**Lab 5: 802.11a Image Transmission and Reception**

|  |  |
| --- | --- |
| **Author** | Name: 孙逸涵 (Yihan Sun) Student ID: 12012128  Name: 张旭东 (Xudong Zhang) Student ID: 12011923 |
| **Introduction**  In this lab, we have learned how to use MATLAB to implement the 802.11a image transmission and reception. Firstly, we learned the data structure transmitted in 802.11a, i.e., how the data structure is changed from MSDU to PPDU. To have a better understanding of this encapsulation process, we use MATLAB programming to achieve it. Then, the encapsulated packets are sent to the receiver through AWGN channel. At the receiver, we can recover the packet data transmitted from the transmitter by capturing the data packets, processing data packets and reconstructing the image. In addition, we discuss the effects of different modulation schemes and SNR by comparing the performance of the constellation and the recovered image. Finally, we change the AWGN channel into the HiperLan/2 channel, which is commonly used in WiFi. We use five submodels to discuss the relationship between SNR and packet error rates under different models and the relationship between SNR and packet error rates under different modulation schemes, respectively.  **Theoretical analysis**   1. **Introduction to functions** 2. **ResizeImage.m**   ResizeImage.m is used to resize the transferred image. First, it reads the image file and its dimension. Then, it reassembles the image data into the size of the image that needs to be transferred.   1. **createPSDU.m**   createPSDU.m mainly implements the process of encapsulating binary image data stream into PSDU. In the process of encapsulation, we first define some MSDU parameters, and assemble a complete MSDU. For the last MSDU which has insufficient length, we need to complement 0 to it to construct a complete MSDU. Then we set some parameters of the frame dawn check sequence. By dividing the data into blocks and setting the number of bits required for the MPDU header, we encapsulate the MSDU into MPDU.  And we need to notice that:  Length of MPDU = Length of MAC header + Number of MSDU bits + Length of FCS  Then, we construct the image according to the MPDU format in a circular form and get the complete PSDU of the image.   1. **createTxWaveform.m**   createTxWaveform.m creates a WLAN packet for our transmission based on the PSDU we have passed in. In the process of creating the packet, we firstly determine the modulation scheme, i.e., set the value of MCS, the number of transmitting antennas, the value of bandwidth, and the other parameters. Then, we can generate a baseband Non-HT packet. In addition, we need to add the preamble field (L-STF, L-LTF, L-SIG) to the packet header of the Non-HT packet. And we can use inverse Fourier transform to obtain the waveforms of those training sequences in time domain. Finally, by resampling the WLAN transmit waveform and normalizing the signal, we can get the final WLAN packet that needs to be transmitted.   1. **createAWGNChannel.m**   createAWGNChannel.m is used to configure the AWGN channel. When configuring the channel, we will focus on determining the signal power, the signal to noise ratio (SNR), and the other parameters. From the SNR we set, createAWGNChannel.m can help us to calculate the SNR of the current channel, so that make the input signal waveform through the channel into the subsequent processing.   1. **ReceiverProc.m**   ReceiverProc.m is used to further process the received data packets for subsequent image reconstruction. In this process, we first need to obtain the position of the PSDU header, and then downsample the received signals. At the same time, we need to generate the corresponding FCS for MPDU unpacking. After that, we unpack the received packets. In the process of unpacking, we first need to detect the packet and adjust the offset in the packet. Then, we extract the Non-HT field of the data packet, and carry out corresponding coarse frequency deviation correction and symbol synchronization. After coarse frequency offset correction, we can carry out fine frequency offset correction. And we can also use the L-LTF part of PDSU header for channel estimation. Finally, we recover the L-SIG part and the parameters in the packet, and correct the entire packet with CFO. Besides, we use the results of the previous channel estimation to recover the bits in the PSDU and display the current constellation. We need to notice that we also need to remove FCS from the MAC header at the end to process the MAC information.   1. **reBuildImage.m**   reBuildImage.m is used to firstly calculate the bit error rate of the 802.11a image transmission and reception process. Then reconstruct the image, and display the recovered image.   1. **L-STF sequences in Non-HT format** 2. **Generating formulas**   L-STF, means L-STF(short training field), whose function including detection of packet arrival, coarse time synchronization, coarse frequency offset estimation and adaptive gain control. The formula is  Where   1. **Composition of sequences**   The frame structure of L-STF is    The first six subframes is used for energy detection and the last four subframes is used for coarse frequency offset correction. The total time of L-STF is     1. **Generation process of sequences**   The MATLAB code of generating the L-STF sequence is as below:    Firstly, a sequence of virtual subcarriers whose length is 11 is created. Then is inserted between the sixth element and the seventh element of virtual subcarriers to construct short preamble slot frequency. After that, do inverse Fourier transform for short preamble slot frequency to get short preamble slot time. Finally, take the first 16 elements of short preamble slot time and copy them for 10 times to obtain L-STF sequence by multiplying the copied sequence by 20.   1. **L-LTF sequences in Non-HT format** 2. **Generating formulas**   L-LTF, means L-LTF(long training field), whose function is channel estimation, fine time synchronization and fine frequency offset estimation. The formula is  Where   1. **Composition of sequences**   The frame stucture of L-LTF is    L-LTF consists of three parts, including cyclic prefix, OFDM symbol 1 and OFDM symbol 2. The duration of cyclic prefix is , which make convolution cyclic convolution. Cyclic prefix is the second half of OFDM symbol. The duration of OFDM symbol 1 and OFDM symbol 2 is and the second half of OFDM symbol 1 can ba treated as the cyclic prefix of OFDM symbol 2.     1. **Generation process of sequences**   The AMTLAB code of generating L-LTF sequence is as below:    The process of generating L-LTF sequence is similar to that of generating L-STF. Firstly, a sequence of virtual subcarriers whose length is 11 is created. Then L\_k is inserted between the sixth element and the seventh element of virtual subcarriers to construct long preamble slot frequency. After that, do inverse Fourier transform for long preamble slot frequency to get long preamble slot time. Finally, take the second half of long preamble slot time as cyclic prefix and concatenated with two long preamble slot time to obtain L-LTF sequence by multiplying the concatenated sequence.   1. **L-** **SIG sequences in Non-HT format**   L-SIG consists of 24 bits, which includes the rate, length, parity check and tail. The formula for generating L-SIG sequence is  The frame structure of L-SIG is    The rate, which contains 4 bits, represents modulation and coding scheme. Different sequences of bits represents different modulation and coding schemes. The relationship between the two is as follows.    The length field represents length of the PSDU in octets in the range of 1 to 4095. The P field represents parity check  **Lab results & Analysis**   1. **Method of generating training sequence of Non-HT format PPDU**     In this part, we borrowed Dr. Wu’s program and write a new one which is named as generateTrainingSeq.m to plot the waveform of the training sequence of Non-HT format PPDU generated by using wlanWaveformGenerator, the waveform of the training sequence of Non-HT format PPDU generated by using createSTF(S\_k) and createLTF(L\_k), and also the comparison between them.    In addition, we choose 5MHz channel bandwidth in this part, so the duration of the training sequence is .   1. **Using wlanWaveformGenerator**       The figure shown above is the waveform of the training sequence of Non-HT format PPDU generated by using wlanWaveformGenerator.   1. **Using createSTF(S\_k) and createLTF(L\_k)**       The figure shown above is the waveform of the training sequence of Non-HT format PPDU generated by using createSTF(S\_k) and createLTF(L\_k).   1. **Verification of waveform consistency**       From the figure shown above, we can verify that the waveform of the training sequences of Non-HT format PPDU generated by using wlanWaveformGenerator and using createSTF(S\_k) and createLTF(L\_k) are almost the same and have only a little difference of amplitude at some point.   1. **receiverProc function signal processing flowchart**   The signal processing flow chart of **receiveProc** function is as below    First, information contained in PSDU should be obtained, including L-STF, L-LTF and L-SIG. The processing is done using **wlanFieldIndices** function. This function returns a structure, ind, containing the start and stop indices of the individual component fields that comprise the PPDU, given a format configuration object. It only supports generation of field indices for OFDM modulation. If **field** is specified, the function returns **ind** as a 1-by-2 vector consisting of the start and stop indices of the PPDU field.  The second step is to downsample received signal and generate FCS for MPDU. After that, computation of EVM and receive loop processing is to be done.  In the process of receive loop processing, the first step is to packet detect and adjust packet offset. Then the Non-HT domain is extracted to perform coarse frequency offset correction using **wlanCoarseCFOEstimate** function. This function returns a coarse estimate of the carrier frequency offset (CFO) given received time-domain L-STF samples and channel bandwidth. After that, symbol timing synchronization is performed.  Immediately after symbol timing synshronization, fine frequency offset correction is performed using **wlanFineCFOEstimate** function. This function returns a fine estimate of the carrier frequency offset (CFO) given received time-domain L-LTF samples **rxSig** and channel bandwidth **cbw**.  Then, using L-LTF to do channel estimation by using **wlanLLTFDemodulate** function and **wlanLLTFChannelEstimate** function. The **wlanLLTFDemodulate** function returns the demodulated L-LTF waveform given time-domain input signal **x** and channel bandwidth **cbw**. The **wlanLLTFChannelEstimate** function returns the channel estimate given channel bandwidth cbw. The channel bandwidth can be used instead of the configuration object. After doing channel estimation, noise estimation is performed.  Besides, L-SIG field is recovered using **wlanLSIGRecover** function. The **wlanLSIGRecover** function returns the recovered L-SIG information bits, **recBits** and the status of a validity check, **failCheck**, given the time-domain L-SIG waveform, **rxSig**. Specify the channel estimate, **chEst**, the noise variance estimate, **noiseVarEst**, and the channel bandwidth, **cbw**.  After that, parameters of packets is recovered and the whole packet is corrected using CFO.  What’s more, the bit sequence of PSDU is recovered with the result of channel estimation by using **wlanNonHTDataRecover** function. The **wlanNonHTDataRecover** function returns the recovered **Non-HT-Data field** bits and the equalized symbols, **eqSym**, given received signal **rxSig**, channel estimate data **chEst**, noise variance estimate **noiseVarEst**, and wlanNonHTConfig object **cfg**.  Finally, FCS is removed from the head field of MAC and update the index of searching. When duplicated packet is detected, the processing ends   1. **Validation under AWGN channel model** 2. **MCS = 2**   In this part, we fix MCS to 2, i.e., we choose QPSK as the OFDM modulation and coding scheme to transmit packets. And then change the value of SNR to compare the constellations and the recovered images of the results.     * **SNR = 23**       The figure shown above is the constellation and the recovered image of the 802.11a image transmission result when the SNR is 23 when we choose QPSK as the OFDM modulation and coding scheme.   * **SNR = 29**       The figure shown above is the constellation and the recovered image of the 802.11a image transmission result when the SNR is 29 when we choose QPSK as the OFDM modulation and coding scheme.   * **SNR = Inf**       The figure shown above is the constellation and the recovered image of the 802.11a image transmission result when the SNR is Inf when we choose QPSK as the OFDM modulation and coding scheme.   1. **MCS = 4**   In this part, we fix MCS to 4, i.e., we choose 16-QAM as the OFDM modulation and coding scheme to transmit packets. And then change the value of SNR to compare the constellations and the recovered images of the results.     * **SNR = 23**       The figure shown above is the constellation and the recovered image of the 802.11a image transmission result when the SNR is 23 when we choose 16-QAM as the OFDM modulation and coding scheme.   * **SNR = 29**       The figure shown above is the constellation and the recovered image of the 802.11a image transmission result when the SNR is 29 when we choose 16-QAM as the OFDM modulation and coding scheme.   * **SNR = Inf**       The figure shown above is the constellation and the recovered image of the 802.11a image transmission result when the SNR is Inf when we choose 16-QAM as the OFDM modulation and coding scheme.   1. **MCS = 6**   In this part, we fix MCS to 6, i.e., we choose 64-QAM as the OFDM modulation and coding scheme to transmit packets. And then change the value of SNR to compare the constellations and the recovered images of the results.     * **SNR = 23**       The figure shown above is the constellation and the recovered image of the 802.11a image transmission result when the SNR is 23 when we choose 64-QAM as the OFDM modulation and coding scheme. From the figure, we can find that when the SNR is 23, we cannot even recover the image.   * **SNR = 29**       The figure shown above is the constellation and the recovered image of the 802.11a image transmission result when the SNR is 29 when we choose 64-QAM as the OFDM modulation and coding scheme.   * **SNR = Inf**       The figure shown above is the constellation and the recovered image of the 802.11a image transmission result when the SNR is Inf when we choose 64-QAM as the OFDM modulation and coding scheme.  From the figures shown above, we can find that when the SNR is 23 or 29, the performance of QPSK is better than 16-QAM and 64-QAM, which also can prove that if we increase the data rate, i.e., the number of constellation point used to do the OFDM modulation, the bit error rate will also increase, or the performance of the constellation and the recovered image will decrease when the SNR of each modulation scheme is the same which can lead to bit error.   1. **Validation under HiperLan/2 channel model**   The main of this part is to verify BER under different SNR with the model of HiperLan/2 channel. According to the document of 802.11p™ and 802.11a™ Packet Error Rate Simulations, HiperLan/2 channel is created to verify BER under different SNR.    Firstlsy, we simulate the relationship between BER and SNR under channel model A, doppler frequency shift whose value is 0 and different MCS. The code is below:          The result is as below.    From the above figure, it is obvious that under the same modulation scheme, the BER decreases with the SNR increasing. This is because as the signal-to-noise ratio increases, the packet error rate decreases as the noise becomes less disruptive to the decoding. Then we compare the BER of different modulation scheme under the same SNR. From the trend of these curves, it can be predicted that BER increases with the increase of modulation order under the same SNR. This is because the higher order modulation scheme increases the transmission rate while causing the packet error rate (is the same as the BER) to be higher than the lower order modulation mode under the same SNR. Therefore, for higher order modulation processes, better transmission channels and less noise interference are needed to achieve the same packet error rate range.  Then, we simulate the relationship between BER and SNR under the value of MCS is 4, doppler frequency shift whose value is 0 and different channel mode. The code is below:          The result is as below    From the above picture, what can be known is that the trend of the relationship curve between BER and SNR under different channel models is consistent. What’s more, different channel models have a greater impact on the transmission effect.In general, the error packet rate of each channel model tends to decrease with the increase of the SNR. | |
| **Experience**  **Experience**  孙逸涵:   1. In this lab, I have learned how to use MATLAB to implement 802.11a image transmission and reception, and after this lab, I also have a better understanding of the framework of the 802.11a transmission system. 2. When I simulate the image transmission and reception when choosing 64-QAM as the OFDM modulation and coding scheme and the SNR is 23, I cannot receive the recovered image, but when we change the SNR to 29, then we can get the recovered image. So I think the reason for the failed recovery of image is that the value of the SNR is too low. 3. In addition, after I analyze and compare the training sequence of Non-HT format PPDU generated by using different methods, I am more familiar with the packet format of the Non-HT format PPDU, especially its header format.   张旭东:   1. In this lab, I have learned the generation formula, frame structure and the generation process of L-STF sequence and L-LTF sequence in Non-HT format. Besides, the meansing of L-SIG in Non-HT format is to be mastered. 2. I am more familiar with the signal processing flow chart of function **receiverProc** and have s deep understanding to effects of these functions, including **wlanFieldIndices**, **wlanCoarseCFOEstimate**, **wlanFineCFOEstimate**, **wlanLLTFDemodulate**, **wlanLLTFChannelEstimate**, **wlanLSIGRecover** and **wlanNonHTDataRecover**. 3. What’s more, I am familiar with the process of creating HiperLan/2 channel and how to verify BER under different SNR with the model of HiperLan/2.   **In-class lab screenshot**  孙逸涵:    张旭东: | |
| **Score** | 100 |