**Lab 2：16-QAM Transmitter and Receiver**

|  |  |
| --- | --- |
| **Author** | Name: 孙逸涵 (Yihan Sun) Student ID: 12012128  Name: 张旭东 (Xudong Zhang) Student ID: 12011923 |
| **Introduction**  In this lab, we focus on learning MATLAB object-oriented programming methods and keywords, and try to create new classes and corresponding attributes.  After that, we interpreted the object-oriented programming method of QPSK transmitter and receiver through official documents, and further deepened our understanding of MATLAB object-oriented programming through the interpretation of QPSK communication system program.  Finally, by comparing the difference between 16QAM and QPSK, we changed QPSK to 16QAM and added one bit to the output, so as to verify the feasibility of the 16QAM communication system in MATLAB.  **Theoretical analysis**   1. **Basic principles of 16-QAM modulation / demodulation**   16-QAM (Quadrature Amplitude Modulation) is a digital modulation scheme that is widely used for transmitting data over wireless communication systems. It is a combination of Amplitude Shift Keying (ASK) and Phase Shift Keying (PSK) techniques, where 16 different combinations of amplitude and phase are used to represent 4 bits of digital data.   1. **16-QAM modulation**   In 16-QAM modulation strategy, we map 4 bits into a symbol, for example, bit stream 10110111, we divide it into 1011 and 0111, and from the process of converting binary to decimal, we know that the highest bit in the case has a weight of 8, and the second has a weight of 4, and the third and the lowest has a weight of 2, and 1, respectively. Therefore, we can convert every 4 bits to its corresponding decimal, which is also the index corresponding to the symbol in 16-QAM symbol mapping table, which is also called constellation diagram. The constellation diagram is a square grid with 16 equally spaced points, where each point represents a unique combination of amplitude and phase. The amplitude and phase of the carrier signal are adjusted based on the selected 16-QAM symbol, and the resulting modulated signal is transmitted over the communication channel.  The 16-QAM symbol mapping table is shown as below:     1. **16-QAM demodulation**   The 16-QAM demodulation process starts with receiving the modulated signal. The received signal is first passed through a matched filter to remove the noise and distortion. Then, the signal is sampled and quantized to obtain the 16-QAM symbols. And each 16-QAM symbol is demapped to its corresponding group of 4 bits using the constellation diagram. Finally, the demodulated digital data stream is then processed for error correction and decoding to recover the original data.   1. **Functions of six subcomponents in example QPSK Transmitter and Receiver** 2. **Automatic Gain Control (AGC)**   Sets its output power to 1/sqrt(Upsampling Factor)(0.5) so that the input amplitude of the *Coarse Frequency Compensation* subcomponent is stable and roughly one. This ensures that the equivalent gains of the phase and timing error detectors keep constant over time. The AGC is placed before the *Raised Cosine Receive Filter* so that the signal amplitude can be measured with an oversampling factor of four. This process improves the accuracy of the estimate.   1. **Coarse frequency compensation**   Uses nonlinearity and a Fast Fourier Transform (FFT) to roughly estimate the frequency offset and then compensate for it. The frequency offset is estimated by using a *comm.PSKCoarseFrequencyEstimator* System object and the compensation is performed by using a *comm.PhaseFrequencyOffset* System object.   1. **Fine frequency compensation**   Performs closed-loop scalar processing and compensates for the frequency offset accurately, using a *comm.CarrierSynchronizer* System object. The object implements a phase-locked loop (PLL) to track the residual frequency offset and the phase offset in the input signal.   1. **Timing recovery**   Performs timing recovery with closed-loop scalar processing to overcome the effects of delay introduced by the channel, using a *comm.SymbolSynchronizer* System object. The object implements a PLL to correct the symbol timing error in the received signal. The Zero-Crossing timing error detector is chosen for the object in this example. The input to the object is a fixed-length frame of samples. The output of the object is a frame of symbols whose length can vary due stuffing and stripping, depending on actual channel delays.   1. **Frame Synchronization**   Performs frame synchronization with the known Barker code and, meanwhile, convert the variable-length symbol inputs into fixed-length outputs, using a *FrameFormation* System object for examples. The **step** method of the object has a secondary output that is a boolean scalar indicating if the first frame output is valid.   1. **Data decoder**   Performs phase ambiguity resolution and demodulation. Also, the data decoder compares the regenerated message with the transmitted one and calculates the BER.  **Lab results & Analysis**   1. **16-QAM Tranceiver** 2. **Programming Procedure**   The following figures show what the program changed.  SimParams    QPSKTransmitterR    QPSKBitsGeneratotR    QPSKReceiverR    QPSKDataDecoderR       1. **Simulation results**     *The Overall Result*    *The Constellation Result*    *The Spectrum Result*    *The Error Rate of the command line window*   1. **Bit error rate curve with SNR**     *BER with SNR*  From the above figure, it is obviously that BER is gradually decreasing with SNR increasing om the general trend. When SNR is in range [0 6](6 is an estimate value), BER has a significant downward trend. When SNR is in range [6 60], Its value is relatively stable within the margin of error. So it is concluded that when SNR increases to a certain range, BER will not change, which coincides with the theory. | |
| **Experience**  **Experience**  孙逸涵:   1. In this lab, I have learned how to use MATLAB to carry out object-oriented programming. Before this lab, I just know how to use MATLAB to implement simple programs which includes how to create a function, how to plot a figure of something such as an electric field, how to call the existed functions in MATLAB, and so on. But after this lab, I have a deeper understanding of MATLAB object-oriented programming. 2. And this is the first time for me to try to understand so many programs with objects, and in different objects, they have different properties and functions or methods to produce different operations. 3. I have tried my best to understand the QPSK official program, but there is still something which I am unfamiliar with in MATLAB which means that there are still many knowledge I need to learn and improve in the future study.   张旭东:   1. In this lab, I have a deeper understanding of MATLAB object-oriented programming. 2. Through the simulation of QPSK and 16QAM communication system, I have some practical experience in object-oriented programming of MATLAB 3. In this lab ,I have a clear cognition of the advantages and disadvantages of the modulation mode of PSK and QAM   **In-class lab screenshot**  As nobody has finished the in-class lab test in class, there is no in-class lab screenshot in this part. | |
| **Score** | 100 |