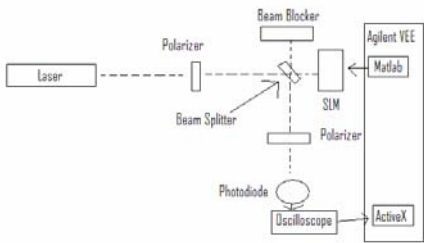
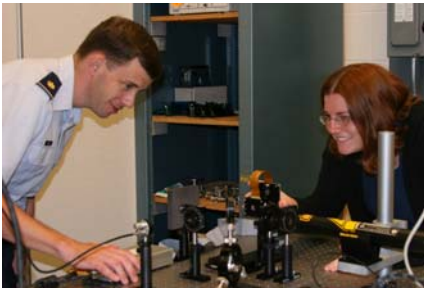




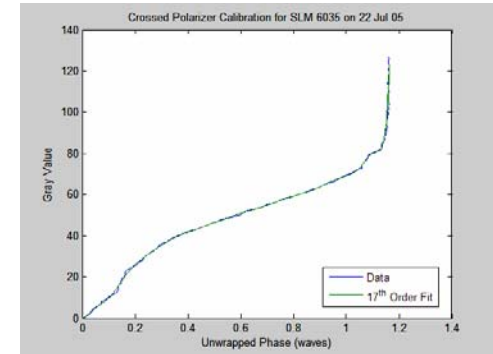
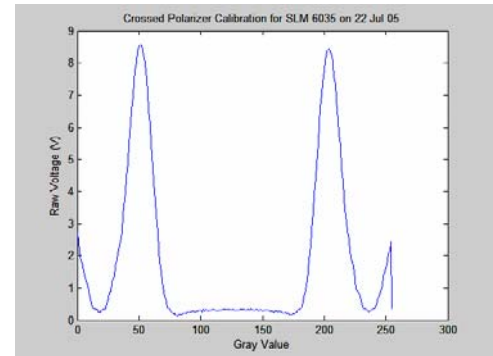
Automated Calibration and Control of Liquid Crystal Spatial Light Modulators:

Julie Chapman (Case Western Reserve University), Matthew Goda (Air Force Institute of Technology), and Jason Schmidt (University of Dayton)
 Air Force Institute of Technology, Department of Electrical Engineering

Overview: We used a liquid crystal spatial light modulator (LC SLM) to simulate the aberrations on a beam of light caused by atmospheric turbulence. The SLM applies a voltage to the liquid crystals in the SLM which changes the orientation of the liquid crystals; therefore, the index of refraction is altered. When the light passes through the liquid crystals, the phase of the light is shifted. The computer can send 256 different gray values to each pixel of the SLM; each gray value is representative of a different voltage. There are 512x512 pixels. We had to calibrate the SLM to find the relationship between the 256 gray values and the change of phase. Our goal was to calculate a polynomial fit, so that we could convert the phase we want on each pixel of the SLM into a gray value to apply to the SLM.



I wrote an Agilent Vee program that uses ActiveX to communicate with the oscilloscope and Matlab to send frames to the SLM. This set-up (middle left) turns a change of phase into a change in intensity in a known way. Using the program, we would send a frame of constant gray value, take 100 data points from the oscilloscope, average the data, and then send another frame of constant gray value. We repeated this for each gray value. The plot of intensity vs. gray value applied is shown to the upper right. After using the wave in the first half of the data (0 to 127 gray values), we were able to make a phase to gray polynomial fit (middle right). We repeated this process for each SLM. Using a similar program, we tested our calibration by giving a phase, converting to gray, and then applying to the SLM. This way we were able to compare the phase commanded and the phase measured.



I designed and programmed this Matlab graphical user interface (GUI) to control our SLM (bottom left). The GUI takes user inputs to create 512x512 bitmaps in three ways. The first method is to take a pixel (or list of pixels) and apply the phase specified by the user. The second technique is to use Zernike polynomials given by the user to create common aberrations. The third is to load a Matlab file which has a 512x512xN matrix in it; this matrix can be played like a movie. The user also specifies which SLM is being used and if they want the backplane warping of the SLM to be compensated. When the user applies the frame, the frame is converted to gray using the polynomial coefficients that we calculated when calibrating the SLMs. The backplane is added if the user indicated that the backplane should be compensated. The frame is then loaded and sent to the SLM. The bitmap can be saved using the save function of the GUI.

