautical channel during simulations in OPNET Modeler.

II. MARKOV MODEL

Markov Model (MM) is a stochastic model in which a system can move from one state to another following the Markovian Property, which says that the probability of the system moving to a certain state at time (t+1) depends only on the state of the system at time t, and not on any other previous states or pattern of states [2], [3]. The system must be in any of the defined states at any given time, so that the probability that the system will be in any of the defined states is 1 [3]. When the state changes, it means that it only transitions from one of the defined states to another defined state or bounce back to same state. This behavior of MM can also be represented using a Finite State Machine (FSM) structure. To fix ideas, a simple 3-state MM is illustrated in Figure 1 which depicts an FSM with arrows indicating the probability of system transitioning to another state or to itself in next time-slot.

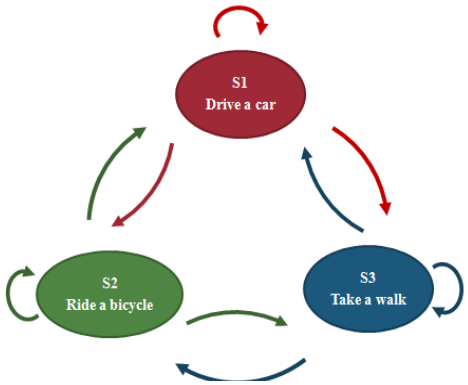


Figure 1: A Simple 3-State Markov Model

A. States

Suppose a system have three states S = {S_1, S_2, S_3}, each representing the observations for a person to either drive a car, or ride a bicycle, or take a walk respectively. The system starts in one of these states and moves successively from one state to another. If the system is currently in state S_i, then it moves to state S_j at the next step with a probability denoted by A_{ij}, and this probability does not depend upon which states the chain was in before the current state. This is the basis of forming a Markov Chain.

B. Initial State Probabilities and Transition Probabilities

A particular state is randomly chosen as the starting state. This is usually done using an initial state probability distribution (π) matrix. For any Markov system with N number of states, the π-matrix will be of size [1 x N]. Therefore, for a 3-state MM, π-matrix will be denoted as

$$\pi = [p_1, p_2, p_3], \quad (1)$$

Where, p_i = probability of the system starting from state i.

The probabilities A_{ij} discussed in previous section are called transition probabilities. In the above figure, all the arrows indicate the transition probabilities. Also, the process can remain in the same state it is in, and this is given by transition probability A_{ii}. For ease, these transition probabilities are also arranged in a matrix, called the State Transition Matrix or simply A-matrix. For any Markov system with N number of states, the A-matrix will be of size [N x N]. Therefore, for a 3-state MM, A-matrix will be denoted as

$$A = \begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{bmatrix} \quad (2)$$

Where, A_{ij} = probability of system transitioning from state i (time t) to state j (time t+1) [2], [4].

C. Simulation using Markov Model

In modeling a Markov Model in computer simulations, we introduce randomness in system transitions. The randomness begins from the starting (initial) state of the system, and is inhabited throughout the simulation using Markovian transitions. Therefore, using only π and A-matrices, a complete modeling of MM can be accomplished during computer simulations. After the initial state has been chosen using π-matrix, the A-matrix will dictate the next random states:

$$\text{Next Probable State} = \pi * A \quad (3)$$

III. HIDDEN MARKOV MODEL

A Hidden Markov Model (HMM) is an extension of Markov Model in which the system being modeled is assumed to be a Markov process with unobserved (hidden) states [2][3][4]. In a regular Markov Model, the state is directly visible to the observer, and therefore the state transition probabilities are the only parameters. In a Hidden Markov Model, the states are not directly visible, but outputs, dependent on each state, are visible. Each state has a probability distribution over the possible output tokens. Therefore the sequence of tokens generated by an HMM gives some information about the sequence of states. Hence, an HMM model requires an additional parameter called Emission Probability for complete mathematical representation. HMMs can be represented by the following useful compact notation:

$\lambda = (\pi, A, B)$ (4)
Where,
π = [1xN] initial state distribution vector $\pi = \{\pi_i\}$
A = [NxN] state transition probability distribution given in the form of a matrix $A = \{a_{ij}\}$
B = [NxM] observation symbol probability distribution given in by $B = \{b_j(k)\}$

A Hidden Markov Model is "hidden" because the parameters of the model are not initially known [3]. The model has to be "trained". The previous section demonstrates that a discrete-time, discrete-space dynamical system governed by a Markov chain emits a sequence of observable outputs: one output (observation) for each state in a trajectory of such states. One can infer the most likely dynamical system from the observable sequence of outputs. The result is a model for the underlying HMM process. To fix ideas again, the MM model discussed in previous section is illustrated with addition of the hidden states.

A. States and Observations

The FSM system presented in the previous section can be presented using the HMM. In addition to the original three observations for a person to either drive a car, or ride a bicycle, or take a walk respectively, the HMM will consist of some hidden states. If there were four states (sunny day, rainy day, foggy day and cloudy day respectively), the FSM would look like one in Figure 2. The FSM now has four different states (indicated by the white circles), and three possible observations for each state (indicated by the dark circles). The states provide the intuition as to what the weather condition is in the given time, while the observation shows what the person is most likely to do (drive/ ride/ walk) for each weather pattern. Just by observing whether person A is walking, driving or riding a bike in New York as seen on a TV set at his home, person B living in California could make an inference as to what the weather is like in New York.

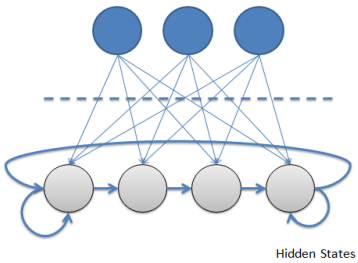


Figure 2: A 4-State Hidden Markov Model

B. Transition Probabilities and Emission Probabilities

Like in the MM, the probabilities of the system moving from one state to the other in an HMM are given in State Transition Matrix (A-Matrix). These probabilities are indicated by dark blue curved arrows in the figure, and are called transition probabilities. The addition of possible observations for each hidden state adds more complexity, and can be seen in the figure by the presence of straight blue arrows. These are the observation probabilities, or more commonly known as emission probabilities. The emission probabilities indicate the probability of observing certain emission, given the system is in some predefined state at the given time. The emission probabilities can be arranged in a matrix, known as Observation Matrix, or simply B-Matrix. For a system with N states and M different possible observations for each state, the A-Matrix will be of size [NxN], and B-Matrix will be of size [NxM]. Therefore, for the example given above where N = 4 and M = 3, the HMM parameters ($\boldsymbol{\pi}, \boldsymbol{A}, \boldsymbol{B}$) will look like the following:

$\boldsymbol{\pi} = [P_1\ P_2\ P_3\ P_4]$

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix}$$

$$B = \begin{bmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \\ b_{31} & b_{32} & b_{33} \\ b_{41} & b_{42} & b_{43} \end{bmatrix}$$

C. Answers to Three Problems (using HMMs)

The most important characteristic of an HMM is that it is able to reduce a complex system described by many observations, into a simple MM with fewer states. Given a sequence of outputs observations, HMM can infer the most likely sequence of states. HMM can also be use to model or predict the next observation or more generally a continuation of the sequence of observations. Looking back to the above example, if Person B were to know of Person A's observations, but nothing about how the weather changes (transition probabilities) and the observation probabilities, how would Person B model the weather in Person A's location. Can he even be able to do so? With a HMM, Person B should be able to model the weather in Person A's location just off of observation data made by person A. In general, HMM can solve three important problems; Evaluation Problem, Decoding Problem and Optimization Problem.

Evaluation Problem: Given the observation sequence and the model, what is the probability that the observation sequence was produced by the model?

Decoding Problem: Given the observation sequence and the model, what is the most probable sequence of hidden states?

Optimization Problem: Given a sequence of observation, how can the model parameters be optimized? I.e. find $\lambda = (\pi.A.B)$.

Each of the above problems is solved with a different algorithm. The "forward-backward algorithm" can solve the first problem addressed by HMM. The output of the "Viterbi algorithm" is called Viterbi path, and is the solution to the second problem solved using HMM. The most important and difficult problem that can be solved using HMM is the Optimization Problem. Studies about Hidden Markov Model have demonstrated that "Baum-Welch algorithm" can be used to find optimized Markov Model parameters. The underlying math for each algorithm is beyond the scope of this paper. Detailed algorithms for each solution can be found in [2] and [4].

IV. PROPOSED WORK

The key improvement in this project involved extensive use of all three algorithms, especially the Viterbi algorithm and the Baum-Welch algorithm, to decode the training data and optimize the MM parameters. The overall approach for this project is shown using the block diagram in Figure 3. As seen in the diagram, the work was divided into four blocks. First, SNR outputs from real OFDM data transmission was taken as training data to feed into HMM. Second, an HMM toolbox was to be embraced, capable of patterns recognition, to extract channel behavior from the training data. These first two parts were done in the Matlab platform (top half of the figure).

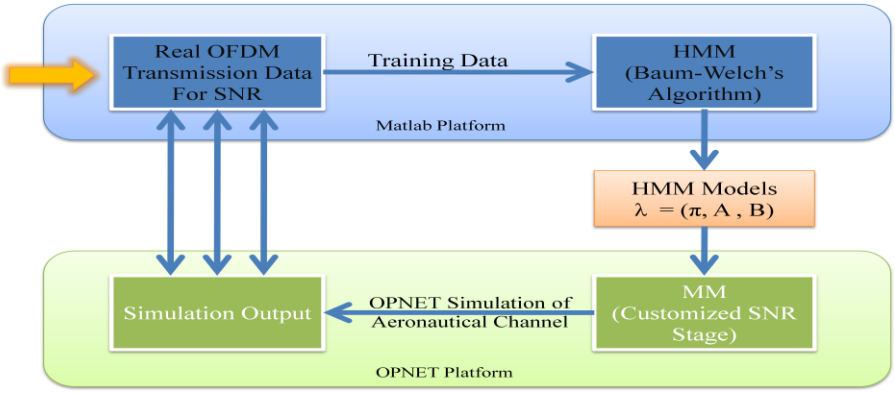


Figure 3: Illustration of Approach for the Project

The third step was to replace the MM model in the customized SNR stage that would now be able to intake all three parameters obtained from HMM learning. SNR stage was chosen because the HMM would be learning from training data which was SNR output from actual OFDM transmission. In the final step, comparison of SNR patterns in original training data and the OPNET simulated output should show similarity between the training data and the reproduced sequence. By doing so, it could replicate the propagation properties of aeronautical channel in time, frequency and space, which is useful for the analysis and simulation of the aeronautical links. The last two parts were done in the OPNET Modeler platform (lower- half of the figure).

V. VALIDATING HIDDEN MARKOV MODEL

For ease in maneuvering this project, HMM toolbox developed at MIT was adopted to learn from the training data [5], [6]. This toolbox has all the features needed to accomplish our target for this project. Most importantly, it incorporates both the Viterbi Algorithm and Baum-Welch's Algorithm. However, before we could use it in our experiment to learn from real OFDM data, we needed to verify that the HMM toolbox works as expected. A Matlab function was developed that would take number of states (N), number of possible observations (O), and number of iterations (T) as input parameters. The function would then generate random HMM model parameters ($\lambda = \pi$, A, B), based on the inputs. This model would first generate a state vector of length T using Markovian Properties, and for each state, assign an observation from the set of O possible observations. By only using this observation vector as our training data, we try to estimate the original HMM model parameters that produced the training data. The Baum-Welch's Algorithm was used to learn the HMM parameters from the training data. A second set of observation data would then be generated using these estimated HMM model parameters. A comparison was done between two sets of HMM models, state vectors, and observation vectors (original, and estimated data set) to verify the HMM toolbox [6].

After several experiments over same training data, it was concluded that the HMM toolbox worked better under two conditions: first, the toolbox gave better results when the number of states (N) was guessed to be 5, to make the initial guess parameters. Second, when the diagonal elements of the A-matrix used as initial guess parameter were reinforced, and the matrix was renormalized before using it in EM procedure, the Baum-Welch's Algorithm performed faster, as well as yielded better output observation sequence. These conditions suited in our case because

we would be using the HMM toolbox to train from real OFDM data, which would have several observations affected by noise, Doppler shift, multipath and some other external factors. So, it is harmless to assume there were 5 channel states during the flight, and possibly multiple observations while in same state. Also, it can be presumed, that when the aircraft entered any state, it would tend to remain in that state for a while before jumping to other states. The reinforced diagonal A-matrix would ensure that this practical trend is mimicked in simulations.

Figure 4 shows a section of Viterbi path obtained by estimation made on the training data, using 5 as possible number of hidden states. The plot reveals how the system made random transitions from one state to the other, but following some patterns. The patterns were that the system tended to remain in any one state for a longer time, and that it spent more time on lower states (states 1-3) than in higher states (states 4-5). This can be attributed, to the fact that the diagonal of the A-matrix was reinforced as explained in the above two conditions.

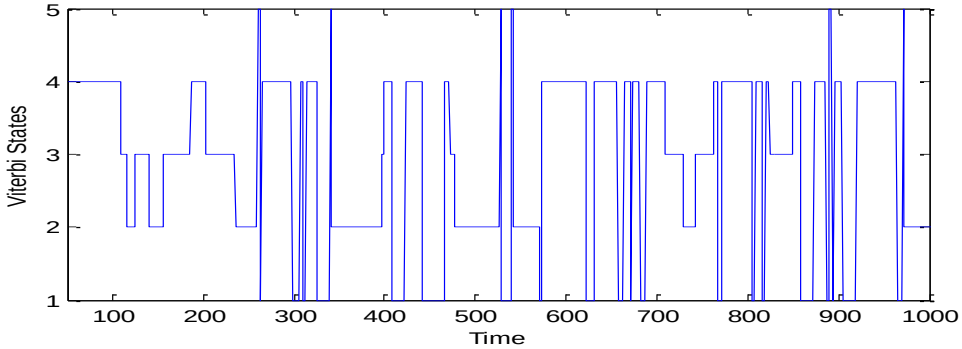


Figure 4: State Transitions (Viterbi Path) for 5-State System

Figure 5 illustrates the histogram comparisons of original and estimated states sequences (L) as well as observations (R). The histogram comparison of state sequence reveals there were three states in total that produced the training data. Since estimated number of states was entered as 5 before creating the initial guess parameter and then using thus obtained parameters to estimate the Viterbi path, the histogram of Viterbi path correctly determined all 5 states. Obviously, since the number of states did not match in the two cases, their histograms also mismatched.

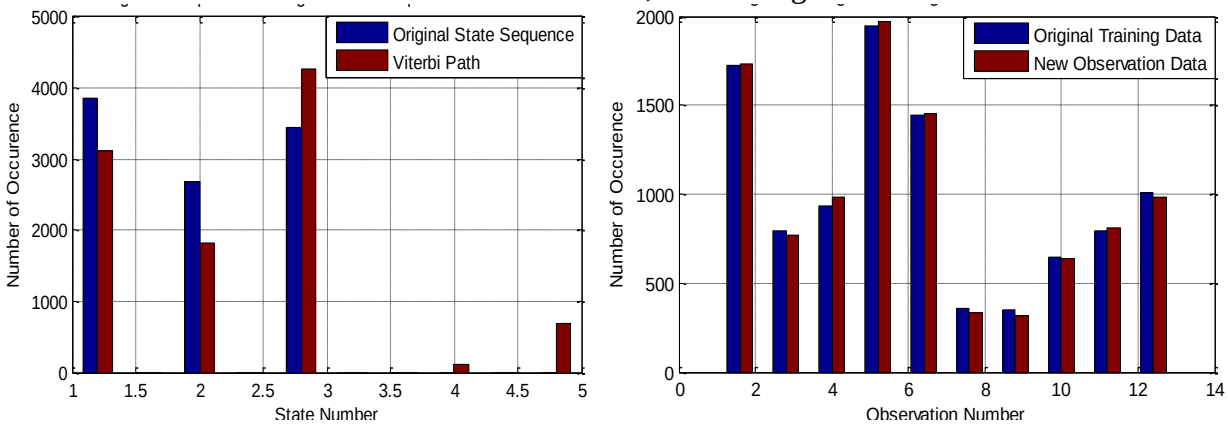


Figure 5: Histogram Comparison of Original vs. Estimated State (L) and Observations (R)

It can be observed that there were 10 different observations throughout the training data, and the HMM correctly identifies all 10 of them, as well as the counts for each observation. It is to be noted that the regenerated observation in this case did not yield exact match to the training data,

but the overall pattern was followed as shown in the histogram. This shows that although the HMM input parameters and states did not match that to the output, observation data and training data were following similar patterns. In modeling of random processes like aeronautical channel, successful modeling of these observation patterns is of more importance than the hidden states sequence themselves. This verified the authenticity of the HMM toolbox we adopted.

Now that the HMM toolbox was verified, this toolbox could be used to read any training data set and it would provide the most likely HMM parameters that can closely describe the data using simple mathematical mode. This meant that a complex set of aeronautical effects that would otherwise be difficult to explain and recreate, could now be easily reduced to a set of matrices, and these matrices could be used during simulations to recreate similar patterns. The MM that would take all the parameters given by HMM toolbox was also verified in reproducing similar observation data with patterns. The SNR pipeline stage in radio communication mechanism within OPNET Modeler was modified to be able to incorporate the new parameters given by HMM toolbox. Now, the only task remaining was to simulate a scenario in OPNET and verify that the aeronautical effect was working as expected during simulations.

SIMULATION SCENARIO AND RESULTS

A scenario was created in OPNET Modeler comprising of a TA and GS prototype. Both the TA and GS are yet to be completely designed for iNET, so the ones used in this simulation were defined with only the basic components. The TA had a source module, a queue, a transmitter, and one transmitting antenna. The TA was assigned a trajectory (white curved arrow in Figure 6), and would travel from Saint-Louis towards Denver at the average speed of 3 Mach. The average height of the TA throughout the simulation was 25,000 m. The total length of trajectory was 1144.70 km, and the TA would complete the path in about 12 minutes and 43 seconds. Constant Bit Rate (CBR) was used for packet creation. The transmitter is operating at a base frequency of 1GHz, with a bandwidth of 5000 KHz, which produces a data rate of 5 Mbps. This is the maximum data rate that can be archived [1]. The transmission power allocated to packets transmitted through the channel is set to be 1 watt. The antenna offers a transmitter Gain of 1db.

The GS was placed near the city of Tulsa, mid-way through the trajectory. The GS had three modules: receiving antenna, a radio receiver, and a sink where the packets would be destroyed. The receiver Gain is set at 15db. It is done by customizing the GS antenna which is derived from the isotropic antenna model. A 15db Gain is assigned to every direction of an isotropic antenna. The ECC threshold was set to be 0.2, meaning that packets with 20% or more bits in error would be dropped. All the models and parameters used in this project were adopted from [1].

The 5-stage MM parameters, obtained as outputs of HMM by training from real OFDM data, were used to mimic the aeronautical channel effect during simulation. Figure 6 shows three snapshots taken during a simulation that used the new SNR stage customized for this project. The three figures indicate the status of packet reception at three different time slots. Figure (a) indicates that the transmission began when the TA just started to move towards the GS. Figure (c) indicates that the GS was successfully receiving packets transmitted by the TA as the TA was moving away from the GS. The TA is closer to the GS in figure (b), as compared to the other two

figures. Yet, it is at this time when a packet was dropped due to "reception" fault, as indicated by the red-cross mark on the packet. This pattern of "dropped packet" was not observed while simulating with default SNR stage. When the customized SNR stage was used on the receiver, the unique aeronautical effects caused the signal level to drop at random times and the packets got dropped. Therefore, the unique aeronautical effect was successfully simulated.

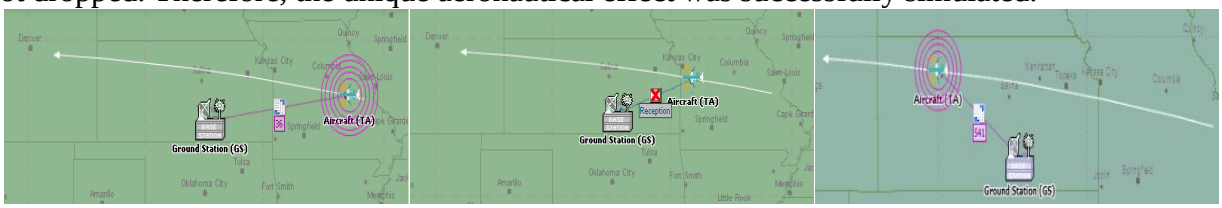
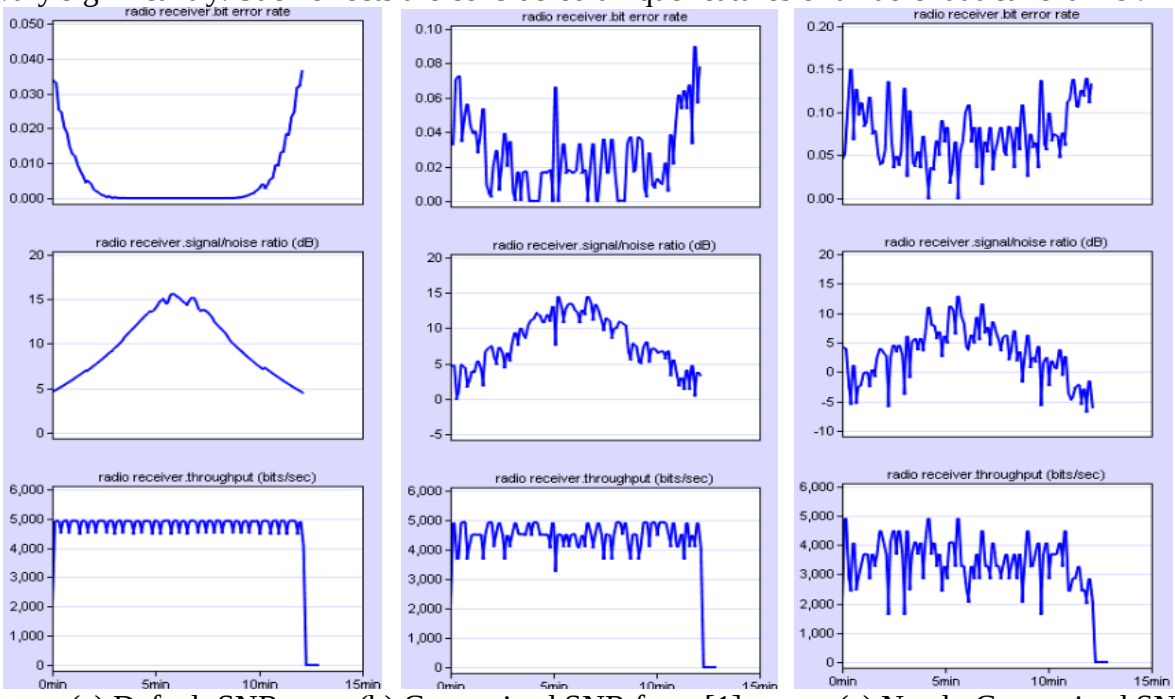


Figure 6 (a), (b), (c): Aeronautical Channel Simulation

To verify whether this simulation gave the desired randomness in output while following some state structure, three simulations had to be run for the same scenario and the results had to be compared. Figure 7 shows plots for bit error rate, signal-to-noise ratio, and throughput at the receiver for the three simulations. 7 (a) shows the results obtained when the default SNR stage was assigned to the receiver. In the second run, shown in (b), the SNR stage customized in [1] was used to simulate aeronautical channel. The third run, shown in (c), used the SNR stage customized in the build-up to this paper to simulate aeronautical channel. Figures 7 (b) and (c), both taken over MM driven aeronautical channel, show that the MM successfully created an aeronautical effect by introducing randomness in signal reception during simulations. However, the major difference in these two charts is that the plots on (c) yielded more randomness and less predictability in the results, which is more realistic representation of an aeronautical channel. Due to Doppler Shift, multipath, shadowing and test article status, the channel quality may drop or vary significantly. Such effects are considered unique features of an aeronautical channel.



(a) Default SNR (b) Customized SNR from [1] (c) Newly Customized SNR

Figure 7: Comparison of OPNET Simulation Results Obtained using Different SNR Stages

CONCLUSION AND FUTURE WORKS

The simulation results showed that the MM which was incorporated in OPNET by modifying the SNR pipeline stage was able to recreate the unique behaviors attributed to the aeronautical channel. This also proved that using HMM and MM, we could train from real data and mimic all the uniqueness posed by complex aeronautical effects during simulations. More importantly, the results proved that we could actually reduce the overall complexity of the problem into a simple mathematical model. If all the uniqueness within the physical layer were to be modeled by mathematical calculations, the simulation would have been computationally expensive and time consuming. Calculating individual phenomenon at random times, and erratically ignoring them at other instances could bring unrealistic results during simulations. By using simple mathematical and probabilistic model instead, the results demonstrated that we could reproduce the randomness, while maintaining patterns similar to that found in real life.

The next step in our research is to develop various sets of HMM models for all possible channel conditions, and use them accordingly to profile different situations during iNET simulations. This will allow us to easily represent various channel conditions using simple matrices, and perform simulation of aeronautical channel utilizing less computation, yet with parallel accuracy.

ACKNOWLEDGEMENTS

I would like to thank our sponsors TRMC, SRC and CRC for their support of this work. I would like to also thank my advisor, Dr. Richard A. Dean, for his support leading to this paper and OPNET Technologies Inc. for providing software license to carry out the simulations.

REFERENCES

[1] Zhang, T, and Jaber, N. "Aeronautical Channel Simulation in Network Simulators for Incorporation into OPNET". ITC 2010 Conference, San Diego, CA, October 2010.

[2] Rabiner R, L. “A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition”, IEEE. 1989

[3] Kamirah K, Daniel. “Estimating Channel Parameters using a Hidden Markov Model”. Morgan State University Symposium. 2010.

[4] Shokhirev, Nikolai. “Hidden Markov Model by Nikolai Shokhirev” <http://www.shokhirev.com/nikolai/abc/alg/hmm/hmm.html>

[5] Murphy, Kevin. “Hidden Markov Model (HMM) Toolbox for Matlab”. <http://www.cs.ubc.ca/~murphyk/Software/HMM/hmm.html>

[6] Ornek, Masum. “Modeling a Wireless Channel with a Hidden Markov Model.” Senior Design Project Report, Morgan State University, Spring 2010.