Characterizing holographic displays via numerical simulations

Jani Mäkinen

August 31, 2023

1 INTRODUCTION

The goal of this assignment is to familiarize students with the basic concepts of holographic displays and how such displays can be characterized using numerical simulations. No previous knowledge of holograms or wave optics is required, however basic MATLAB programming skills are needed to successfully complete this assignment. Majority of the implementation is already provided in the zip-archive (available in Moodle), though the students are asked to fill in short sections of the code. Furthermore, larger focus is on using the code, plotting and analyzing results.

The particular use case in this task is the characterization of accommodation cues in near-eye holographic displays. For this purpose, the viewing process needs to be simulated. This includes numerical field diffraction and a simplified computational model of the human eye. The simulation model allows us to obtain images on the retina, which can be used to analyze the perceived sharpness of the displayed content.

2 IMPLEMENTATION STEPS

The major steps of the MATLAB implementation are described here. The end of each section summarizes the tasks and questions of the step. The report should include the requested plots and answers to all of the questions in order to receive a passing grade.

2.1 Hologram synthesis from a single point source

The computational synthesis of holograms is not a trivial task. A significant portion of computational holography relies upon accurate modelling of wave optics and coherent imaging techniques. For the purpose of this assignment, it is sufficient to know that the sampled hologram (i.e. the holographic display) is described as a set of complex values on a plane. The complex amplitude on the hologram plane contain both the intensity of the light (as the amplitude) and the depth information (as phase). In this assignment, the analysis can be simplified significantly by considering only a cross section of the entire 3D space. Thus, the hologram is given by a 1D array of complex values. This simplification also reduces the computational burden in the propagation step later in the process. This is crucial as the number of samples in a hologram (or holographic display) can be extremely high, which can cause issues e.g. with the amount of memory required to store the values.

Let us consider how to obtain the complex values on the hologram plane. The complex wave field is generated based on the content to-be-recorded on the hologram. As our simulation setup models the content as point source(s) of light, the field is obtained as a summation of contributions from each point source p. Utilizing the Fresnel diffraction kernel, the 1D field $U(x;0)$ is defined for a single point source at position (x_p, z_p) as

$$U(x;0) = a_p \frac{\exp(jkz_p)}{\sqrt{j\lambda z_p}} \exp\left(jk\frac{(x-x_p)^2}{2z_p} + \phi_p\right), \tag{2.1}$$

where a_p is the amplitude of the point source, ϕ_p is the relative phase, λ is the wavelength of the monochromatic light (i.e. the color, if in visible spectrum) and $k = 2\pi/\lambda$ is the wave number. For the purpose of this assignment, the amplitude and relative phase can be left to their default values in the code ($a_p = 1$, $\phi_p = 0$).

The hologram can be synthesized using various different methods. In addition to the Fresnel kernel, one option is to use the Rayleigh-Sommerfeld diffraction kernel, for which the field is defined as

$$U(x;0) = a_p \frac{z_p}{\sqrt{j\lambda}\left[(x-x_p)^2 + z_p^2\right]^{3/4}} \exp\left[jk\sqrt{(x-x_p)^2 + z_p^2} + \phi_p\right]. \tag{2.2}$$

Using the definitions of the Fresnel and Rayleigh-Sommerfeld holograms, fill in the missing lines of code for the hologram synthesis in the script `mainSimulations.m`. **Note:** for implementation purposes, use the difference between the hologram position z_c and the point source z_p in the z direction, i.e. $z_c - z_p$, in the equations in place of z_p. Additionally, familiarize yourself with the overall structure of script. Find the implementation of the 1D holographic stereogram and discuss in the report whether there any differences between it and the other two methods.

TASKS AND QUESTIONS OF THIS STEP:

1. Implement the 1D Fresnel hologram synthesis using Eq. 2.1.
2. Implement the 1D Rayleigh-Sommerfeld hologram synthesis using Eq. 2.2.
3. Find the implementation of the 1D holographic stereogram inside the provided code. Are there any differences between it and the other two methods?

2.2 Retinal image formation model

The retinal image formation model is a key part of the analysis. It provides a method for mimicking the visual process of a human eye through a numerical simulation procedure. In general, these simulation tools consider the human eye as a camera with a thin lens, where the lens is equivalent to the pupil and the sensor to the retina. This configuration defines three parallel planes: display (or hologram), lens and sensor. Each plane needs to be discretely sampled according to their corresponding sampling requirements for numerical simulations, which is typically done in a uniform fashion. The pupil is modelled as a thin lens of width D. The 1D transmittance function $T(s)$ of a thin lens is defined as

$$T(s) = \exp\left(\frac{-j\pi}{\lambda f} s^2\right), \tag{2.3}$$

where f is the focal length of the lens.

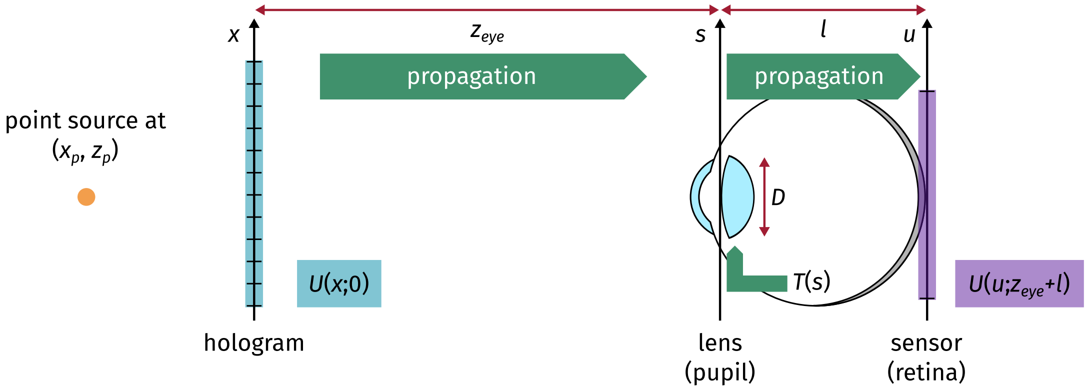

Figure 2.1: The general schematic of the simulation model. The hologram $U(x;0)$ synthesized from the point at (x_p, z_p) is first numerically propagated towards the thin lens, then multiplied with the lens transmittance function $T(s)$ and finally propagated to the sensor plane. The retinal image is obtained from the field at the sensor plane.

In addition to the computational eye model, the propagation of light from the display to the eye needs to be modeled. The scalar optical field $U(\xi;z_0)$ is propagated utilizing plane wave decomposition, i.e. for each spatial frequency component f_ξ in the field, a plane wave is propagated with its corresponding complex amplitude and sampled at the desired secondary positions (x, z). The contributions of each plane wave are combined to obtain the total field $U(x;z)$ as

$$U(x;z) = \int \mathscr{F}\{U(\xi;z_0)\}(f_\xi)\exp\left[-2\pi j(f_\xi x + f_z z)\right] \mathrm{d}f_\xi, \tag{2.4}$$

where

$$f_z = \sqrt{\frac{1}{\lambda^2} - f_\xi^2}. \tag{2.5}$$

The fields are discretely sampled in the numerical simulations, thus the integral in Eq. 2.4 can be replaced with a summation and the discrete Fourier transform operation to obtain the complex amplitudes for each plane wave. The MATLAB implementation of this is provided in the function `plane_wave_decomp.m` and the general schematic of the process is visualized in Fig. 2.1.

The function `propagateField_PWD.m` contains the entire process of propagating the field from the hologram towards the eye, and finally to the retina to obtain the retinal image. Familiarize yourself with the function and summarize the key parts of it in the report. Take special note of the focal length and its role in the model. How is it related to the function input parameters? Moreover, it should be noted that the computational model of the eye utilized here is significantly simplified from a real human eye. Discuss these differences and how they might affect the results.

TASKS AND QUESTIONS OF THIS STEP:

1. Familiarize yourself with the function `propagateField_PWD.m`. Summarize the key parts of it in the report.
2. What is the role of focal length f in the eye model?
3. Discuss the differences between the simplified computational eye model and the human eye.

2.3 Analysis of the retinal images

The retinal image can be interpreted as the perceived point spread function (PSF), i.e. how a single point encoded on a holographic display is perceived when viewing it. The MATLAB function `propagateField_PWD.m` gives as an output this image, as well as the sample grid of the retina (or sensor). Plot the PSF samples as a function of sensor sample positions and describe its behaviour. Given the PSF, one can obtain also the modulation transfer function (MTF) as the magnitude of its spectrum. The MTF describes the contrast magnitude across different spatial frequencies. The relative magnitude between the MTFs at varying focal distance of the simulated eye is crucial, as contrast magnitude is one of the factors driving the accommodation response of the eye. This forms the basis for our analysis in this assignment.

The accommodation cue provided by the display can be characterized by altering the focal distance of the eye model and evaluating the sharpness of the perceived PSF. For this purpose, modify the main function such that the process is repeated for different values of the focus distance of the eye z_f. Choose values on both sides of the point (i.e. both larger and smaller than z_p) and relatively close to it. Store the PSF and MTF for each value of z_f. Use the example code to evaluate the value of the MTF at a certain spatial frequency (e.g. 15 cycles per degree), again for each z_f. Plot these values as a function of z_f. In addition, plot all of the PSFs in one figure (e.g. by using different coloured line plots). Fit a continuous, well-defined function to the MTF data points and evaluate at which value of z_f the function is maximized. **Note:** the chosen function does not have to pass through all data points, rather it should be chosen such that it best describes the expected behaviour of the data. In the report, remember to also discuss which type of function you have chosen to fit to the data and the motivation for it (i.e. how does it relate to nature of the problem). Importantly, do not just interpolate values in-between the data points. The MATLAB functions *fit* or *cftool* might be helpful for this task. The value of z_f which maximizes your chosen function will be the estimate of the most likely depth the eye will accommodate to. Include in the report a plot with the data points, the fitted function and the maximum of the fitted function. In an ideal case, the estimate should be equal to the exact location of the point source. However, approximations either in the encoding step (i.e. hologram synthesis) or in the simulated wave field diffraction can cause it to change.

After completing the previous tasks, experiment with changing some of the parameters (e.g. the position of the point source, size of the pupil) and see how the results are affected. You may also try out different hologram synthesis methods and compare the results between those. Include the results from these experiments in the report.

TASKS AND QUESTIONS OF THIS STEP:

1. Run the main script to obtain a single PSF. Plot the PSF samples as a function of sensor sample positions. Describe the behaviour of the function. How would you expect it to change when the focal distance of the eye is changed (in relation to z_p)?
2. Repeat the process for varying values of z_f. How does changing its value affect the PSF and the MTF? Include a plot of all PSFs obtained from the set of different z_f values.
3. Calculate the MTF values around certain spatial frequencies for each value of z_f. Plot these values as a function of z_f. Fit a function (using e.g. *fit* or *cftool* in MATLAB) to the data points and evaluate at which depth the function is maximized. Include both the function and its maximum value in the figure. Discuss also which type of fit you used and why.
4. Experiment with changing some of the parameters and/or the hologram synthesis method. How do the results change?

3 Completing the assignment

3.1 General guidelines

As the majority of the implementation is already provided, very little changes are needed in the code. The missing parts are highlighted in the script and should be easily found. The code should be returned in such a state that it can be run without any errors. If you make any changes to the existing code, highlight the changes with descriptive comments.

The final report must be in PDF format; reports returned in any other file format (e.g. Word doc) are immediately rejected and will not be evaluated. Please make sure that all your figures are easily viewable, i.e. large enough font size, appropriately labeled axis and high enough visual clarity. Preferably prioritize the use of vector graphics (e.g. PDF or EPS file formats). The report should be structured cohesively in a standard scientific format, including an introduction, explanation of the overall method and the results. Please make sure you have answered every question in your report and completed every task before submitting your work.

3.2 Submitting the report

Collect all MATLAB scripts and your report into a zip-archive and return it via Moodle. The deadline for the submission is 15th December 2023. Late submissions can be considered only if the instructor is contacted before the deadline.