Lensless Digital Holographic Microscope

Senior Design II
Final Project Document and Group Identification
University of Central Florida
Department of Electrical Engineering and Computer Science
College of Optics and Photonics

Group 7

Nicolas Bonaduce	CREOL (OSE)
Julian Correa	Computer Engineering
Parker Crooks	CREOL (OSE)
Nick Scarlata	Electrical Engineering

Table of Contents

1.0 Executive Summary

As the world gets increasingly more and more high-tech, the divide between those with access to the new technology and those without gets larger. The aim of this project is to bring about access to microscope technology to a wider group of people. This includes people who would prefer to not spend so much money on microscopes, and people who find microscopes too clunky to do field work. This is where the lensless microscope comes in. Using optical technology and computer software, we created a product that will be portable, low cost, and useful. As CPE, EE, and COP students, we are the perfect fit to tackle this job; seeing as we have been trained in a breadth of optical technology throughout our career in the CREOL labs, learned how to program and understand exactly what we need in COP, as well as the electrical know-how to tie it all together. Working as a group, we presented a small field-portable microscope that will increase interest and access into the field of microscopy, and who knows what secret of the universe of new technology that will bring. There has already been one documented case of a lensless digital holographic microscope and we have been studying Yichen Wu and Aydogan Ozcan's paper on the topic. The resources we needed was mainly financial as this project will take us some time to complete and we undoubtedly ran multiple iterations to find the perfect fit. The profitability is very high because these microscopes can be mass manufactured and sold for low cost. This project was completed within the year.

2.0 Project Description

A lensless microscope that will be viewed digitally either through a computer or a smartphone. The imaging was constructed using back-propagation and shift-and-add pixel super-resolution of in-line holograms captured on a CMOS Sensor, see figure 1 for example. This was done using fiber optics and an LED array to slightly shift the attack angle of the incoming light onto a semi-transparent sample.

Possible uses include Oil monitoring in Ocean Water (Tracking runoffs, seeking oil sources) Air health (Bioaerosols, Pollution, Mold Spores, Bacteria) Plant health (Cell burst, Cell Hyponatremia)

We worked with the CPE and EE to create a microcontroller that will understand the optical formulas enough to create a visible hologram. Along with this, we had the hopes of creating an iPhone app that will display the hologram on the screen for easy access. In terms of the electrical components of this product, they were rather low cost and low in weight, they shouldn't become too unwieldy to feel awkward to work with either in the field or in a lab setting, current concept in figure 2.

For the optical design of this project, rather than splitting into two separate optical designs, we worked together with the CPE and EE students to create a good blend between the optical side and the computer and coding side. We needed to both design the physical device and work the formulas out enough to be translated into usable code for our teammates. Our optical design was something similar to the device shown to the right. We might change the device to be able to fit as an attachment on an iPhone, while still functioning well.

This section contains:

- The background of the project.
- The goals and objectives our project hopes to tackle.
- A motivation discussion section detailing the influences the members had on
- choosing to work on this project.
- A specification list showing the necessary requirements and core, advanced, and
- stretch features of our project.
- Highlighting each team member's area of contribution
- House of the quality diagram
- Functional diagrams detailing how the optical components of this design will
- operate.

2.1 Project Background

To complete this project we needed to be on time and have the hologram imaging done by December - mid-January. This allowed us to spend the remaining time on refining the computer software we needed to properly back-propagate and image the subject. Prerequisites for this include a substantial amount of research into the different components of the project as well as group meetings where we come together to see how far everyone got. To produce the expected amount of work we met every other week to ensure every member was doing their part. We also had meetings with advisors both associated with the senior design class and advisors separate from the class. These will be people with expertise and experience in the field of microscopy, computer software, imaging hardware, image manipulation, and fiber optics.

2.2 Motivation

To create a simple, cost-effective, and portable digital lensless microscope that can be constructed at home or mass manufactured with smartphone compatibility either through Bluetooth or direct lens imaging. Our team has worked together using our knowledge from our time at the University of Central Florida to form this project. By using effective communication, we've been able to demonstrate this knowledge and accomplish our common goal. The hurdles and challenges presented in a project like this will be a fantastic experience for the "real world" projects we will encounter down the road. By creating a smaller and cheaper microscope that is lensless, it would be better to use with students, as breaking the device would be a much smaller financial burden. The biggest hurdle with this current technology is the price point which is too high to get into as a hobby. Other limitations include the restrictions on what can be imaged. While we would still be restricted to only imaging semi-transparent subjects, a stretch goal of ours is to provide documentation on how to manipulate the specifications we provide to be able to image a much wider variety of subjects. This is our biggest weakness. Depending on what we would like to image, we need to be locked into a wavelength and distance. Our description of the microscope would provide what will most likely be an excel chart and program detailing what must be changed to image whatever the end user would like.

2.2.1 Goals

The primary goal of the Lensless Digital Holographic Microscope was to deliver a high-quality product that was able to deliver high-resolution and high-throughput imaging using compact, portable, and cost-effective optics to create a visible hologram for any field our product will be used in. Optics and Photonics will be used to digitally reconstruct microscopic images without using any lenses. As a result, was able to be made much smaller, lighter, financially cheaper, and incorporates software that would bring together a product that is affordable for any user in many applications.

2.2.2 Objectives

The main objectives of this project will be:

- Coupling as much LED light radiation into multimode fibers as possible
- Sequentially flashing the LED array with variable timing for imaging at the CMOS sensor
- Program PSR/Gabor Algorithms
- Ensure power supply falls in line with device goals

2.2.3 Design Diagrams

For a quick explanation of figure 2.2.3-1 this is meant to show our group's workflow, how even though we each have our independent responsibility, what we end up working with or finishing a task for is still affecting another one of our group members or the final result as a whole. Meaning we each have a desire to desire our parts with the rest of us in mind since a small change in one of our fields that makes it slightly harder for the first person to finish an aspect of there work can mean that the person that receive our work does not have to do what could be a confusing work around later down the line in the project's lifespan.

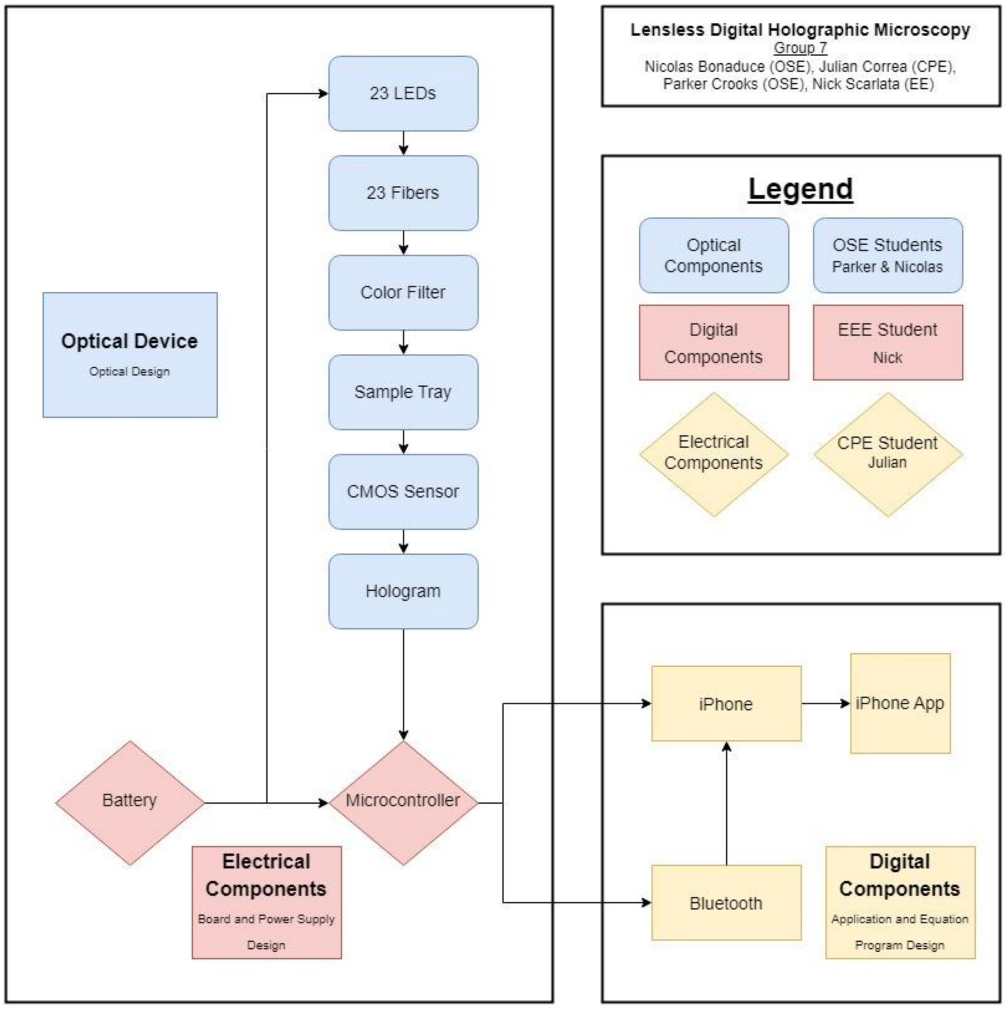
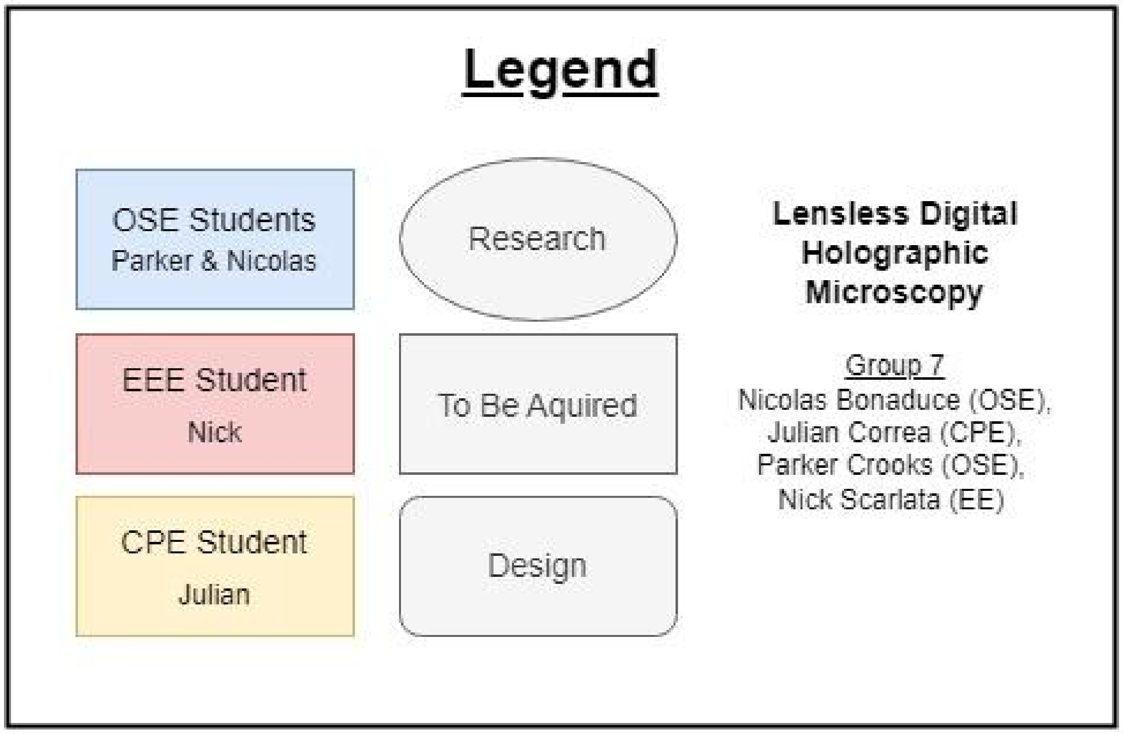


Figure 2.2.3-1. Design Specification Diagram



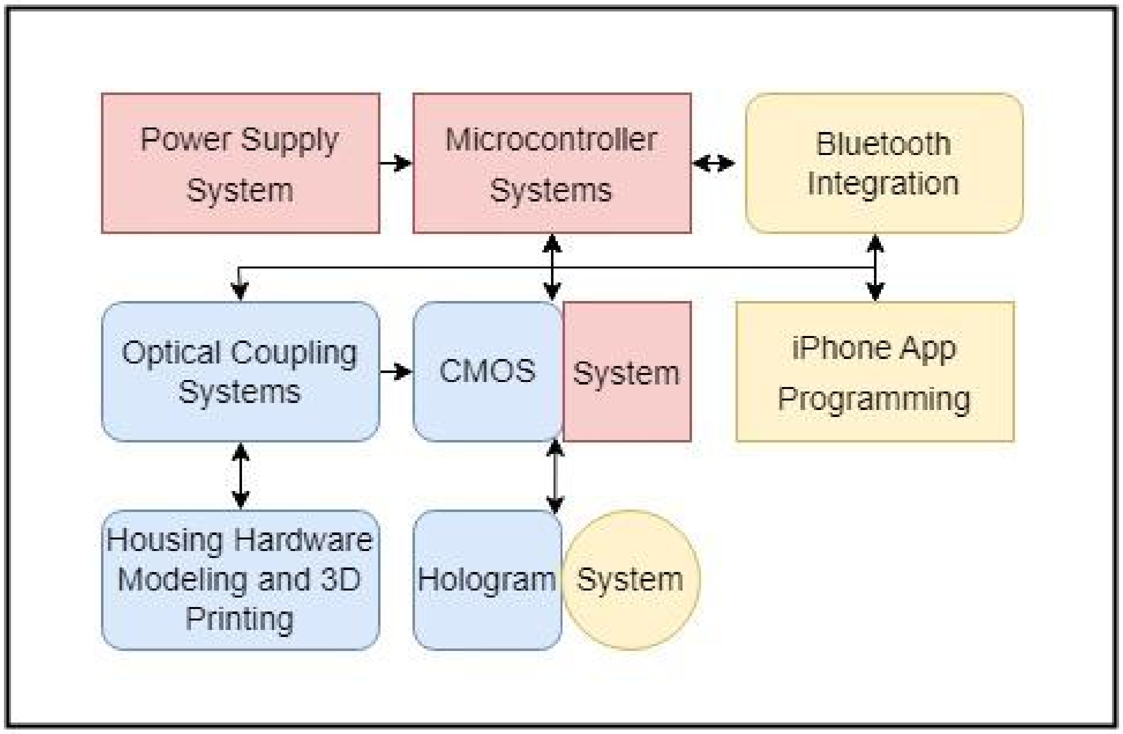


Figure 2.2.3-2. Block Diagram

2.3 Requirement Specifications

The requirements specifications of The Lensless Digital Holographic Microscope project can be broken down into three main categories: Electronic, Photonic, and Programming. The first category covers the main electrical handling of components and how components work with one another. The second category covers the photonics within the project, which deals with the implementation of photonic principles and theories to connect the mathematics to the components and their placement. The last category covers the programming of the project, which covers the programming of the photonic theories and being able to have something to be able to display taking what we are testing and having something to demonstrate when it comes time to present. The Final section highlights the engineering requirements that our project will contain.

2.3.1 Electronics Requirements

Below there is a table to show the requirements of the Lensless Digital Holographic Microscope, a simple technical answer will be found in the table and a written explanation can be found below explaining our initial reasoning.

Electronics Requirements	Justification
1. A source of power either on board or using the device it is mounted to for power	>15V
2. A sensor (the related material showed a specific CMOS sensor, there will be a pro-con list between this sensor and all relevant substitutes)	>6V Pin to pin or USB connection
3. A microcontroller (Anything as robust as a raspberry pi or simple embedded system that can send a Bluetooth signal can be utilized)	>8V Two microcontrollers needed after research
4. LEDs (25)	>1.5V

For the first requirement, there is going to be a relatively low load requirement, due to the small list of electrical components, but since there could be an available phone we need to see if it is efficient and not too unwieldy to use the phone’s battery. However an onboard rechargeable battery might be implemented anyway. For the second requirement, this device will need an imaging sensor that supports color in some capacity, there are a lot of possible sensors on the market on of which will be needed to complete this device. For the third requirement, Ideally the microcontroller should be performing as much as the computations as possible, so there will be a cost-benefit analysis in future papers about what each relevant microcontroller can contribute to this device. But the bare minimum is just to send the Bluetooth signal and have the phone do the computations. Additionally we have decided that two microcontrollers will be necessary, this explains the increase in voltage then initially thought. Lastly, the fourth requirement, from an electrical point of view we just need to have them be powered, and connected separately for how the image process is intended to work.

2.3.2 Photonics Requirements

In this section there is a table to show the Photonic requirements of the Lensless Digital Holographic Microscope, a simple technical answer will be found in the table and a written explanation can be found below explaining our initial reasoning.

Photonic Requirements	**Justification**
1. LEDs (>20)	Light source, what encodes the data into the CMOS sensor
2. Fiber	What guides the light and allows us to position the light in specific positions
3. CMOS Sensor	What will end up capturing the in-line hologram data and communicating it to the computer
Resolution Size 50 µm - 250 µm	Will be tested with desired field's subject matter, as well as micrometer beads in the design phase
For the spatial frequency component of the object to be imaged at the sensor plane, the optical path length difference (ΔOPL) of the high frequency scattered wave must not exceed the temporal coherence length (ΔLc)	This is what will create the Fresnel diffraction required to properly send the holographic information to the imaging screen.

2.3.3 Programming Requirements

In the table below we discuss the requirements of the Programming and Software side of the Lensless Digital Holographic Microscope's project cycle. This table offers a simple technical answer that will be found in the table and a written explanation can be found below explaining our initial reasoning as to why that requirement is necessary.

Programming Requirements	**Justification**
1. Should be able to implement phase retrieval with the use of deep learning	Phase information was crucial to be able to back-propagate the holographic interference pattern and reconstruct the subject.
2. Should have the ability to generate an in-line hologram/ Gabor Hologram	the main ability to record and present information for the project. The ability to get the hologram allows for the implementation method to record the entire field information (i.e. amplitude and phase), not just the usual intensity
3. Should be able to create a focused image with the use of simple backpropagation in order to create a focused and defocused image (two-image artifact)	this allows for no loss of resolution
4. Should implement the "shift-and-add" in order to take low-resolution holograms and up-sample, shift, and digitally add them in order to create a good image	in order to create an image with the highest resolution possible
5. Should be able to implement digital hologram reconstruction (digital back-propagation) with the method of angular spectrum method	The reason is that the image quality would be compromised by the use of twin image and self-interference related noise terms, thus it needs to be improved by phase retrieval techniques. This is because the recorded hologram we get has only the amplitude information and its phase is originally missing

For the first requirement, the programming side of the project was be able to convert the concepts of holography and other photonics theories mentioned later in the research section in order to be able to understand the information that the censor was gathering and being able to use that phase retrieval with the use of deep learning we are able to gather phase information to then back-propagate the holographic interference pattern and reconstruct the subject to show only the understanding we have of the photonic process but to has to show that the microscope works as intended. For the second requirement, having the ability to generate an in-line hologram/ Gabor Hologram allows us to record and present information for the project and to show customers or advisors that the project works as we intend for and its applications are just as we initially discussed. The ability to get the hologram allows for the implementation method to record the entire field information (i.e. amplitude and phase), not just the usual intensity. For the third requirement being able to use the two-image artifact which allows for us to create a focused image with the use of simple backpropagation is important for the project as a whole because it allows use to show that in order to create a focused and defocused image we must undergo this process and the benefits would be that we would not experience loss of resolution presenting the best resolution images for our paper and presentation to show how effective our microscope is. For the fourth requirement it is important to implement the shift-and-add method because it allows for us to use the low resolution images we can first gather to then digitally add them in order to create a good image which again allows us to present a higher resolution image if we did not implement these methods. Lastly, the last requirement of using back propagation to reconstruct the digital hologram is crucial as many times the image quality we capture of what we are testing be it oil filled water, blood, plants, water quality, etc, would be comprised of twin images and self-interference related noise meaning that using phase retrieval techniques allows us to improve the images as the recorded holograms we get has only the amplitude information and its phase is originally missing.

2.3.4 Engineering Requirements

The below table summarizes the above tables and highlights important requirements, this is the table of requirements that will be listed in the house of quality:

Engineering Requirements	**Values**
Cost	>$800
Power	>15V
Microcontroller	>20s latency while being able to control the system
Optics	Can visualize a subject between 50-250 µm
App	>60s latency
Sensor	Pixel size of >3 x 3 µm
LEDs	An array of 25 at 220 lumens

2.4 Features

This section discusses the core, advanced, and stretch features that the Lensless Digital Holographic Microscope will implement and hope to implement. These features are implemented depending on the many different constraints discovered during the project life cycle. These features help show distinctive attributes and aspects that help show what Lensless Digital Holographic Microscope is.

Our core features include imaging at a 250 micrometer level, while we have advanced and stretch goals to bring that down to 150 micrometers and 50 micrometers, respectively. Another core feature we have is wiring and programming a microcontroller to be able to communicate with the optical device and relay the image data to a smart-phone IOS application via bluetooth integration. A stretch feature we have considered is scalability of this device as a product and differentiation based on specifications of what is being imaged.

2.4.1 Core Features

The core features are the central, innermost, and most essential part of our product. They highlight the major must-haves in our design and product.

The core features we plan to implement are:

- The Device is able to turn on and power the LEDs in sequential order, via an form of power source (onboard battery or plugged into a computer)
- The Device is able to take images from the CMOS sensor and store them in the Microcontroller (they will be basically incompressible but useful)
- The Device can make use of the Raspberry PI feature to email the user the results.
- The device is portable and can be used for field testing.

2.4.2 Advance Features

The advanced features outlined below will help elevate our project and show a higher level of design implementation. These goals and their completion help take our product to the next level.

The advanced features we plan to implement are:

- The Device is powered by an onboard battery, and all the electrical components are stored on the device, as if it was a modern product
- The Device is able to process the images into a single usable image, and be able to send it to either the user's phone or computer of choice
- The Device will be able to provide useful imagining of the desired goal at the practical magnification
- The Device can make use of the Raspberry PI feature to instead airdrop the end results to all relevant people who could benefit from the information in the area

2.4.3 Stretch Features

The following stretch features may be implemented if time, budget, and capability enable. These are set in place in order to inspire growth and counter complacency within our team and product development.

The stretch features wish to implement would be:

- In order to view different specific things a change is necessary both to the optics and the programming we've done for the imagining, therefore as a stretch goal we would want to work on other versions of this device that can effectively do the same thing but for other relevant subjects.
- To make our device more accessible we want to make the blueprint, design specifications, and code open source, so anyone with reasonable knowledge and equipment can replicate our work, with our bill of materials
- Lastly find a suitable bluetooth adapter for the Raspberry PI to send the end results to the user via bluetooth.

2.5 Programming, Electronic, and Photonics Requirements

The General Design block diagram and the Requirement Specifications can be found in figure 2.2.3-1. The responsibility of each aspect is determined by the shape of the block, and the color will determine the current state of that feature. The responsibilities go towards our respective majors, so the optical students will share that load. We also worked jointly on select aspects of the design and computation.

2.6 Marketing Requirements

Marketing requirements that are needing to be considered was the size of the market that we are trying to get involved with. Hand in hand with the previous requirement was the understanding of the market we're targeting, and why we believe it's worth pursuing. Next was considering the type of clientele that we wanted to target, who are the buyer and user personas in this market. Within this market we also needed to take into account what the problems are that we can solve for these potential customers and how we solved those problems while keeping in mind potential competitors. Identifying certain goals was important to ensure the highest quality product possible for the end user. The next section of the House of quality was important to have during the planning stages of a new project as its creation and development allowed for the team to keep track of the requirements set forth by the customer alongside those created by the design team while staying aligned with the marketing requirements of the project. The benefits and pay offs of a strong house of quality was invaluable. This key element of the project helped reinforce the goals of the project and pursue what the customer has asked for in the product.

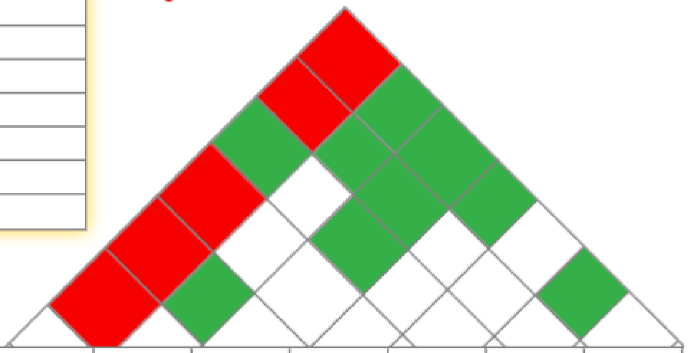
2.7 House of Quality

In the figure (Table 2.6-1) below, we illustrate the various tradeoffs and benefits that we tried to provide our customers, as well as show the correlation between our design concepts. The exact strength of the correlations may change over time from the time we began to develop the core of the product, rather than the concept that we were trying to illustrate at the moment. In addition, most of the engineering requirements featured here are closer to placeholder ideas, rather than the standards we wanted to meet at the end of the project. This was only the second iteration of our house of quality. It will likely change again by the end of this experience.

Very Strong +	+3
Strong +	+2
+	+1
Neutral	0
-	-1
Strong -	-2
Very Strong -	-3

+ positive, supporting
- negative, tradeoff

Interrelationship matrix



Product Requirements / Customer Requirements	COST (-)	POWER (-)	MICROCONTROLLER (+)	OPTICS (+)	APP (+)	SENSOR (+)	LEDs (+)
Cost (-)	+2	+2	+2	+1	0	+3	-1
Accessible (+)	0	+1	+2	+1	+2	+1	+3
User interface (+)	0	0	-2	-2	+2	0	0
Precision (+)	-2	-1	+1	+2	+1	+2	-1
Visibility (+)	-2	0	+1	+2	+2	+2	+2
Latency (-)	+2	-3	+3	0	+1	-2	-2
Features (+)	-1	0	+2	+1	+2	-1	0
Engineering Requirments	$800	>10V	>120s	50-250 µm	>60s	>3x3 µm	220 lm

Table 2.6-1. House of Quality Diagram

2.8 Functionality of the Lensless Digital Holographic Microscope

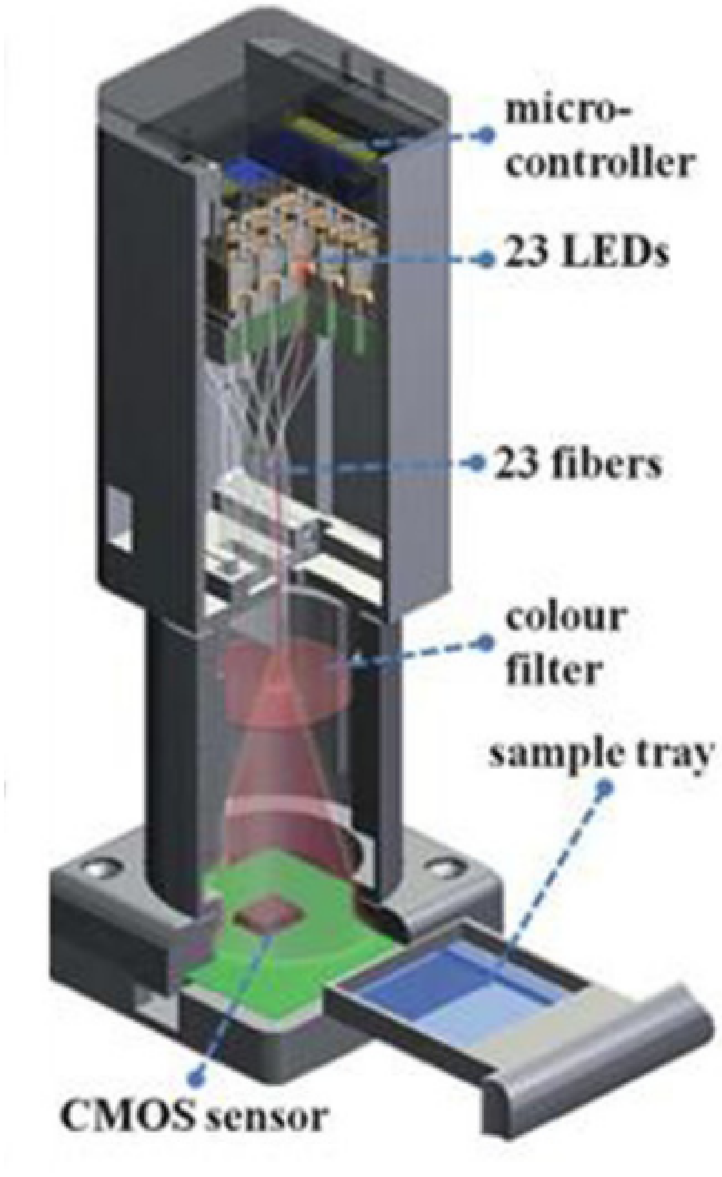


Figure 2.8-1. UCLA Optical Depiction Layout of Ozcan and Wu's Lensless Digital Holographic Microscope

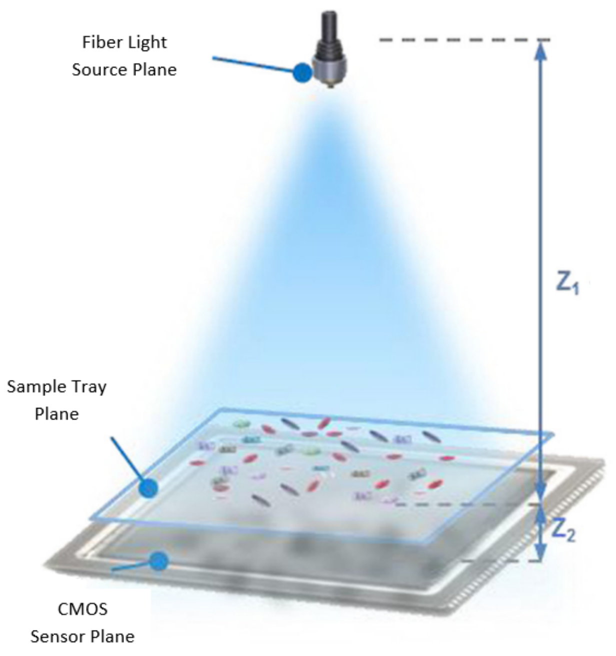


Figure 2.8-2. Spacing representation of imaging plane subject plane and illumination point

3.0 Research related to Project Definition

This section focuses on related research towards related technologies used within the Lensless Digital Holographic Microscope. This provided an opportunity for the group to see an overview of the components and how they will function with the device along with the other photonic elements seen later in the paper.

The following section contains:

- A list of existing projects and products pertaining to the development of the Lensless Digital Holographic Microscope
- An overview of the accessories and features related to the design.
- An overview of the general components used in the design

3.1 Existing Projects and Products

Lensless Digital Holographic Microscope products are not a new idea. As mentioned before one research group has already demonstrated the viability of this product. It was up to our group to improve upon this technology and add something novel and useful to it. This was why we believe smartphone integration was the right way to go. The group did not have integration and only imaged from their computer. The field portability of the device is significantly less if it must be connected to a computer to image/record data.

3.1.1 CMOS imaging software

There are companies that already had software that is available for purchase and use that deals with the imaging of CMOS sensors. CMOS imaging software was usually proprietary and we would follow whatever the manufacturer's recommendation was. But if we ran into financial constraints, there are cheaper and even free CMOS imaging softwares online. Two companies that we have found are AMS OSRAM and Hamamatsu Photonics. Hamamatsu offers software that provides the interface to access all their carefully engineered camera features, from simply setting exposure to orchestrating complex triggering for multidimensional experiments. Not only were the cameras and sensors supported by most imaging platforms, they also offered software development tools for Windows, Linux, MATLAB and LabVIEW. For AMS they were a company that is a global leader in optical solutions that offer unique products and technology for sensing, illumination and visualization. From prime-quality light emitters and optical components to micro-modules, light sensors, ICs and related software they have software that can help for all areas in photonics. For the CMOS software they have a variety of products and one that is highlighted for us is their New Image Sensor Evaluation Kits. The new NanEyeC demo kits include all necessary drivers to interface to the camera module as well as to the reference schematic for fast hardware implementation with the Arm Cortex M and Cortex A processor platforms.

3.1.2 RP Photonics

RP Photonics is a software product that uses powerful script language for Computer Modeling for Laser Development and Laser Science. This product uses powerful simulation software in order for the users to execute complex scripts for the photonics calculations that are seen within the Research and Part selection later in the paper. This software essentially provides the customer with a plug and play type product, making the idea of creating things within MATLAB nonessential. The product is able to be used without MATLAB or similar, even in applications not related at all to the physics and optical simulations. RP Photonics also offers the development of specialized software for customers with some examples being the following:

- that they highlight being the various kinds of simulations, e.g. of laser and amplifier dynamics
- calculations in optics, e.g. concerning material dispersion, interferometers, Kramers-Kronig relations, etc
- specialized data acquisition and processing, e.g. calculation of noise spectra from data recorded in the time domain

As of now this would be one of the relevant products that are seen within the world that would be considered for the photonics equations when we began the imaging and software development cycle of the project.

3.2 Similar Technologies

(3)
This section is for technologies that are in the same realm that provide inspiration or a call to action for our design.

3.2.1 Compound Confocal Microscope

Confocal microscopes use light from a laser and an objective lens to excite a specimen within a narrow focus plane, and any out of focus plane emissions are rejected by the confocal aperture. They can optimally reach resolutions of 180nm laterally and 500m axially. Confocal microscopes are the most commonly used microscopes for experiments done in vitro. This Technology is pretty expensive and is usually quite large in size, comparable to what we are aiming to accomplish with our design. Compound microscopes are expensive, heavy, and not very usable in the field, where a lot of studies are done.

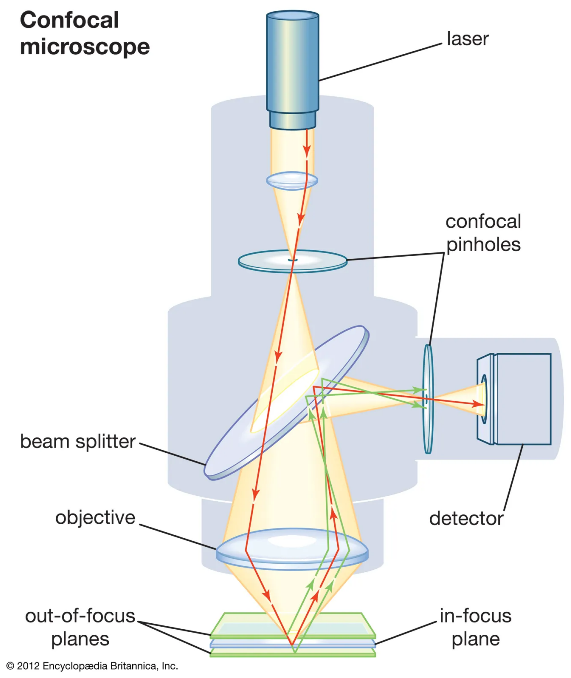


Figure 3.2.1-1. Confocal Microscope Depiction

3.2.2 Electron Microscope

Electron microscopes use a beam of electrons, rather than photons, to illuminate a sample, This is usually done by focusing the beam using electromagnetic coils instead of lenses. This is similar to our scope in that it is a lensless microscope, but completely different in design and theory. Electron microscopes have a theoretical resolution of .23nm for an accelerating voltage of 100 keV and .12nm for 300 keV.

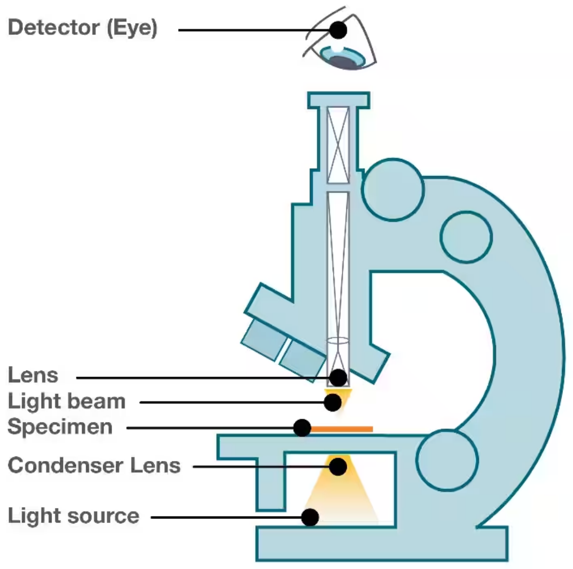


Figure 3.2.1-1. Electron Microscope Layout

3.2.3 Traditional Lensless Microscope

This microscope would use a similar design to what we are doing, but would not include the holograms and its related content. The optics would be as simple as a CMOS or CCD to image a semitransparent sample using some light source, being laser diodes or LEDs, with fiber optics.

3.3 Relevant Technologies

In this section, a brief overview of the technologies that will be used with the device is given along with a brief explanation of how these different components operate. This section is intended to serve as a quick reminder to us about the kinds of parts that we plan to use, so that we know what to research and what we need to do market analysis for. Additionally it allows us to show that we do have a good initial grasp on the components that we intend to put to use in the design and final build of the Lensless Digital Holographic Microscope, and gives us a chance to have a formal answer to prove that we know what we are talking about if asked on the spot during a demonstration, or presentation of this device.

3.3.1 Microcontroller

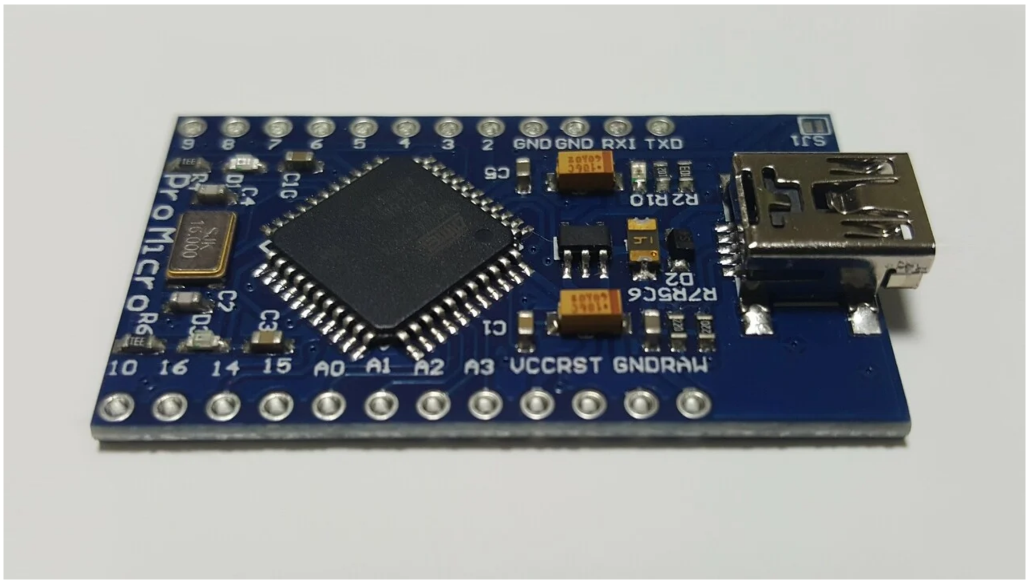


Figure 3.3.1-1. An example of a microcontroller

A microcontroller or embedded system is a small, often simpler computer, they operate on the same logic, and can be programmed like a computer, but the main purpose is to accomplish a given task that a normal computer can definitely handle while being far more efficient and cost-effective. A good microcontroller is a device that is able to do exactly what it needs to do and nothing more, otherwise, you could probably make use of a cheaper microcontroller. We intend to use an embedded system to both control the Lensless Digital Holographic Microscope, and process the resulting images, to do this we are going to need a relatively complex microcontroller with enough processing power. The microcontroller will be the center of the electrical design of the Lensless Digital Holographic Microscope and will have essentially all other components in the design somehow wired to this component, therefore it also needs to be able to handle this load of connections.

3.3.2 LEDs

A LED or light emitting diode, is exactly what the full name is, it is an electrical component that solely contributes light when given any amount of voltage, with the brightness determined by the intensity of the voltage up to an upper limit. For example, a LED with a 1 V upper limit, is as bright as it every while be, therefore, to reduce any issues with hurting the LED the voltage it should be given will at most be 1 V or lower if a lower brightness is desired.

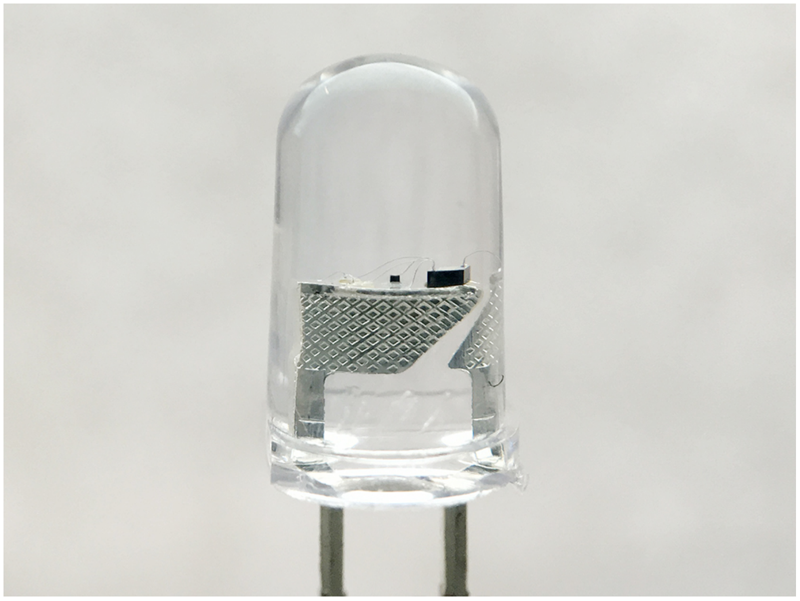


Figure 3.3.2-1. LED

3.3.3 Fiber LED Holder

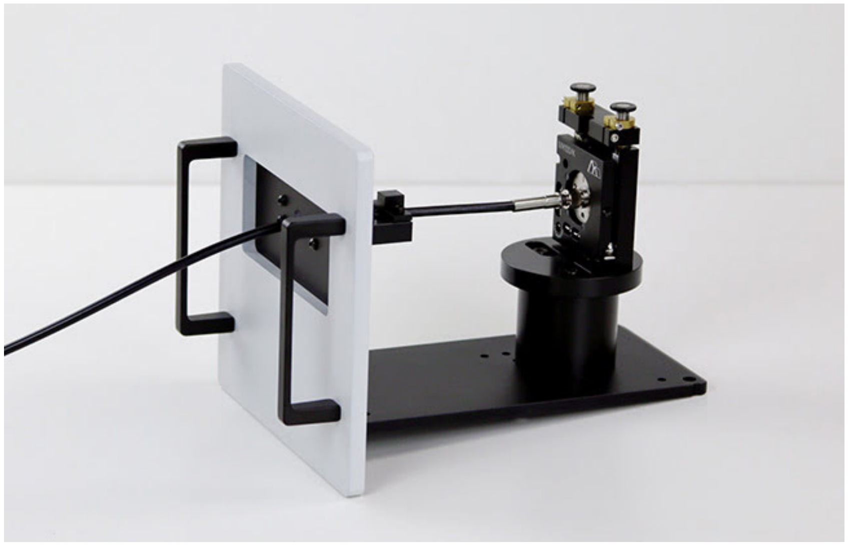


Figure 3.3.3-1. Example of a fiber LED Holder

We have modeled a breakaway piece holding the fiber chuck and LED together so once the resin is fully cured the LED-Fiber system can remain intact. This was necessary to do because the LED has a flared base, and the fiber is housed in the chuck which is then housed in the ThorLabs mount. This means once the LED is wired up to our power supply, then adjusted for maximum coupling, then held in place with resin, it is incredibly difficult to be able to pull the system out without undoing the resin binding. That is why the breakaway piece was so instrumental.

3.3.4 Resin

As seen in Figure 3.3.4-1, UV resin belongs to a group of synthetic resins and cures from the energy of the sun or devices that deal with UV. UV resin is used for sealing, bonding, and coating materials and many times the time it takes for these processes to occur is just a few minutes. Some of the reasons UV resin is used in this case is because we don’t need to mix anything, it has a quick cure time, works great for small projects which means it's great for the small components and photonic components for the Lensless Digital Holographic Microscope.

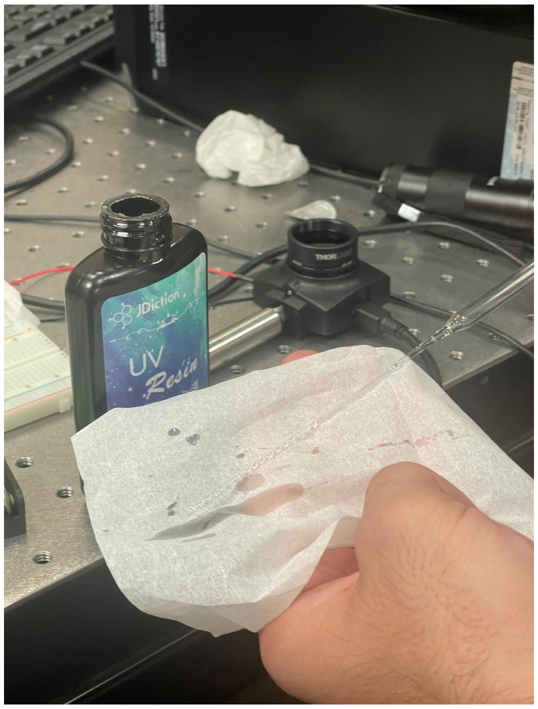


Figure 3.3.4-1. UV Resin To Seal LED-Fiber Coupling

There are different types of UV Resin. The most common synthetic resins are Epoxy resin, Polyurethane resin, Acrylic resin, and Silicone resin. One of the most common forms of resins is epoxy resin. This form of resin is a little bit more confusing in a sense that it's its own type of compound made up of resin and hardener. When these two components are mixed, a chemical reaction happens where the liquid resin gradually hard into a solid plastic. This results in a glossy and clear surface which is why epoxy resins is one of the the considered resins to be used within this project. One of the reasons that epoxy resin is being considered is because it's able to bond together Plastics, glass, and metals. But the reasoning that UV resin is being considered of more is because the pox Rising comes in two parts. One part of the epoxy is the actual epoxy resin while the other part is the hardener which means that we have to mix together the chemicals in order to create a chemical reaction to use the epoxy resin. And the Cure time is a lot longer with a minimum of five to six hours was often it being at least twelve hours before using the epoxy resin the main advantages of UV resin that outweigh and make epoxy resin the not ideal choice is that UV resin does not require any mixing. In order to cure UV resin all that is required is UV light. Compared to epoxy resin, uv resin cures in a matter of minutes so the projects won't take long and allows us to use are optical fiber components and in a more efficient manner.

One of the key problems with UV resin is that it's prone to Bubbles in particularly thin layers. But many times the advantages outweigh the disadvantages. We stated before there's no mixing needed for this type of resin and this resin can be used for a very long time as it does not cure without the aid of a uv lamp. The curing time is one of the most advantageous things UV resin has as it has such a short curing time compared to other resins.

3.3.5 Fiber Cleaver

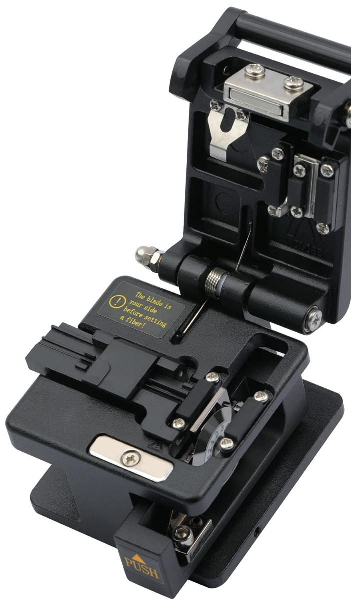


Figure 3.3.5-1. Example of a Fiber Cleaver

The fiber optic Cleaver is a piece of tool or equipment that is often used to make an almost perfect fiber and face cut. This is a very similar tool to a diamond scribe tool when you're cutting glass, a fiber's cutting wheel which is its blade makes a tiny cut on the fiber first, and then the fiber is pressed against the little cut and forced in to break at a 90-degree angle and exposes a mirror like fiber end face.

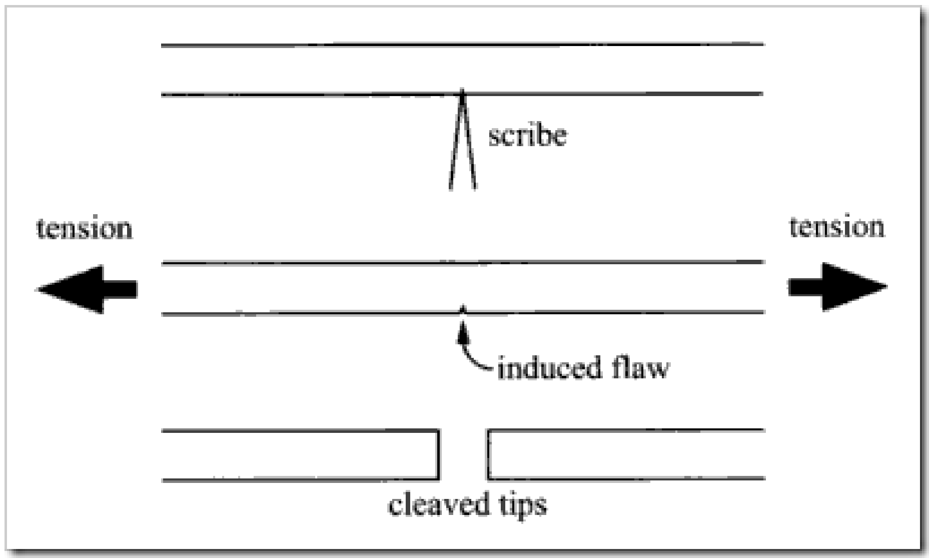


Figure 3.3.5-2. Example of a Fiber Cleaver

The reasoning behind using a fiber Cleaver is that optical fiber needs to be cleaved for fusing splicing. Optical fiber Fusion splicing nearly always requires that the fiber tips exhibited a smooth end face that is perpendicular to the fiber axis. This type of smoothness is only achieved via the fiber cleaving process as the cleaver is able to give sufficiently perpendicular and planar fiber end faces. Another reasoning is that with the fiber Cleaver and this process we are able to polish a fiber tip which can result in an even higher quality fiber and faces, but polishing requires more expensive equipment and more processing of time so it is very rarely employed for Fusion splicing and is considered as a trade-off for other things to be done.

Fiber optic Cleaver often have different designs. And optical fiber is cleaved by applying a sufficiently high tensile stress in the vicinity of a sufficiently large surface crack, which is then rapidly expanded across the fiber cross section at the Sonic velocity. This idea has different practical implementsations and a variety of commercial cleaning equipment. The different designs can be seen in the way that tensile stress is applying. Some cleavers apply a tensile stress to the fiber while scratching the fibers surface with a very hard scribing tool, usually known as a diamond Edge. Other designs scratch the Fiber surface first and then apply the tensile stress. Other designs also apply tensile stress that is uniform across the fiber cross section, while others designs bend the fiber around a tight radius producing High tensile stress on the outside of the bend. Any technique or tools is capable of good cleaves; the trick is consistent finishes time and time again. In general there are many different designs but the less costly approach requires more skill and training for the technicians to make the cleaving.

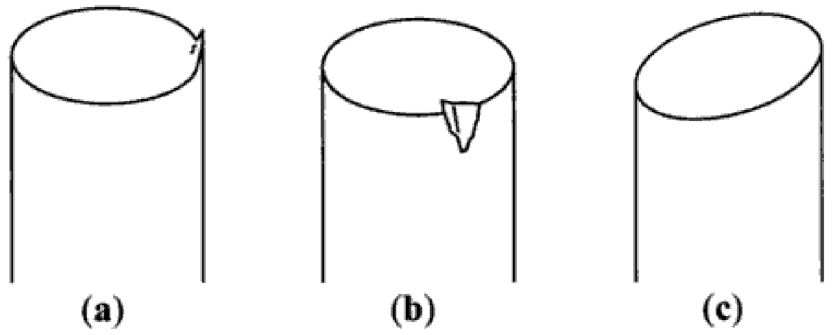


Figure 3.3.5-3. Example of a different Fiber cleave probelms (a) chip, (b) lip, (c) angle

For this project the importance of the cleave quality cannot be understated. The impact of the cleave quality on the quality of the resulting fusion splice should not be underestimated. Defficiencies within the fiber cleave are one of the most common causes for geometric deformation in the resulting supplies which is particularly onerous for single mode fiber has. The common cleave problems include the lip, a chip, and an angle as seen above. The reasoning behind this is that when we have a poor cleave quality it can compromise the performance of image processing routines that performed during fiber alignment. Cracks in the fiber's end face can lead to bubbles to be formed at the splice joint, which usually requires the splice to be remade making the amount of time of work to double and that is not good for efficience.

The necessity of the fiber cleaver can not be understated. We need clean, consistent, and accurate cleaving of the fiber to get a perfectly flat, unobstructed edge to couple as much light as possible into. This is why we are using the Newport Fiber Cleaver, 0.5 deg., 6-10 mm Cleaver.

3.3.6 Optical Table

The use of an optical table can not be understated. Having something heavy and having a pegboard function allows all of our testing to be done quickly and securely. And since everything is screwed into the board, there is no risk of pieces being lost or distances being thrown off between experiment dates. The optical board simply holds it all together while we pop it into a corner of the senior design lab. This lets us have every step of the process on the same board. From the cleavings, to the power meter readings, to the coupling, to the CMOS imaging, it is all neatly contained within the board. The optical table/board not only serves as an excellent organizational tool, but it ensures very little outside manipulation.

Figure 3.3.6-1. Example of a Optical Table

3.3.7 Heat Shrink

Heat shrink has been tested as a possible addition to strengthen the connection between the fiber and the LED. By wrapping the LED-Fiber system in heat shrink, we can not only add to the binding strength, but also protect the fiber end from accidental bumping, and from accidentally snapping off the fiber entirely.

Figure 3.3.9-1. Example of a heat shrinks (courtesy of FS)

3.3.8 Resistors

Every electrical component in the Lensless Digital Holographic Microscope uses a resistor to some extent. Their role is simple, a resistor gives us the ability to better control the flow of electricity in the system. The more complex parts will require a certain amount or range of voltage to flow into it via a certain resistance in the Vin port of the given device. Additional resistors can be added to the design of the Lensless Digital Holographic Microscope to ensure that the right amount of voltage is given to each part.

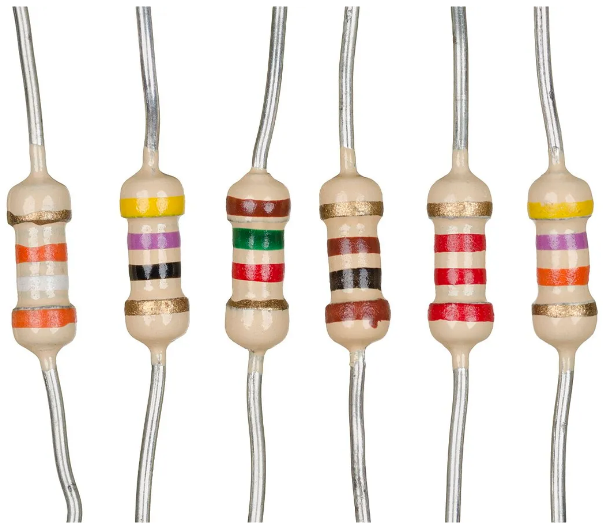


Figure 3.3.9-1. Example of a common resistors

3.3.9 Battery Pack

A battery pack can be an individual battery or a set of identical batteries used to provide power to something. These batteries can be configured in series, parallel or a combination and mixture of the two in order to deliver the desired amount of voltage. The term battery packs are usually referred to the power source that hobby toys contain as well as the electric vehicles. Battery packs can be rechargeable and contain temperature sensors which help the batteries indicate when the end of the charging cycle is complete. A well-balanced pack lasts longer and delivers better performance. Within these battery packs they can sometimes be held by a battery holder which is the housing that accepts a battery or batteries, or a separate plastic holder that is mounted with screws, eyelets, glue, double-sided tape, or other means. Battery packs are essential for products, toys, etc that contain electronic components and help them with the necessary energy to do the product's features.

For this project a simpler, more direct version of a battery pack can be considered. We have decided that a 5V at 2A battery pack is necessary and have gone ahead and researched several options to be listed in the following table. In the end we went with the RP-PB19 simply because it was convenient to develop with due its consistent output and long battery life, however any battery pack with a micro-USB output with this voltage requirement will work.

Table 3.3.9-1: Battery Pack Comparison

Part Name	Cost	Voltage Specifications	Battery Life	Connections
Folken Wall charger	$4.00	5V @ 2A	Constant	Wall to micro -USB
RP-PB19	$45	5V @ 2A	100+ hours	Rechargeble to micro-USB
A1271	$49.99	Up to 20V @ 4.8A	92+ hours	Rechargeble to micro-USB
HYD007	$14.99	5V @ 2A	20+ hours	Rechargeble to micro-USB
MPM 10-5	$13.17	5V @ 2A	Unknown	On board/ Pin-to-pin

Figure 3.3.9-1. Example of a battery pack

3.3.10 3D Printer

3D printing was a process that was devised in the 1980s and was originally known as ‘Rapid Prototyping’. It was developed as a way for companies to improve their companies efficiency and as a means to save time and money. This enabled companies to develop prototypes quickly and more accurately than with other methods and today its innovation has allowed its uses to become more diverse.

3D printing also known as additive manufacturing is a process of making three dimensional solid objects from a digital file. The action process of making a physical object from a three-dimensional digital model is typically done by laying down many thin layers of a material in succession to create the object. This material is often molten plastic. As the object is being created each layer is set, the next layer is then printed on top and the object is built up. This process of adding layers is known as the additive process as each of these layers can be seen as a thinly sliced cross-section of the object. 3D printing is the opposite of subtractive manufacturing, for example CNC machining, which is cutting out / hollowing out a piece of metal or plastic with for instance a milling machine. The use of 3D printing enables you to produce complex shapes using less material than traditional manufacturing methods thus saving you money and time. The idea of creating an object all starts with a 3D model. In order to make a print we need a digital file which you can opt to create one from the ground up or download it from a 3D library. These files can be created from 3D modeling software with the most common being Tinkercad and Solidworks. These programs allow you to create 3D models and export your model as a printable file e.g .STL or .OBJ. The importance of these digital files cannot be overlooked. These files are needed as they tell the 3D printer where to print the material. The most common file format used for 3D printing is G-code files. G-code files essentially contain ‘coordinates’ to guide the printer’s movements, both horizontally and vertically – also known as the X, Y, and Z axes. Since 3D printers are adding multiple layers together to create an object 3D printers have the ability to print layers at different thicknesses, known as layer height. This is similar to pixels on a screen. When you go up in resolution for youtube videos you get a higher resolution because you have more pixels on the screen, with 3D printing the more layers in a print will give a higher ‘resolution’. This will give a better-looking result, but take longer to print. So the trade off for a considerably better looking print is the print time. Now that you understand how printing is done and that you have a file to prepare your printer. This is called slicing. Slicing basically means that you are slicing up a 3D model into hundreds or thousands of layers and is done with slicing software. When your file is sliced, it’s ready for your 3D printer. Feeding the file to your printer can be done via USB, SD or Wi-Fi. Your sliced file is now ready to be 3D printed layer by layer. Once that is ready you decide the material you want your object to be. Multiple materials can be used in additive manufacturing: plastics, metals, concrete, ceramics, paper and certain edibles (e.g. chocolate). Materials are often produced in wire feedstock a.k.a. filament, powder form or liquid resin. Plastics are a versatile type of material, and as a result Plastic polymers are the most common material used in 3D printing allowing for many different ways of manufacturing with it 3D printing being no exception. Using other materials is possible. But many times using other materials such as metal is often niche compared to polymer materials. There are three different 3D printing technologies which are FFF 3D printing, SLA (stereolithography), and SLS (selective laser sintering).

Figure 3.3.10-1. Example of a 3D printer

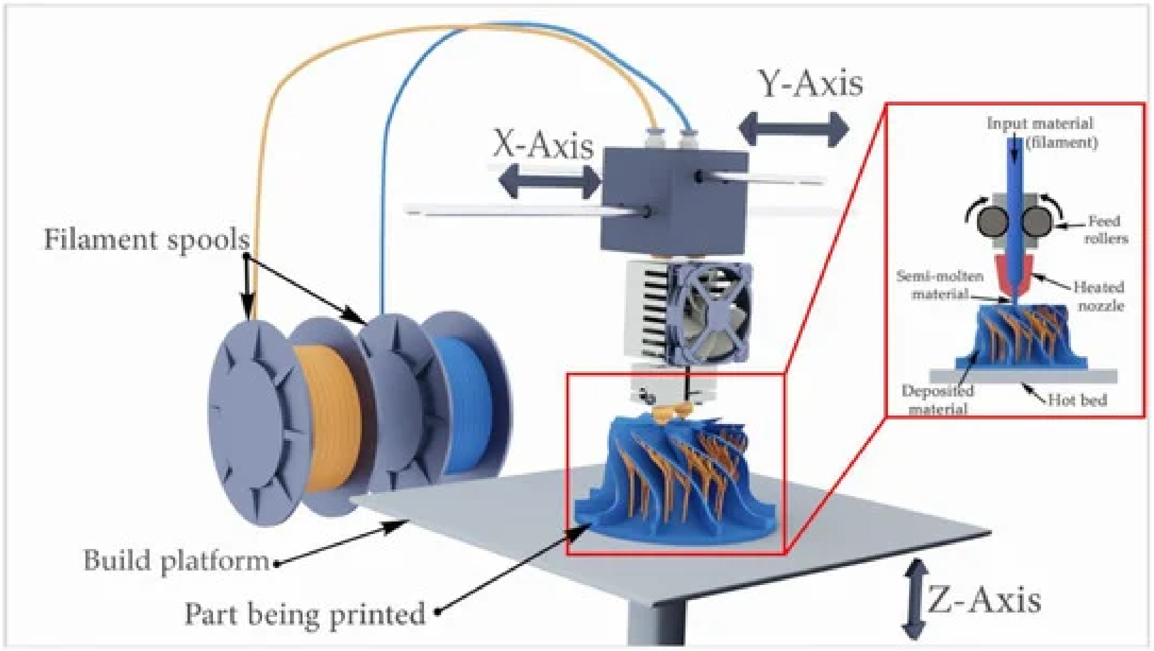


Figure 3.3.9-2. Example of a FFF 3D printing process (Elkaseer)

In figure 3.3.9-2 you will see an example of FFF 3D printing. FFF 3D printing, also known as fused filament fabrication, is an additive manufacturing process in which thermoplastic material is pushed through a heated nozzle to create objects layer by layer. The layers are placed on one another through a heated nozzle that is mounted on a motion system that moves it around a build area, where melted filament is deposited onto a build plate. Since it uses a heated nozzle the material is placed, cooled, and solidified with the build plate moving down by a fraction of a millimeter each time layer by layer until the object is complete.

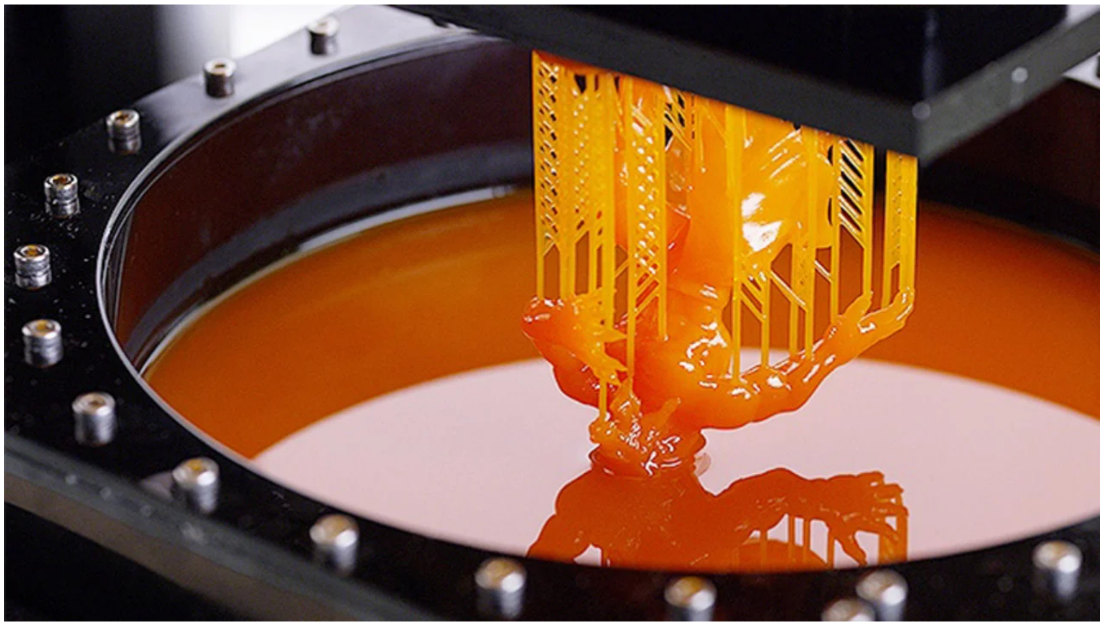


Figure 3.3.9-3. Example of a SLA 3D printing process (3D Natives)

In figure 3.3.9-3 you will see an example of SLA 3D printing. SLA 3D printing also known as stereolithography is an early method of 3D printing. This 3D printing process is done using a UV-curable resin as raw material. This resin is then poured into a glass-bottomed container, into which a build platform is submerged and the it undergoes printing using a method called photopolymerization. Photopolymerization is a technique that uses UV lasers or a UV light to initiate and propagate a polymerization reaction to form a linear or crosslinked polymer structure, to selectively cure a polymer resin to produce 3D models using UV-sensitive resin. From this method the 3D object is able to solidify by the passage of a laser layer after layer. Each time the platform gradually raises out of the container to build up the print. This method provides one of the most qualitative printing surfaces of existing 3D printing technologies.

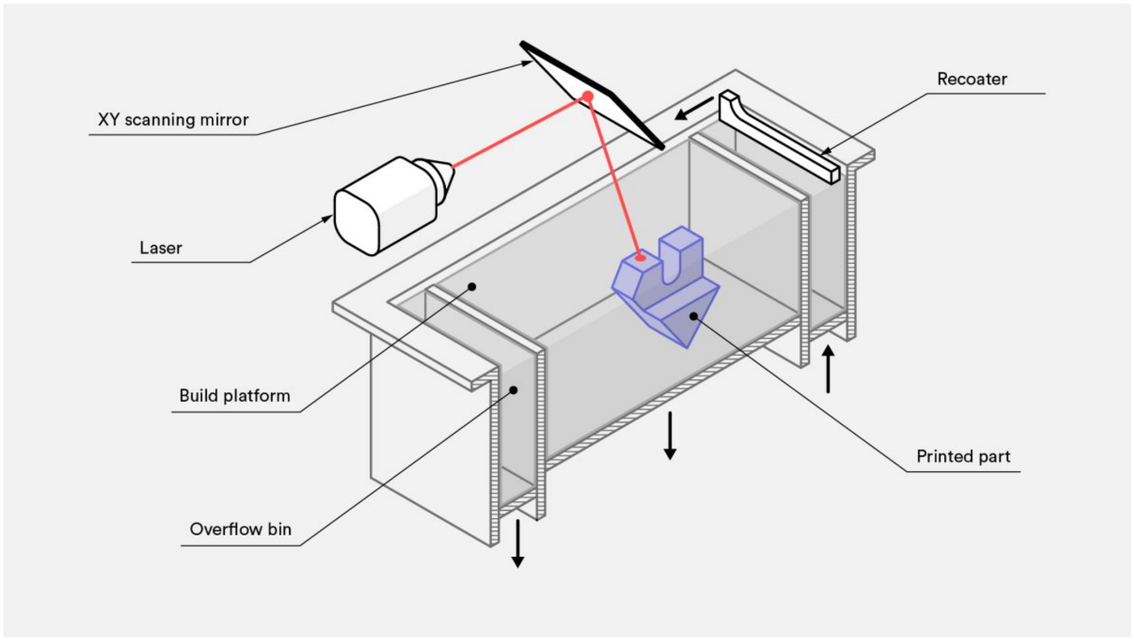


Figure 3.3.9-4. Example of a SLS 3D printing process (Hubs.com)

In figure 3.3.9-4 you will see an example of SLS 3D printing. The last 3D printing method is the SLS 3D printing. In this method of 3D printing the printer uses powdered raw material which is typically a polymer. This powder then sits in a container, where a blade distributes a thin layer of material onto the build area. SLS 3D printing then uses a high-power laser to sinter small particles of polymer powder into a solid structure based on a 3D model. This laser fuses the small particles of the material together to form a single horizontal layer of the part. The container continues this process moving a fraction of a millimeter to start a new layer with the blade swiping across the build area to deposit a new layer of raw material. This process is repeated until the finished object is created.

3.4 Holography Research and Theory

Holography in its simplest form is simply recording/writing a wavefront/interference pattern, and then reconstructing/reading the original wave. This way information about what the wave has passed through can be read by an observer. Holo being greek for full and graph being greek for write, means to encompass something's information fully. Holograms initially used to be recorded on film, where the interference pattern would etch into the holographic film. Then with a reference beam, one could reconstruct the image fully in the film. The way holograms work is by shining an original reference beam, originally a sodium lamp passed through multiple filters to get spatial coherence and temporal coherence, but now it is lasers because for the most part they come out spatially and temporally coherent. The original reference beam is then sent through a beam splitter and then the original beam continues to the imaging plane. The other beam would then go around, hit the object someone would like to image and then that beam would be passed on and it would interfere with the original beam. That interference pattern is then recorded. The reading portion of the holograph happens when you then shine the original beam once more upon the film. The etchings provided a full view of what the object imaged was. The interference recorded on the film is a combination of four terms, the original beam, the holograph, the holograph twin image and the scattered information. The scattered information can be ignored because the little amount makes it negligible. The first term can be removed because we know the original beam we're sending out. There are multiple ways to remove the twin image.

3.4.1 Fresnel Diffraction

Fresnel Diffraction occurs when the distance from the point source to the subject, or the subject to the imaging plane have similar measurements as the subject itself. Fresnel diffraction is necessary to our imaging. The holographic information needs to come off fresnel diffraction to be recorded.

3.4.2 Interference Patterns

The interference patterns are what are recorded

3.4.3 Off-Axis vs Fourier Transforms

Whereas one way to remove the twin image would be using off-axis imaging, what is more likely the route we took will be using fourier transforms to get a symmetric fourier image, then we delete one of those images, and then undo our fourier transform to get simply one image. Off-axis using a pinhole and a larger hole might be something we consider doing in the future.

3.4.4 Reading Hologram with Digital Reference Beam

Whereas historically one would have used the original reference beam mentioned above, with software today we can record the information at the imaging plane and digitally reconstruct the original beam. Using that digitally reconstructed beam we can digitally reconstruct the subject we are imaging.

4.0 Research and Part Selection

This section will discuss the research that was conducted for our Lensless Digital Holographic Microscope project in order to ensure that the best parts are selected for our desired resulting product. By doing thorough research on each component we choose, we can ensure that our product will function as we expect it to. The best-chosen component from each of these sections will be further outlined in section 6.0.

The following section contains research for:

- The Microcontroller
- The Microcontroller comparison
- LEDs
    - Electrical and Photonic sides
- CMOS Sensor
- Software Tools
- Photonic Theories and Equations
- Fiber-Optic Cables
- 3D Printing

4.1 Microcontroller

This part really is a core piece of the Lensless Digital Holographic Microscope, it should be a more cost-effective and efficient piece over a standard computer. The tasks that it needs to accomplish on a basic level are as follows: store and send the images from the censor to another computer, and controller for the LED array. However, we would like to task the microcontroller with more intensive responsibilities so that the Lensless Digital Holographic Microscope is much more self-sufficient. Therefore, the tasks that the microcontroller we choose should be able to handle, in addition, to the following task: process the images from the sensor into a readable image, and then send the image via a Bluetooth connection to a connected device with an app that we plan to develop. Additionally, there isn’t a limit on speed now so optimization is not exactly a priority now, since we would like to see it work once, even if that one working run takes about five minutes to completely process. At this stage of development, we know the importance of just getting anything to work will be more impactful than spending an amount of time on optimization, big or small.

A development in our research has shown that some additional thought should be put into how exactly the microcontroller receives the information from the CMOS sensor. Doing the full development from scratch might not be an option, even given the amount of time that we have left. Therefore, we may need to look for another kind of solution for handling the process of converting the output of the CMOS sensor into an actual image. This could involve either a specific change in how we approach a microcontroller or an overall workaround.

4.2 Microcontroller Qualities

This section highlights the qualities of the microcontroller that we are looking for.

4.2.1 Cost

We have a budget that we would like to keep low but considering the shortages from the pandemic the optimal microcontroller may not be attainable to us, however, we are looking for a microcontroller that can accomplish the bear minimum of what we need it to do at the lowest cost. This aspect will be listed in the comparison table, as well as a discussion in the individual part's summaries, found in Table 4.2.10-1.

4.2.2 Power consumption

This shouldn't be an issue considering the low number of components in this design, but this metric will ensure that we always have a practical microcontroller for our onboard battery. This will be compared in the part comparison table to see if there are any outliers that need to be considered.

4.2.3 Power-output

The LEDs may require a higher amount of power than a standard microcontroller pin, so if there is a microcontroller that can provide a higher amount of power, which can fulfill the higher quality LED then it might be worth more to the Lensless Digital Holographic Microscope then initially thought. Additionally, we want the highest brightness for the LEDs possible so a marginal increase of the output pin power would enable the LEDs to consistently meet those power requirements. Since this is an aspect that will have a work around it will not be in the unit comparison table, found in Table 4.2.10-1.

4.2.4 Pin amount

This count is more for the number of pins that the device supports for input and output signals rather than just the total amount of pins. Because many CMOS sensors tend to have 12 output pins and the LED array may need as many as 25 pins, the number is going to need to be a bit higher than the average simple microcontroller. Additionally, not all the pins in the pin count on any part can be used for this purpose easily. The real pin count work is what is measured instead we are listing whether or not the individual part has the amount of pins needed to fulfill the role. Therefore, this will be listed in the comparison table, found in Table 4.2.10-1.

4.2.5 Processing-speed

While we do not have a specific standard that we want to meet in terms of compiling speed, although we attempted to optimize the speed to as low as we can get it, the exact processing power was noted but the main contributor to the speed of the process was likely the algorithms we used rather than the microcontroller we used. To show any possible correlation with the cost this is listed in the comparison table, found in Table 4.2.10-1.

4.2.6 Size

While the exact size of the final product is yet to be determined, the size of the microcontroller will not be the deciding factor to increase the size of the product, in short, any microcontroller that will be listed here will more than likely be able to be included in the final design without much issue, but it will be helpful to have the exact dimension here going forward. Although an edge case example might be present if the specs are interesting enough to discuss. This will be mentioned in the summary of the part, and since there should always be a work around it will not be in the comparison table.

4.2.7 Coding-Language

In case there is a restriction on the microcontroller about what languages it can or can't use, it should be noted. We are currently looking for one that can primarily run python and secondarily run C+. This can be an important distinction between the microcontroller and therefore will be featured on the comparison table, found in Table 4.2.10-1.

4.2.8 Bluetooth compatible

Whether the microcontroller is already Bluetooth compatible would certainly mitigate the amount of additional work and components that would be necessary to get the Lensless Digital Holographic Microscope operational. It is a part of our goal to make the product Bluetooth compatible since it is one of the best ways to make the product easily accessible and convenient. It appears that there is a module that we would have to look into to make a non-prebuilt Bluetooth-compatible microcontroller feasible to accomplish our goal. This will be a metric when considering a microcontroller and if there is a microcontroller that would otherwise be good for the Lensless Digital Holographic Microscope whether the additional module is feasible will be discussed in its summary section. While this is a nice accessory to have it will not be on the comparison table, since this is closer to a stretch goal than a requirement of a microcontroller. A recent development has shown that we could get the same effective result in a less complex way, for example with the same internet connection a bluetooth device would need we can instead email the final results to the user, this method would not need the additional component if necessary.

4.2.9 USB compatible

The Lensless Digital Holographic Microscope will be using a CMOS sensor, which might be easier to implement via USB cable. While this functionality might not be used due to a restriction of the microcontroller or if at the time, we feel like a pin-to-pin connection is easier, or more feasible, it is still worth considering since this gives us another option to connect the microcontroller to the CMOS sensor in case there is an issue. This is a feature that will greatly impact the implementation of a given part and should be only mentioned in the summary of the part, since there could be a different kind of work around to consider.

After further development of our research, a USB port went from something we would need to work around to vital since a pin to pin connection with the CMOS sensor did not seem to be feasible given our time constraints. The raw input from the CMOS sensor is far more complex than initially thought and at the rate the data transmission is meant to be we could not reasonably expect our code to efficiently compute the input, if we could even manage to make it work without outside help.

4.2.10 Microcontroller Selection

Here is a table that is meant to display the key aspects of the microcontroller. The columns are not ranked in any order; however, cost is still the most important aspect for us. The aspect that we have chosen to present here are cost, power consumption, number of pins, processing-speed, coding language, these are in short, the aspects of the part that we can not change, they are like this when we would choose to buy them, and they will be like this when the final design is done. All other aspects of the microcontroller will still be featured in each part's individual summary below when necessary.

Table 4.2.10-1: Microcontroller Comparison

Part Name	Cost	Power-consumption	Number of Pins	Processing-speed	Coding-Language
STM32WB55VCQ6	$10.92	3-5.5 V	Too few	64 MHz	Linux device
Raspberry PI 2B	$35 - $200	5 V	Too few	1.5 GHz	C+
RM48L952DPGET	$44.65	1.2-3.3 V	Enough	220 MHz	C+
MSP-430F5519	$6.81	1.8-3.6 V	Enough	25 MHz	C+
XMC4800F100 K2048AAXQMA1-ND	$35.64	3.13- 3.63 V	Enough	144 MHz	Open

4.2.11 Microcontroller Summarie

While there is clearly a different amount of importance in each of the aspects of a microcontroller that were discussed here, an exact breakdown is not really warranted and will be shown in the pros and cons of each microcontroller. The only thing that is worth noting is that cost is the largest deciding factor. Therefore below will be a summary of each individual part to show that there is some additional forethought into each part, and how they would try to fit into the design of the Lensless Digital Holographic Microscope, and if they can be implemented easily or not.

4.2.11.1 STM32WB55VCQ6

In short, this is the best simple microcontroller that has access to Bluetooth capability naturally. All other simple microcontroller's don't seem to have that capability so it is interesting to see that this one is only priced at $10.92, the issue here would be the lack of Pins and no USB support, going off the data sheet there is just too few pins that we need in the design therefore can't be a candidate unfortunately. Additionally, the processing speed might be at a lower standard and the development of the code for this device seems like it would be hard to develop for due to the lack of visual feedback and low impact on the market so we wouldn't be able to find out if a problem we have has a simple solution. Upon further research, this part won't have the capability to process the images and since there aren't enough pins this can't really be considered for any kind of role.

Figure 4.2.11.1-1. Example of a STM32WB microcontroller

4.2.11.2 Raspberry PI

Not only does this fulfill nearly every metric that we need to fill, but it also accomplishes the additional stretch goals without the need for an additional component. The only negative is that no matter which raspberry PI we go with there will be some aspects neglected, either the HDMI port or the additional USB ports. Also, since this is one of the most used microcontroller developments for the code, implementation of this part should be easier since we would likely not be the first people to have the same issue with this part. Therefore, this is effectively a price ceiling on the role of microcontroller, and this is a product that won't be going out of stock so this option will always be here for us, so the only way we don't go with this part is if we find another part that can feasibly fulfill the same requirements, and cost less, even if there are more cost to implement another microcontroller. Another, small issue would be these microcontrollers look like they would want to use their individual on board battery, there might be a workaround but if there isn't and this is the only option, then the real circuit design would end up very different from the theoretical design. After additional research this part is much more complex then first thought and can handle the installation of free use software to greatly reduce the amount of work from scratch that would be needed to otherwise make use of the images of the CMOS sensor.

When this part was initially being researched it was believed that any relatively recent version of the Raspberry PI would be usable as long as it was at least part of the 2 series, however further developments have shown that the Raspberry PI 4 would be especially useful since we might now need a second USB port, which is worth the additional $10 dollars for the upgrade. However, due to a shortage in supply the only way to get a Raspberry PI 4 of any model would not be on an official online distributor, and the prices would be too high. So, after additional research we settled on a Raspberry PI 2, this will be at the lowest price on digikey and presents a bit of a complication. This part can handle the image processing, and has multiple USB ports, but needs an additional microcontroller to handle the LEDs since this part has roughly eight pins to be used in this many, besides the fact that it would be part to put the LED array task on another part anyway. Additionally, this part does not have the built in Bluetooth feature that the most recent parts have so that will require some more work to solve, but the trade is worth the difference in cost, and the safety we have in obtaining the Raspberry PI 2 over the preferred version. Also the work to upgrade the microcontroller to the new one would be easy since their design is standardized between each other.

Figure 4.2.11.2-1. Example of a Raspberry PI 2

4.2.11.3 RM48L952DPGET

This is a higher quality industry standard microcontroller, the processing speed and pin count is sufficient, and the voltage requirement should be a no issue. The cost is a bit close to the most that we would be willing to spend especially since we would need to use a Bluetooth adapter. However, while the microcontroller itself is perfectly good for what we need it to do, it has been utilized on similarly complex tasks before, there is quite little content about it available to us so if we run into any problems it would be on us to solve them. Overall, it is a solid piece that would be able to get the job done, but would be a difficult process to figure out every step to a complete product. However, there might be difficulties properly setting up this part to process the images from the CMOS sensor therefore can't be considered for that role, however could handle the control of the LEDs sufficiently.

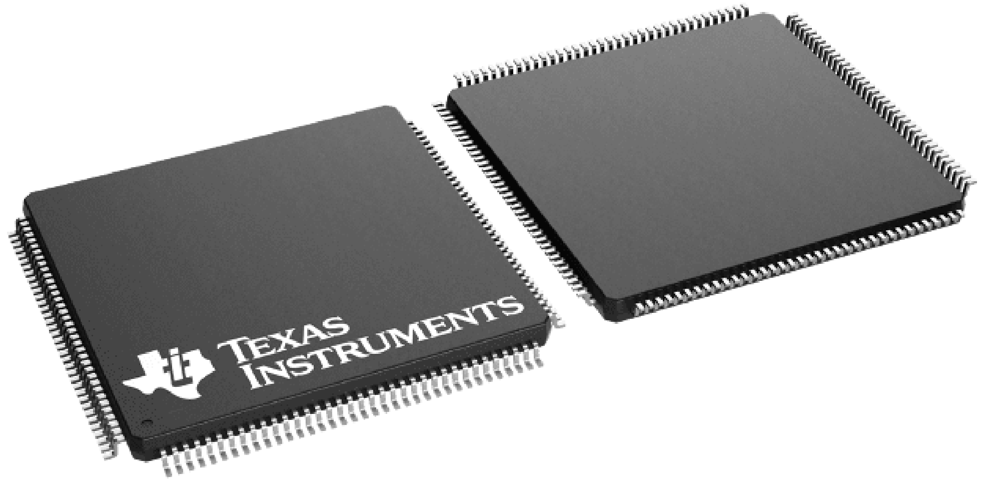


Figure 4.2.11.3-1. Example of a RM48L952DPGET from Texas Instruments

4.2.11.4 MSP430F5519

This is in short, a simple option, one that is relatively cheap, but powerful and large enough to have enough pins to handle 25 LEDs. While it won't be able to do any image processing this can be a more supportive option for the Lensless Digital Holographic Microscope, rather than the brain of the design, this part should be able to be implemented as a side microcontroller to just handle a role, to put less stress on the part performing the image processing. The cost is a bit of an issue when thinking about this part in that way, since it is just controller LEDs, there should be cheaper microcontrollers that have the requisite amount of pins to do the task. However, this component is benefiting from our history with this part, we know the limits and we know how this part works, therefore since this part can be treated in the development of the Lensless Digital Holographic Microscope as if it were just another assignment since there were similar project regarding LEDs and pin to pin connections, we can be confident that there are no issues with this part. Additionally, we can experiment with the low power mode of this microcontroller very early to help the power flow of overall design in case that is an issue during early prototyping.

Figure 4.2.11.4-1. Example of a MSP430F5519

4.2.11.5 XMC4800F100K2048AAXQMA1-ND

This is more of a representation of the median between the low and the highest end that we are willing to purchase. There is a sea of middle-of-the-line microcontrollers available to us in this price bracket, and they all on the surface have the qualities that we are looking for, the real problem between all of them is that there is a sea of products and each of them would require much more research to determine if any of them would be better than another. It will take a standout feature or resource that will determine if this is the route that we would like to take, but for the moment they all represent the issues of the unknown. We are always going to have some issues with the microcontroller, either in coding or implementation, and with boards like these we are going to be on our own, and just have to figure it out, but the difference here versus a higher quality microcontroller is the answer to our problem might turn out to be that this microcontroller isn't worth the effort and we should just go for something more expensive that is easier to develop for.

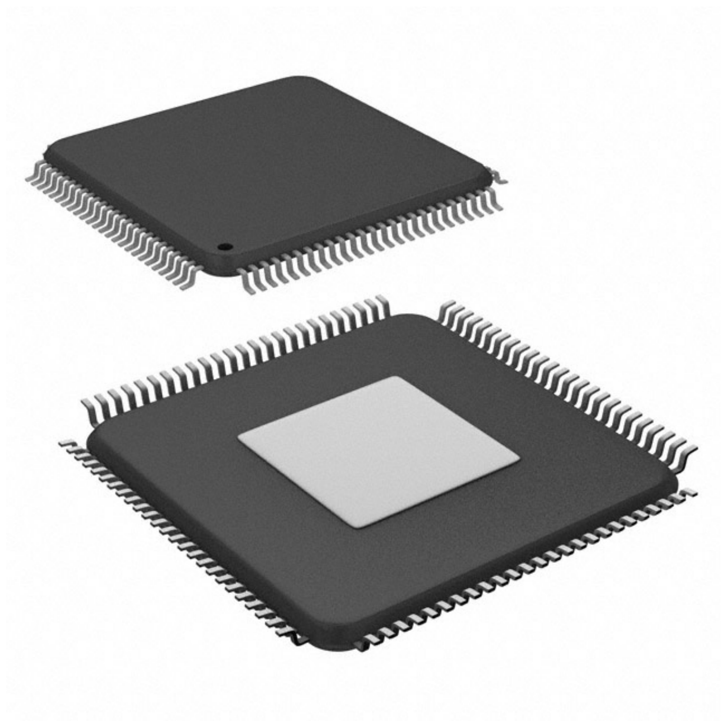


Figure 4.2.11.5-1. Example of a XMC4800F100K2048AAXQMA1-ND (Digikey)

4.2.12 Microcontroller Decision

Before detailing exactly why we did and didn't go with a certain microcontroller, the following table 4.2.12-1, will show the main deciding reason for why they won't be a part of the design. The table is meant to show a quick bullet point worth of reason, there was a lot to consider when comparing all the different microcontrollers and how each could be implemented or the difficulties we would have to work around to create an interesting design, a more detailed reason can be found below, but this table will be able to convey the lasting memory we have about each of them, and why we choose not to go forward with them. In conclusion, this are all fine parts, if the needs of the design were slightly different we could see myself working with any of these parts, and Iwashonestly happy we have done as much market analysis for this project as we have, since we was now fulling confident in the electrical components that we have chosen, and will be able to make my case on this matter when asked in the next semester.

Part Name	Main Reason
STM32WB55VCQ6	This was just here to see if we can get a cost-effective Bluetooth option, this could even be used to get and transfer an image due to its lack of memory. Additionally this board doesn’t have enough pins for the LED array.
Raspberry PI	Great computer, a poor microcontroller, doesn’t have enough pins but can be used for computing the images.
RM48L952DPGET	Maybe halfway to getting enough memory to do the image processing and would require a pin to pin connection so won’t work for that role. While being overqualified for controlling the LED array, this alone would be able to control five of the Lensless Digital Holographic Microscope with a single board, cost efficient but not practical.
MSP430F5519	This is the microcontroller that we have the most experience working with, so we know it wouldn’t be able to handle the image processing by a long shot. However, this could handle the LED array quite well.
XMC4800F100 K2048AAXQMA1-ND	This was originally picked to be a middle of the road, before we understood the complexity of the code that we would need to write to get this to work, if it was possible. The higher end version of this part has already fallen short of what we need for this project, so the more budget version of that part fails by extension.

Table 4.2.12-1: Microcontroller Decision

Upon further research it has been found that a microcontroller with a slower processing speed wouldn't just be very slow but would require redoing a lot of work from scratch, since the input from the CMOS sensor would be too much for a simple microcontroller to handle. Originally the intent was to wire a pin-to-pin connection from the microcontroller to the CMOS sensor thinking that we could time the input for the multitude of incoming pixels to create an array of binary values and reconstruct the picture from that. While that is how it would work if we had the time and the ability to do that work from scratch, in reality we need a computer with access to pre existing software to process the images. The Lensless Digital Holographic Microscope does still need a microcontroller to control the LED array, and the microcontroller still needs to be accessible to the user, so ideally be Bluetooth compatible in some way, and still have enough pins to control the LEDs naturally.

Therefore, the parts that we would choose are the Raspberry PI 2 for handling the input from the CMOS sensor, and the image processing, as well as a MSP430 for controlling the LEDs. For the latter the MSP430 has enough pins to handle 25 pin to pin connections reliable and since this parts role will just be to toggle on and off the LEDs the processing speed of this part will be more than enough to fulfill this role, in addition we can say personally that this part will be able to do the task efficiently. Between the part's low power mode and reset pin the MSP430 will be able to easily accomplish the task efficiently, additionally there is a great amount of synergy between this microcontroller and the other microcontroller responsible for the other tasks of the Lensless Digital Holographic Microscope, the Raspberry PI 2. The MSP430 Launch board can use a USB connection to power and program the microcontroller and the Raspberry PI 2 has two USB ports therefore the Raspberry PI 2 can easily control the MSP430 microcontroller via simply resetting the MSP430 code which would toggle each LED in the array creating different viewpoints for the CMOS sensor that will be sent to the Raspberry PI 2 microcontroller.

The reason why we are choosing the Raspberry PI 2 microcontroller is that it is simply a comparatively cheap computer that we can install in the Lensless Digital Holographic Microscope as if it were a microcontroller. The part can have an imagining software installed on it as if it were a normal computer making what could have been a near impossible task for this group within the time limit, nearly done, since it will be considerable easier to have the Raspberry PI 2 use the third party software to create the images for later processing then having us take the raw input from the CMOS sensor and create the image somehow before we can even process it into our final output. We are confident that the Raspberry PI 2 will be able to handle the input of the CMOS sensor since it has a USB port that will be compatible with the CMOS sensor that we have chosen, has the storage space for the imaging software, and has the processing speed that once we have the pictures stored on the microcontroller we can process the images back to the final output that we want to deliver.

Figure 4.2.12-1. Example of a MSP430F5519

Figure 4.2.12-2. Example of a Raspberry PI 2(digikey)

4.2.13 Microcontroller Implementation

The implementation of the Raspberry PI 2 into the overall design should be straightforward. There are two ways we could power this electrical component, which could change how the system overall receives power. The two methods are to either have a pin-to-pin connection for the power, which would be better if we had one battery pack for the entire system. This method would be more challenging to wire, and more dangerous on the components, but would be a challenge worth completing for this project. The other method would be having a dedicated battery pack for this part, this would be safer for the microcontroller and more consistent and since this is the part that needs to be constantly running the safe option might be worth considering, and this gives us the option to power other parts such as the other microcontroller through the original with no issue.

As for how this part would connect to the other parts of the design of the Lensless Digital Holographic Microscope, the Raspberry PI has two USB ports making it very easy to receive the information from the CMOS sensor. Additionally, the second port can connect to the other microcontroller. This would still require a pin on the Raspberry PI to connect to the MSP430's reset pin, but this method would make it so the battery pack only provides power to the CMOS sensor and the Raspberry PI, testing is needed to see if the MSP430 can provide enough power to the LEDs for them to operate at maximum brightness, if the MSP430 cannot the array of LEDs would also need to be powered by the single battery pack, since only one is used at a time it shouldn't need too much power. In conclusion these two microcontrollers have a great amount of synergy, and are easy to implement into the Lensless Digital Holographic Microscope's design rather seamlessly, and we look forward to testing these parts in the general design. The microcontroller will tell the CMOS sensor to take an image directly after each LED flashes, continuing until the sequence has completed for all of the LEDs.

Featured below is a PCB diagram of how we would integrate the MSP430F5519 to have the micro USB header, and the female pins that it would need to properly function in the final build of the product.

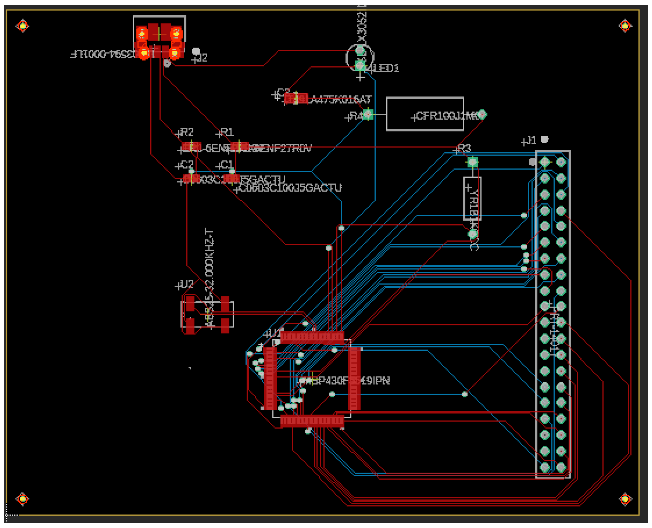


Figure 4.2.13-1. PCB design with MSP430F5519

Alternatively from this design, we have a MSP430FR6989 development board to use as a backup, it meets all the same requirements that our chosen MSP430 microcontroller meets, but is more expensive, and not significant to the overall design.

4.3 LEDs

These small diodes are incredibly important to the optics side of the Lensless Digital Holographic Microscope, and are relatively easy to understand for the electrical side of the device. For now, we plan to use 15 LEDs for the upcoming demo and 25 for the final build. On the optics side this means adjusting multiple LEDs very carefully to ensure they perform optimally to give the maximum amount of brightness. Meanwhile electrically speaking this is a diode with a clear purpose and a voltage requirement to be filled. This is a topic best discussed with a bit more focus on the independent fields which is what the following sections aim to do. The physical dimensions of the LED need to be taken into account, as our LED array design is dependent on the diameter of the LED, since we are 3D printing a flat plane with holes in it the size of the LED diameters.

4.3.1 Electrical side of LEDs

While the LEDs will be important to the optical portion of the product, they naturally need to be wired correctly first, the specifics of how much voltage the chosen LED will need can change what we choose to do, so both options will be detailed here. Additionally, the goal for the LEDs is to provide the highest amount of brightness while being able to toggle them on and off, two simple tasks but for a simple component that is all we need them to do. If the voltage from the power source that we are using to provide power to the LED is somehow far too high we can implement a voltage divider similar to Figure 4.3.1-1, this is a textbook voltage divider with a diode so it doesn't really match what we would do one to one if we needed to lower the voltage but the concept is similar enough.

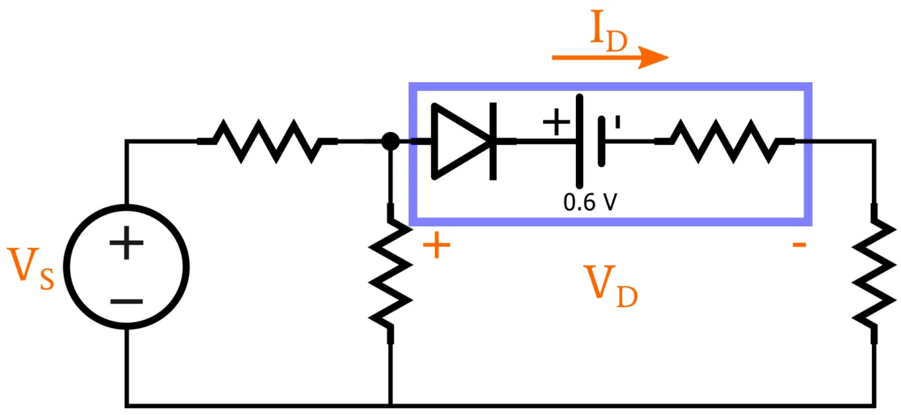


Figure 4.3.1-1. Voltage divider for 1 LED

In short, there are two kinds of LEDs that we can go with, the easier choice to implement could prove to harm the performance of the product. These would be a simple LED which would cost about a dollar per LED in an array of 25, the other choice would be an LED designed to be used in optics, which is more expensive and requires far more energy to be sufficient. The former of the two would result in a design where each LED in the array would receive its power from a pin on the microcontroller which can be easily toggled. In this design, there would be a voltage divider for each LED to ensure that the voltage is relatively stable, which can result in the LEDs brightness being configured to better suit our needs, although to configure it someone would need to change the resistance of the other resistor in the voltage divider, and it probably isn't worth it to get an ohmmeter for each LED. The alternative LED would require a considerably higher amount of power, however from what we have seen there would not be a need for multiple pins for each LED, so the wiring would basically be the same if the microcontroller can provide an adequate amount of power, which could be done and there will be a section of the microcontroller selection that will show if the pins can provide an amount of power for the higher tier of LEDs. In this configuration, if the microcontroller cannot provide enough power the pin of the microcontroller would instead flip the switch gate on and off in the same manner we would program the LEDs in the other example, and the power would instead be provided by the onboard battery.

Whether we use the standard LEDs or the higher quality LEDs the resulting circuit board will not affect the way the microcontroller will be programmed to handle the job, so there may be a change in development if we decide to go the cheaper route and the quality is not up to a standard that we would like to meet. The product's size makes it so this change will not make the physical wiring harder, even though the new 25 switches might make the wiring area more cluttered, the current design will leave us the room to do either build. In addition there might be a concern that effectively 25 new elements in the design could be costly, however this is a time where buying in bulk for the Lensless Digital Holographic Microscope is actually applicable for us, there for this element will likely end up being the second cheapest part of the design, right after the wires of course. Unfortunately the way this method looks on the EAGLE CAD schematic is a bit messier, and leads to a lot of clutter since they are all on one section of the schematic, however we don't really have a choice if the method with the switches is the only way to get maximum brightness out the LEDs electrically we have to do it.

4.3.2 Optical Side of LEDs

We decided to go with LEDs for their low-cost, high output-price ratio, and high temporal coherence. Having monochromatic light is an absolute necessity because this is what allows the gabor hologram to be measured. The phase interference with the same frequency yields the information we seek. If there were other wavelengths tossed about in the mix we would have interference on multiple aspects of our project. This is why holographic film usually records in only one wavelength.

4.4 CMOS Sensor

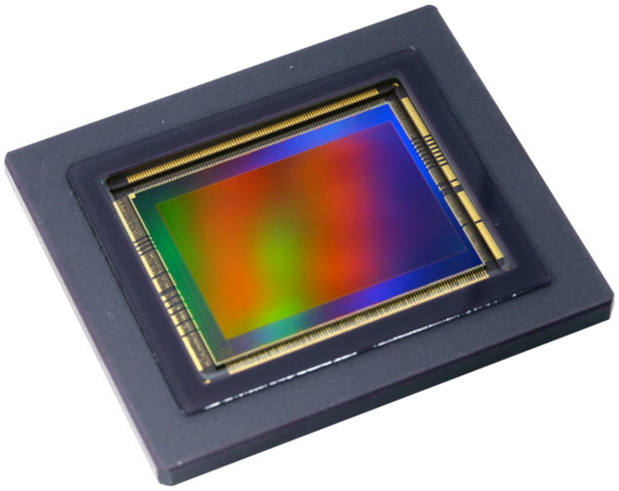


Figure 4.4-1. Example of a CMOS sensor

A good sensor is one of the most important components of the device. To help narrow our research we are going to be defaulting to the industrial standard and the kind of sensor that was present in the research paper that inspired this project. Therefore, we tried to find a CMOS sensor to meet our criteria and perform and a standard with which we are happy. In short, we searched for a CMOS sensor that can successfully create an image of what we are trying to visualize, have that image stored on the microcontroller for further processing, and have an intuitive way of sending that information to the microcontroller. The former can be done by simply looking into the datasheet of any given CMOS sensor, and finding the specs of the given part, the harder part is trying to solve the issue of how to get the information. This will require more research not only on the CMOS sensor but also on the microcontroller and will be a topic of discussion in future papers.

4.4.1 CMOS Qualities and Elements

The aspects of a CMOS sensor that we are looking for are listed in detail below and will be directly compared in a table at the end, found in Table 4.4.10-1. Any aspect of a CMOS sensor will be shown in that table useless otherwise stated in the section regarding said aspect.

4.4.2 Cost

A good CMOS sensor is going to be the most expensive part of the Lensless Digital Holographic Microscope, but there are plenty of cheaper CMOS sensors that not only could result in a minimum viable product but with some extra work and a reasonable budget should perform at a standard that we would like to achieve. However, this is the most impactful part of the product, and having a good CMOS sensor that fulfills our requirements is much more important than a sensor that we believe should be able to work for us, this is the most important part and getting the wrong one will set us very far back, therefore price that we are willing to pay is much higher than any other part. Additionally, supply may be an issue, aside from the normal pandemic shortages there is a different kind of supply problem that we could be facing, since this is a simple project, an older model might be perfect for what we plan to accomplish. Even more so since an older model of a CMOS sensor could have more content made about it, researching how to get past a problem might be easier for an older model since more people would have seen that problem before. In short, all the models listed here should be stock at the time of writing but if there is a notable example of a perfect CMOS sensor that we can't use there will be a section going over that part and why if it comes back into stock why it should be used over the choice we were forced to make. This aspect will be compared in the table below, found in Table 4.4.10-1.

4.4.3 Pixel Resolution

This is the least important contributor to the quality of the image, in short, we would like the most amount of pixels for the lowest cost that we can find. There are no restrictions on formatting, so we do not need an exact ratio for this device. Upon further research, we are looking into a higher quality CMOS sensor than initially thought so the pixel resolution will always be high, but there will still be comparison points between them, found in Table 4.4.10-1.

4.4.4 Pixel Size

This is one of two aspects of the CMOS sensor that we need to be overly critical of. The pixel size will need to be at a certain value or lower. For now, while we research what would be best and what we can accept, the value that we looked for was 3.5 µm or lower. A direct comparison between all candidate parts can be found in Table 4.4.10-1.

4.4.5 Pixel Pitch

This aspect is weird, and we need to keep this aspect in mind, but the range of acceptable ranges or pixel size to pixel pitch ratio is important, however, this information does not seem to be clear or available on the average CMOS sensor. Pixel pitch is defined as the distance between two-pixel centers, which in this case could create a gap in the final picture since a small gap in the pixel size could create a visible gap in the larger picture. Therefore, we searched for this information, and we wrote more about the pixel size-to-pixel pitch ratio in the summary section of a candidate CMOS sensor. However, there may be a workaround via pixel super-resolution which can be implemented into The Lensless Digital Holographic Microscope by placing the LEDs with a bit more care, this workaround is based on the work by Wu and Ozcan (3). Also, due to the discussion around this aspect and how some of the candidate parts don't even explicitly state their pixel pitch this aspect won't be in the comparison table, however, will be noted in each part's individual summary.

4.4.6 Frame Rate

Due to the nature of the CMOS sensor, this component is normally meant to be an active recording, but since we know that you will get higher quality with a static image that has been compiled with each picture that is caused by an LED creating slightly different perspectives. Therefore, we do not necessarily need a high or specific frame rate, we would just be using a specific frame as a picture to correlate with an LED from the array flashing. Meaning that a high frame rate is not as useful as you would normally want, it just changes the calculations that we would need to do anyway to accurately take and store the pictures. A higher frame rate may allow us to get the task done faster but may not be necessary since the point where the frame rate matters is at the beginning of the process, and a difference in the pictures being stored completely would only be a second at most if the frame rate matters. However, this will still be noted when finding candidate CMOS sensors since this metric can influence the price. This aspect will be measured in the comparison table, found in Table 4.4.10-1.

4.4.7 Power Requirement

Power should not be an issue at all, but it is worth keeping track of in case the final build is more power intensive than previously estimated, found in Table 4.4.10-1.

4.4.8 USB Plug-in Available

A pin-to-pin connection might not be the best way to use this component, and unfortunately, we were not be able to know until we have both parts ready to see what we must go with. Whether a CMOS sensor comes with a simple data out pin, a USB plug already attached or a pin to USB plug-in available all should accomplish the same roll equally. This is not a real metric like the rest of the specifications for a CMOS sensor and requires a bit of research if they are compatible with the additional plugin so it will be noted in the summary section of each part. Also, if the CMOS sensor does not have a USB connection natively it is cheaper than the competitors that do so the additional cost of the plug-in would be negligible since we are technically spending the money that we saved.

In short, after further developments in our research a USB plug in seems to be necessary due to the complexity of the CMOS sensors raw input. In order to compute this data we would need to map each individual pixel, and know which color it is picking up, a filter can be used to make it so we would only need the green pixels. However it would be very hard to distinguish where the starting pixel at the beginning of the transmission since I2C connects won't work due to the speed of process, and the average CMOS sensor normally transmits in UART instead. Making it a headache to try to even begin to understand where to start on the implementation, without a USB plug-in, while it would be simple and fun to wire the pin to pin connection, and we are confident we can get that part done rather quickly, that would be the last successful step in the implementation of the CMOS sensor for months without outside help.

4.4.9 CMOS Sensor Selection

Unlike the priorities of the microcontroller, the CMOS sensor does have values that must be achieved, therefore the few aspects that we need will be considered more strongly. These aspects are pixel size, pixel pitch, pixel resolution, and cost. Cost is a given this part on the market has models that cost upwards of one thousand dollars and as little as ten, both of which can be utilized to accomplish this task, in short, we kept an eye on the cost and likely selected the cheapest CMOS sensor that can meet the following achievements. Which is pixel size, and pixel resolution; pixel size needs to be a certain metric for what we are trying to image, and a better resolution will just increase the quality of the image for the end user. On the other hand, aspects like frame rate and USB plugin are only noteworthy from a comparison point of view if a CMOS sensor meets only these standards, then it would not even be considered for the final build.

Note: for some reason a lot of CMOS sensors on the way that we trust to deliver a quality part, do not even have a real datasheet easily or publicly available, all the parts here will have a corresponding data sheet, since this isn't just a few CMOS sensor on sale on Digi key, they all should be able to have a quality worth discussing for the Lensless Digital Holographic Microscope, and in order have any form of quality the part needs to have a datasheet.

4.4.10 CMOS Selection

Table 4.4.10-1 CMOS sensor Comparison

Part Name	Cost	Power consumption	Pixel Resolution	Pixel Size	Frame Rate	USB
NOII4SM6600A-QDCOS-ND	$264.00	2.5 – 3.3 V	2210 x 3002	3.5 x 3.5 µm	5 fps	No
MT9P031I12STC-DP	$26.76	1.7 – 1.9 V	2592 x 1944	2.2 x 2.2 µm	53 fps	No
08529-01	$995.00	5 V	5472 x 3648	2.4 x 2.4µm	20 fps	No
Arducam AR0234	$109.99	3.3 V	1920 x 1200	3 x 3 µm	30 fps	Yes
4016C002	$329.12	3.3 V	2592 x 2056	3.4 x 3.4 µm	120 fps	No

4.4.11 CMOS Summaries

The following section will showcase all the candidate CMOS sensors in an individual way to show that each part was given consideration for not only how they can fulfill the role of a CMOS sensor but also how it could be implemented. The summaries are meant to show more than a pros and cons list, they are intended to show that we have taken exactly what the aspects of each CMOS sensor can provide for this product and how we could implement them into the design. For example if it would be straightforward to implement, would be need to find a work around, or if the part is good but there is a reason why it can't work for us because of a certain, either flaw or quirk, that makes the part something that we would like to use but simply won't be able to.

4.4.11.1 NOII4SM6600A-QDCOS-ND

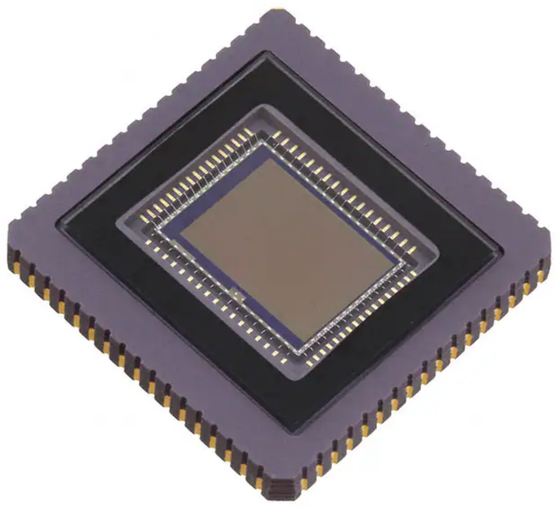


Figure 4.4.11.1-1. NOII4SM6600A-QDCOS-ND (digi-key)

It is unfortunate to say that this falls in around the middle of pricing for CMOS sensors, however this CMOS has one quality that make this part stands out, this part explicitly states its pixel pitch which is an aspect of the CMOS that is quite important for The Lensless Digital Holographic Microscope moving forward despite it not appearing on every CMOS sensor datasheet. This part's pixel resolution is also quite good, a bit above the average competition and is to be expected for the price. However, the frame rate is a bit of an issue, while this aspect was intended to be a low priority, we did not expect any CMOS sensor or any device to be lower than 20 fps, again it is not an issue the factors that will affect the overall performance time of the Lensless Digital Holographic Microscope will mostly be the actual code, but this is surprising low. Also, the output pins seem to be either 10 or 12 bit which is what we are planning to accommodate, and the power is well within a comfortable range as well. Also, this part not having an entry for its pixel pitch is a bit annoying but not the biggest problem. Overall, a part that should work because of the availability of information about it, also with a price tag as high as this it should be able to accomplish its task.

4.4.11.2 MT9P031I12STC-DP

This is a unique part, it is the only part that is single purchase, and currently in production that has a CAD model available for open use. If we do not use this part in the final build the CAD model for this part would probably be used in the schematic, after some adjustments are made of course. The pixel resolution and pixel size are in the acceptable range of values to meet our standards. Not having an explicit pixel pitch is annoying for now, but after some more research on this part exclusively that might not be an issue. Also, the frame rate and power requirements are quite good, well within our standards. The main issue is the cost, while getting a CMOS sensor that will work for us for only $26.76 almost seems too good to be true, it will take some time to see if this CMOS sensor can actually complete our tasks, and since this is the part that we are most willing to spend more on, to ensure that there won't be a massive issue later down the line, this part will be under a different kind of scrutiny since we won't be looking for a sign that this part should be able to work, we will instead be looking for an indication that this part won't be able to work for us for any kind of reason.

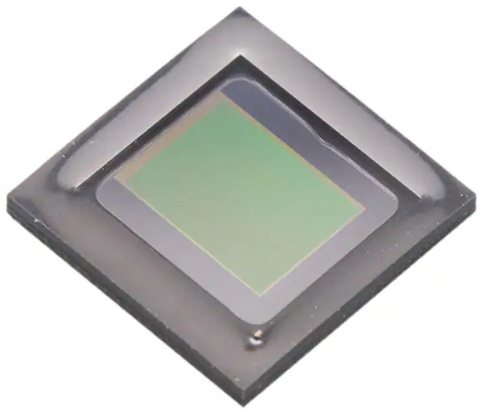


Figure 4.4.11.2-1. MT9P031I12STC-DP (digi-key)

4.4.11.3 08529-01

Figure 4.4.11.3-1. 08529-01 (digi-key)

This is representative of the highest costing CMOS sensor that we would be willing to use in the Lensless Digital Holographic Microscope. For once the power requirement is a bit on the high end so that might require slightly more complex power solutions than a double A battery pack. The pixel size and pitch are available and meet the requirements, and the pixel resolution is quite impressive, to the point that the low frame rate makes sense since we would need more time for the transmission to go through and the next frame is further away than normal to accommodate that. Also, this part's primary outsourcing would be via a USB port, so slightly different from a lot of other CMOS sensors but can be used, either by getting a microcontroller that supports that functionality or by finding a workaround. Altogether, making this a CMOS sensor that we can be confident in making work. The only real downside is the datasheet for this part leaves a lot to be desired, this is clearly meant to be a part that is just plugged in and good to go, but it would be better if a lot more information was on offer, seriously CMOS sensor costing one percent of this total cost can have a datasheet with more than twenty times the length of content, and information.

4.4.11.4 Arducam AR0234

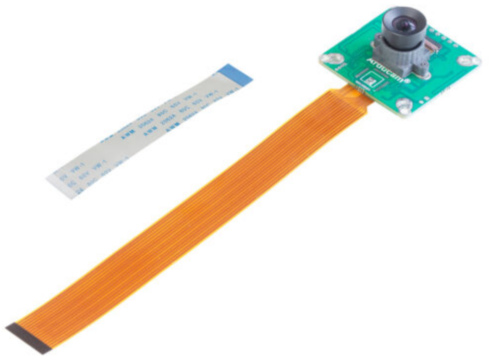


Figure 4.4.11.4-1. (ArduCam)

The only aspect of this part that is novel is the fact that despite having a publicly available datasheet it wasn't on the store site for this part, however this is a part worth discussing despite that minor annoyance. In short this is the cheapest part that we have managed to find that is immediately able to make use of a USB plug in, alternatively, with the provided flex cable, in fact this part was designed for development using a Raspberry PI microcontroller, if we decide to use it. Otherwise, we can go with a different similar CMOS sensor on that store front, this one was chosen to be representative of its family of parts because of that correlation. In addition, the specs of this part are all up to par with our standards, the pixel size and pitch are usable, the pixel resolution, while on the smaller end is completely ok for the price, and the frame rate is much higher than we need it to be. Altogether this is a good part at an affordable price, there just might be an issue with the implementation of this part into the optics portion of the Lensless Digital Holographic Microscope, but a work around might be discovered, so if additional research is put into this part, this might be a great budget choice that actually works.

4.4.11.5 4016C002

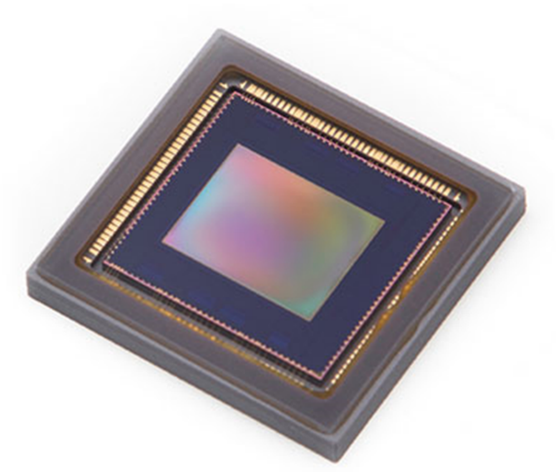


Figure 4.4.11.5-1. 4016C002 (digi-key)

This CMOS sensor strikes a balance of simple and easy to implement that all the other parts simply couldn't. This is a CMOS sensor that has a USB accessory without any strings attached, this is simply a high-quality CMOS sensor that does exactly what we are looking for and nothing more. The pixel size is almost too high, and the pixel pitch isn't accessible now, however, while those are the most important aspects of any CMOS sensor that we are looking into, this part should just barely meet that standard. Meanwhile, all the other aspects are incredible, the frame rate and pixel resolution are great, especially considering the cost, and the power requirement is doable in the current system. All of this and this part comes with a USB port available, at this cost we can be confident that this part will be able to do exactly what we need it to do.

4.4.12 CMOS Decision

The following table, 4.4.12-1, is intended to showcase our current thoughts on each CMOS sensor. A more detailed description of each can be found below, however it can be helpful to have a more shortened reasoning available. We have put in a lot of thought into how the design would be with any of these CMOS sensors and this table is meant to show the ending thoughts on these components.

Table 4.4.12-1 CMOS sensor Decision

Part Name	Main Reason
NOII4SM6600A	Low specs for the cost, and no USB support
MT9P031I12STC-DP	The spec requirements for what we need just ended up being way higher than initially expected when first considering this part.
08529-01	Very expensive while requiring a risk to get this component to work
Arducam OV928	Reasonably priced for the specs but needs additional work to make function.
4016C002	Good specs if a bit overpriced but very accessible.

There are plenty of CMOS sensors on the market, most of the CMOS sensors here are our attempt to get multiple good representative examples, so when we are referring to any given part the same can be said about any similar part. For that reason, it would be beneficial to give an example of the reasoning why we aren't going with the other four parts, instead of just stating which CMOS sensor we are choosing and why. That said we have chosen the 4016C002 for many reasons, and from this point onward will be referred to as the chosen CMOS sensor. Now then the NOII4SM6600A-QDCOS-ND is in short the most out of place CMOS sensor that we could find, with the highest pixel size, average pixel resolution, and the lowest frame rate, there is no reason this should cost as much as it does when the chosen CMOS sensor only cost about $60 dollars more and has a much better frame rate, a better pixel resolution, and a slightly better pixel size, while also having a USB cable accessible out of the box. In conclusion, the NOII4SM6600A-QDCOS-ND is too similar in price to the chosen CMOS sensor while being so much of a downgrade that we have chosen to ignore this part.

Next the MT9P031I12STC-DP aside from these rapidly going out of stock at the time of writing, this part in short presents multiple problems when considering how to implement this part. In short, we would have to use a direct pin to pin connection, while trying to condense the pixel resolution from 2592 x 1944 to 720p since that was how this part was designed, giving us no way to work around. Meaning that this part presents more trouble than it is worth making the choice between this part and the chosen CMOS sensor clear, it is worth the price increase to circumvent this issue. For the 08529-01, firstly all the CMOS sensors costing more than $700 are mostly the same if one was chosen since it was the middle of the pack. The main issue with CMOS sensors of this price range is they are basically all cameras, they are intended to be an actual stand-alone device, not a component to be use in a project like our Lensless Digital Holographic Microscope and would require additional work of removing what was intended to be a luxury at an additional cost to be used in this design. This brings an interesting issue of stress since we would be buying a rather expensive part and risking that money to just make it simpler, while voiding the warranty so a minor mistake, and in this scenario, this is the most likely mistake to make over the course of the project. Alternatively, if we try to make the part work as it is then we would be making a camera with a lens be the most important part of the Lensless Digital Holographic Microscope, which is just counterintuitive at that point. Additionally, the Arducam OV9281 1MP wasn't chosen for much of the same reason, although with this part we could experiment with removing the lens of this camera at a much lower risk. In short, those last two CMOS sensors can do the job we need it to do, and with their USB plugins can be implemented into the design of the Lensless Digital Holographic Microscope but require more work to remove a feature of the part to then have the same process of being used in the actual product.

This was our original line of thought supporting the 4016C002, however the item was incorrectly listed on digikey, the evaluation kit was listed, meaning this part didn't have the USB port that it was advertised to have. In short a $6,000 dollar evaluation board was being advertised when they would only deliver the bare CMOS sensor, while providing no documentation about how to use this part on a PCB, effectively making it useless. Therefore we returned it and instead went with the AR0234 which was what it was advertised as, meaning we were able to get it running almost of of the box with the raspberry pi, and begin testing almost immediately.

4.4.13 CMOS Implementation

Any CMOS sensor only needs two separate tasks to be done to be implemented into the design, three if we can get a mount in the actual body of the device, however for testing its implementation just two things need to be done. The AR0234 needs to be powered somehow and needs to be able to send its data to the microcontroller in some way. One of the reasons we choose this CMOS sensor is because it comes with the USB accessory out of the pack, making it very easy for us to just connect our chosen CMOS sensor to our chosen microcontroller since it has a USB port open for this. This could also translate to how we power the CMOS sensor, however we need the testing to see if the microcontroller can simply power the CMOS sensor via the same USB cable. However, if that is not the case we originally planned to just have a voltage divider to power the CMOS sensor with the same on board battery as the microcontroller.

4.5 Software Tools

A software tool was essential in our project allowing us to work more productively while helping keep track of what needed to be done and when it needed to be done. These tools were just as essential as the electrical and optical components in our project, the selection of the correct software was crucial as it helped our project not only run smoothly but meet the requirements necessary for the final design. In this section, we display all the different software tools that could have been utilized to design, develop, and prepare our project and show which languages we ended up using.

4.5.1 Programming Languages

Being able to make the correct selection for the programming language for the Lensless Digital Holographic Microscope project was key because the programming language was crucial in properly creating a successful working system overall. This selection helps yielded solutions that are concise, easy to debug, easy to extend, easy to document, and easy to fix.

This section highlights all the different programming languages that were taken into consideration of being utilized within the project. Each section helps show the background of the languages and the different methods and features that allowed the programming languages to be unique from one another. This was important because it allowed the Lensless Digital Holographic Microscope team to have a method to design, develop, and prepare our project to get to the finished product that was desired. The detailed comparison table at the end helps fully list, display, and of course compare all the unique and distinct differences between the programming languages that could be put into good use highlighting which language we went with.

4.5.1.1 Python

Figure 4.5.1.1-1. Python logo

Python is known for its object-oriented, high-level programming language that coincides with dynamic semantics. Programming using Python has become very popular in recent days because of how straightforward it is to use. This is largely due to the syntax being easy to understand and because several platforms, such as UNIX, Macintosh, and Windows, can be used to take advantage of the large expanse of libraries available. Another benefit of using Python is that it’s not always necessary to write a lot of code for projects to function. For example, a simple convolutional neural network can be written with 50 lines of code or less. This makes Python a great option for imaging projects that are relatively small. Python also employs features that make it more efficient to use. An example of this is that it supports garbage collection. This is a feature not seen in C++. Programmers have enjoyed this emergence of Python because of the increased productivity it provides and it's easy-to-interpret syntax. With Python there is no compilation step allowing for the edit-test-debug cycle to be incredibly fast and easy as a bug or bad input will raise an exception and never cause a segmentation fault.

Another significant benefit of using Python is that it can also be integrated with C++ and other languages. The ability to use Python alongside other languages makes it a very flexible choice for our final version of the Lensless Digital Holographic Microscope and programming of the CMOS sensor and stretch feature of mobile phone applications. Python is good for the CMOS sensor because of its ability for scripting or glue language to connect existing components together. Python's simple, easy-to-learn syntax emphasizes readability and therefore reduces the cost of program maintenance.

4.5.1.2 C

Figure 4.5.1.2-1. C logo

The C language was developed by Dennis Ritchie between the years 1969 and 1973 at AT&T Bell Lab. The C language is a high-level, structure-oriented language that focuses mostly on the implementation of functions to perform most actions within the program. It is important to understand that the C language does not support polymorphism, encapsulation, and inheritance which means that C does not support object-oriented programming like other languages such as Java. C supports a procedural programming structure which means that it can be used to create stepwise procedures to develop applications but with complex problems of reactive code it may not be the best language. So this means that the C language is a programming language that is used to help aid the hardware and firmware with interactions. C does not support any type of information hiding and it has Built-in data types that are supported in C.C is also a function-driven language meaning functions in C are not defined inside structures and meaning that C is a programming language built over and around logical functions or procedures within its programming structure. C requires a compiler to compile and run the code, meaning that the entirety of a program’s code will be converted into binary code and then executed by the machine (the machine running the code) every time the program is run. Additionally, the C language does not support inheritance, instead, it focuses on data methods or processes. The C language also has a strict language syntax that must be learned and followed, similar to C++ and MATLAB but unlike python (which is written more similarly to a spoken language).

4.5.1.3 C++

Figure 4.5.1.3-1. C sharp logo

The C++ language was developed by Bjarne Stroustrup in 1979. The C++ language is a very similar programming language to C where it high-level general-purpose, object-oriented language that is known for its speed and efficiency. It is also known as the an extension of the C programming language, or "C with Classes". In contrast to the C language, C++ supports polymorphism, encapsulation, and inheritance because it is an object-oriented programming language through the inclusion of classes. So with this super version of C, the C++ language is able to support both procedural and object-oriented programming paradigms when it comes to programming. In this language, the use of data and functions are encapsulated together through objects since object-oriented programming is used. One of the best benefits as well is that the C++ language had built-in and user-defined data types that are supported in C++. Unlike C, function and operator overloading is supported by C++ and it allows for C++ to be object-driven. C++ allows for inheritance and focuses on data instead of focusing on the method or procedure that the C language does. The only thing that turns our group away from using this language is the fact that the CMOS sensor seems to be using the C language but with C++ it maybe a language to be considered for use for the stretch feature of a mobile application and other stretch goals.

4.5.1.4 C#

Figure 4.5.1.4-1. C++ logo

Created in early 2000 by Microsoft's Anders Hejlsberg, C# also known as C “Sharp”, is an object-oriented and type-safe programing language that enables programmers to build many different types of secure and robust applications. One of the most important things to note about C# is that the programming roots of this language are that of the C family which means that it is familiar to C, C++, and Java. c# is also a component-oriented programming language as well. This means that C# allows you to use an existing component without caring about its internals, as long as the component complies with some predefined set of operations or interfaces. So this programming language emphasizes the creation of reusable code in the form of components that can be interchanged. This style of coding heavily relies on polymorphism, encapsulation, late binding, inheritance (through interfaces), and most importantly binary re-usability. Some of the best features that C# has is garbage collection, nullable types, and Lambda expressions. Garbage collections allow programmers to automatically reclaims memory occupied by unreachable unused objects. Nullable types help guard against variables that don't refer to allocated objects. C# is also great for exception handling as C# provides a structure and extensible approach to error detection and recovery.C# offers Language support for asynchronous operations and provides syntax for building distributed systems. C# is widely has been used by programmers for developing desktop applications, web applications, and web services. C# is also somewhat used in the game development industry to improve the quality of games for the respective developers. Since CMOS sensors use .NET Framework redistributable assemblies C# is an option to look at for the programming approach for the CMOS sensor later on in the project development cycle. C# would allow for us to read the sensor data by creating a simple console application.

4.5.1.5 MATLAB

Figure 4.5.1.5-1. C++ logo

MATLAB is a programming and numeric computing platform designed specifically for engineers and scientists. With this program, they are able to use an array of built-in tools and functions in order to analyze data, develop algorithms, create models, and design systems and products that transform our world through computational mathematics. The main language that MATLAB used is a matrix-based language allowing the most natural expression of computational mathematics. The advantages of using MATLAB for the product are that you can analyze data, build and develop algorithms, perform performance testing, and create models and applications for engineers to use to take your ideas from research to production. The advantages also include the fact that MATLAB’s functionality can be expanded by additional tools and has a wide range of usability. MATLAB helps vastly with the number of photonics equations we are implementing with the product.

4.5.1.6 Programming Language Comparison

In this section we will use the tables below to compare all the exclusive and distinct features and specs between the different coding languages discussed in the sections above that could be put into functional use for the Lensless Digital Holographic Microscope project. The reason we found these languages and compared them was to understand which language best suits our project as different programming languages are designed for different purposes and have their own strengths and weaknesses. Through this research we were able to compare and see which programming languages would be the right fit for a particular task our project may handle such as the hologram reconstruction which was very math heavy and the pixel super resolution. Another thing that this comparison allows us to see was the ability for us to compare existing libraries, frameworks, and communities which means that for what we wanted to code the code may already exist and be open source allowing the coding production time to be decreased in the sense of having to rehard code in the different functions and formulas that would be used on the photonics side. Overall by comparing the different programming language options for our project we were able to make an informed decisions on which language would be best for a particular task or function within the project. Coming to this conclusion was possible because we were able to be taking into consideration factors such as understanding each languages intended use, paradigm, advantages, ecosystem and libraries, type discipline, parameter passing methods, disadvantaages, and what each language was most suited for and how it would best suit the project.

One key thing to highlight is that all the coding languages that are mentioned are free for use, but MATLAB. For this reason the cost comparison is kept out of the table comparison since MATLAB is $49.99 for students but with our school we get the software through UCF apps. Since the material is so vast and would not fit within one table the following tables will be setup as a continuation of each other and the comparisons for each coding language will be in a similar manner. The chosen coding language(s) that will be used in this project will be explicitly noted in the tables and explained further in the design section of this report.

Table 4.5.1.6-1: Programming Language Comparison

	MATLAB	Python
Paradigm(s)	Procedural, imperative, array programming	Functional, imperative, reflective, array
Standarized	No	No
Type Discipline	Dynamic	Strong, Dynamically typed
Parameter passing methods	By value	By value (Call by object reference)
Intended use	Numeric computation and visualization	System, Embedded
Best For	Engineering and scientific applications like data analysis, signal and image processing, control systems, wireless communications, and robotics	Data analytics, machine learning, even design
Advantages	Highly suitable for scientific and technical data analysis, Ease of use, Platform Independence, Device-Independent Plotting, Graphical User Interface, MATLAB Compiler	Enhanced productivity, easy to learn and write, dynamically typed, vast library support, hassle-free portability, High versatility and data visualization
Disadvantages	Interpreted language (may execute more slowly than compiled language), Cost (not free)	Limitations of database, slower runtime speed, high memory consumption, runtime errors

Table 4.5.1.6-2: Programming Language Comparison

	C	C++	C#
Paradigm(s)	Imperative	Imperative,object-oriented, generic	Imperative, object-oriented, generic, reflective, functional, event-driven
Standarized	Yes	Yes	Yes
Type Discipline	Static, weakly typed	Static, dynamic, weakly typed	Statically typed, dynamic (for interop)
Parameter passing methods	By value, by reference (through pointers)	By value, by reference (through reference types)	By value, by reference (through managed pointers [explicitly in, out, or in-out])
Intended use	System, Embedded	System, Embedded	System, Embedded
Best For	Scripting of system applications, web services, and web applications	Supporting object-oriented programming features	Development of desktop applications, web services, and web applications
Advantages	Fundamental block for many other programming languages, portable language, middle-level and structural language, built-in functions	Mid-level programming language, high portability, fast and powerful, standard library, multi-paradigm	Effective memory management, fast and powerful, standard library, object-oriented
Disadvantages	Insufficient memory management, absence of exception handling, run-time checking, lack of constructor and destructor	No support for garbage pickup, uninsured system security, can cause overload functions	Run-time checking, test and correct errors, no garbage collection, unsafe, complex

4.5.1.6.1 Programming Language decision

For the project the group ended up choosing the languages of MATLAB and Python to code the necessary functions and methods of the Lensless Digital Holographic

Microscope project. The reason that MATLAB was chosen was for its many open source existing code that dealt with holography as well as twin image artifact removal. The language was also selected due to its ability to handle difficult computations with its numerical computing MATLAB proved to be a no brainer choice in which coding language we would choose for the reconstruction of a hologram. MATLAB also having existing toolboxes that had dealt with photonic methods and holography also meant that our programmer would have an easier time implementing the methods as the group intended. Overall, MATLAB proved to be a very powerful programming language for numerical computing and simulation. With is very user-friendly syntax and vast ecosystem of toolboxes and existing open source code it made our numerical computations, data analysis, and system modeling and simulation it simplified the implementation of this code for the projects function making MATLAB a suitable choice. The other language we decided on using was Python. Python was used because we could code in small programs within the raspberry pi to have live feeds of what we showed during our demonstrations and allowed us to sync the CMOS's capture time with the LEDs being lit. Python was also used because of its simplistic syntax allowing for anyone with little to no coding experience to understand what was being done if they looked at the code. Python also offered a large amount of computer vision and image processing functions for our Multi-frame pixel super resolution. This paired with its large community of open sourced projects and helpful community allowed our programmer to best implement the photonic methods as they were intended to be implemented. Overall, Python's popularity, rich ecosystem of libraries, flexibility, extensibility, deep learning capabilities, visualization and plotting support, and interoperability made it a suitable choice for the Lensless Digital Holographic Microscope project.

4.5.2 Programming Editing Software

Being able to correctly select which IDE or programming editing software that was going to be used was imperative in allowing the team to write, manage, debug, and execute the code for the digital aspect of the project to function efficiently. The selection is crucial as programmers can have the same set of capabilities in one place without needing to constantly switch tools to allow for tasks to be completed in a faster time frame. Tighter integration of development tasks meant boosted developer productivity.

This section displays all the different Programming Editing Software that were utilized to design, develop, and prepare towards our final product. Within this section we contain a detailed comparison table at the end to fully list, display, and of course compare all the unique and distinct differences between the programming language editing software that ultimately lead to the decision of what editing software our project plans to implement highlighting which editing software we ended up chosing.

4.5.2.1 Eclipse

Eclipse is an integrated development environment or IDE that provides a comprehensive facility for computer programmers for software development. This IDE uses many various languages such as JAVA, Python, C/C++, Ruby, and many more. It contains a base workspace and an extensible plug-in system for customizing the environment. It is the second-most-popular IDE for Java development. This IDE is used for mainly JAVA-based programming and Java applications. One of the benefits of using this IDE is that the source files, artifacts, and images someone is working with can all be stored in one palace within the workspace. The user has the complete functionality to select the name of the workspace and manage the projects in a single workspace. One of the best benefits of Eclipse is that it's free. It provides the editors and views for navigating between IDE and changing the content. These different views are known as perspectives in Eclipse IDE. For every particular group of data, separate views are provided to the user. Every view has its own hierarchical data and when the user clicks on another view the data hierarchy is changed and gets displayed for that particular view. For e.g. the project explorer view displays the list of all projects on which the user is currently working. This IDE would be very helpful in the coding of the CMOS sensor.

4.5.2.2 Sublime Text

Originally released in 2008, Sublime Text bills itself as a “sophisticated text editor for code, markup, and prose.” At its core, Sublime Text is akin to a Swiss Army knife that can be applied to any use case or problem involving manipulating Text for its variety of coding languages it covers. Sublime Text editor is a sophisticated text editor which is one of the most popular text editors in the world used by many developers. Coding developers frequently use this editor to write code in many programming languages such as Java, C, or Python. One of the benefits of Sublime Text is that it also supports front-end languages for web development such as HTML, CSS, and JavaScript. One of the key features with Sublime Text is that it is cross platform meaning that it supports Mac, Windows, and Linux, and it’s distributed as “shareware,” which means it’s free to use with the occasional purchase pop-up. This pop-up causes you to see that they use the program on a frequent and consistent basis, the licensing of the product must be purchased one time at a rate of $80 but one thing to note is that the free version has many features already seen within the product they ask you to purchase. The features that Sublime Text has are vast. This vast line of features helps develop once vigorous coding processes become effortless. These wide features include Syntax Highlight, Auto Indentation for the structural organization of the key elements in the code, File Type Recognition for identifying file formats using signatures and extracting metadata and text content from files for the creative and organizational orientation of the code being used, Sidebar for allowing developers to have access to all the code involved in the project in an easy to navigate place, Macros, Plug-in and Packages that make it easy for working with code base. Other powerful features include multi-line editing, build systems for dozens of programming languages, regex find and replace, a Python API for developing plugins, and more. Sublime Text’s ease of use, excellent UI and UX, Supports Plugins and integrations, command palette for easy accessibility to snippets, codes readability, search code in user customizable directories, language support, and Package control allows for developers to code and program projects in a Fast and efficient with the help of a ton of tools to use.

4.5.2.3 Notepad and Notepad ++

Notepad++ is a free and open-source code editor that is used for Windows. Notepad++ supports several programming languages. Notepad++ is written in C++ and uses pure Win32 API and STL which ensures a higher execution speed and smaller program size. The use of Notepad++ is a very good source because Notepad++ features syntax highlighting and folding, auto-complete, multi-document management, and a customizable GUI. The fact that it comes with syntax highlighting for many languages including PHP, JavaScript, HTML, and CSS it also comes with a built-in FTP plugin that allows you to connect to your server and edit files directly without leaving the editor. Notepad++ allows users to work with multiple files in a single window but it also comes with its downsides of not including code debugging and the inability to install advanced packages is something to note when choosing an editor.

4.5.2.4 Repl.it

Repl.it is a free IDE that allows users to write their own program and code within a multitude of different languages. One of the best features that Repl has is that its an online IDE that allows you to work from anywhere. It is completely web browser-based which is different from other IDE software that is normally downloaded to a computer. Another amazing feature that Repl.it has to offer is the ability for developers to work in a collaborative method if the code is shared all at once in real time. This feature is similar to that of a google doc or slides that allows multiple people in a team to work on the same thing at the same time allowing for everyone to understand the changes that occur. So this is also good because repl.it is not tied down by an operating system rather any device can use this IDE. Repl also uses addons like other IDEs in which they are easy to download by just writing their names. Since repl is web based it offers an auto save cloud feature when you create and account. This allows for a feature of downloading all of the files you need if you're offline. Repl also has a debugging feature and also has a console and uses linux so the normal use of IDEs is not lost.

4.5.2.5 Visual Studio Code

Visual Studio Code is a source-code editor made by Microsoft, that allows users to write, edit, and debug the code, all in one place. Within this code editor debugging, syntax highlighting, intelligent code completion, snippets, and code factoring is made simple along with the source code management and version control allowing you to handle small to large projects efficiently with embedded Git. It aims to provide just the tools a developer needs for a quick code-build-debug cycle and leaves more complex workflows to fuller featured IDEs, such as Microsoft's Visual Studio IDE. With Visual Studio Code, you are able to program in a variety of different programming languages such as Java, JavaScript, Python, C, C++, and so much more with plug-ins within VS Code. The ability to have these features within the software is what separates VS Code from other applications.

The embedded Git is very helpful for our project because it allows our team to work on the coding of the CMOS sensor together and allows for the remote work done to be directly placed into GitHub while being able to push and pull source code from different platforms allow for easy editing use for everybody involved.

4.5.2.6 Coding Editing Software Comparison

In this section we will use the tables below to compare all the exclusive and distinct features and specs between the different Coding Editing Software discussed in the sections above that could be put into functional use for the Lensless Digital Holographic Microscope project. The comparison tables will help us narrow down and decide on which software will prove to be the most beneficial for editing the code for our project. Like the coding language comparisons, the cost comparison was taken out of the tables since the only editor that was not free was Sublime Text. Sublime Text is able to be downloaded and used for free, but a license is required for continued use which is not effective for our project. After analyzing the comparison tables, it is apparent that the best code editor that we would use to write and debug our code is Visual Studio Code. The ability to have Git connected with the editor makes code management as well as version management crucial for our project's life cycle.

Table 4.5.2.6-1: Coding Editing Software Comparison

Features	Eclipse	Sublime Text	Notepad/Notepad ++
Programming Languages	Supports C, C++, Perl, Python, Ruby, PHP, Java and others	Supports many languages	80 Programming Languages
Operating Systems	Windows, Linux, macOS, or Unix	Windows, Linux, Mac OS	Windows, Linux, UNIX, Mac OS (By using a third-party tool)
Debugger	Included	Package Available	Available through plug ins
Refactoring	Included (available as a plugin	Package Available	XML tools
Code Completion	available as a plugin	Included	Included
Built-in Unit Testing	Included	Package Available	can be integrated via frameworks
Best Features	Runs on all platforms. Provides easy access to remote targets. Provides easy access to debug tools. Lots of documentation. Integrated git support is excellent. Once you get used to the tool, and you become an expert, the world opens up.	ease of use, excellent UI and UX, Supports Plugins and integrations, Offers command palette for easy accessibility to snippets, codes readability, search code in user customizable directories, language support, Package control, Fast and efficient	Opens large text files gracefully, Auto saving feature, Facilitation of code editing, easy to use function, Multi Window option, Plugin availability, customizable user interface, code syntax is highlighted, Flexibility
Open Source	Yes	No	Yes

Table 4.5.2.6-2: Coding Editing Software Comparison

Features	Repl.it	Visual Studio Code
Programming Languages	50+ programming languages, including Python, C, C#, Node.js, Basic, Machine Assembly and more	Supports many languages
Operating Systems	Online IDE, Editor, Compiler, Interpreter Built using web technologies so can be used on every web browser	Windows, Linux, Mac OS
Debugging	Included	Included
Refactoring	Included	Included (Typescript)
Code Completion	Included (known as autocomplete)	Yes by installing a language extension
Built-in Unit Testing	Included	Yes by extensions
Best Features	Drag and Drop, Syntax Highlighting, Source code editor,Version Control, Real-time Collaboration, Portability, large catalog of languages, cloud tool	Auto-completion Debugging with breakpoints, Command Line, Command Palette, Git Integration, Change language mode, Customization, Zen Mode, Split view, Status Bar, Keyboard shortcuts
Open Source	encouraging open-source development	Built on open source

4.5.3 OS Distribution software

Selecting the correct Operating System was an integral part of our project life cycle. The Operating system was essential to properly compiling the written code to run and function to create a successfully working fully digital system for Lensless Digital Holographic Microscope. In the end we narrowed it down to two choices: Mac OS and Windows. Windows was used because the applications that are available for Windows tend to have exceptional features when compared to other platforms. Also Windows was a common system that the team has so in order to keep everything similar for all the team members Windows was used for the development process. Windows runs everything, meaning that Windows OS was more flexible and runs everything from the latest games and software to the old software that used to be common in corporations all over the world. Almost every version of software will run on Windows and you cannot say the same about the Mac OS.

We chose Mac for the documenting and the programming since many of us have their products, we decided that both would be used for their advantages and used both in order to help one cover the others disadvantages.

4.5.4 Other Software Tools used

In this section we discuss the other tools that were used within this project. Within this section we used software tools for version management, prototyping and 3D modeling, schematic building, circuit building and many other tools that allowed for the project cycle to be productive and effective. Each of the sections highlights the tools advantages and the reasons why the software was used for the project.

4.5.4.1 GitHub

GitHub was a powerful collaboration tool and development platform for code review, management, and version control. With GitHub, users can build applications and software, manage projects, host and review code, etc. GitHub's project management tools help its users to stay aligned, coordinate easily, and get their tasks done accordingly. Github's cloud-based service helps developers keep track of and control the changes they see within their code and projects. This is where version control comes in and why GitHub is the best product for this. As companies and developers move along their project's life cycle, version control becomes essential to see the changes that their code goes through. Version control lets developers safely work through branching and merging. With this, developers continue on in the software development process while working in parallel without any issues as GitHub separates out "in-progress work" from tested and stable code within the repository. The developer can then safely make changes to that part of the code without affecting the rest of the project. Then, once the developer gets his or her part of the code working properly, he or she can merge that code back into the main source code to make it official. All these changes are then tracked and can be reverted if need be. Git is a specific open-source version control system.

4.5.4.2 SolidWorks

SOLIDWORKS is among the most common computer-aided design (CAD) and computer-aided engineering (CAE) on the market. It uses a 2D and 3D parametric modeler to allow engineers to develop mechatronics systems and streamline the design process of engineering and architecture while developing new, innovative products from beginning to end.

At the initial stage, the software is used for planning, visual ideation, modeling, feasibility assessment, prototyping, and project management. The software is then used for the design and building of mechanical, electrical, and software elements while tackling the engineering analysis of it all through processes such as Electromagnetic, thermal, and fluid analysis, and more. Finally, the software can be used for management, including device management, analytics, data automation, visual analytics, and cloud services. SOLIDWORKS allows engineers to take full advantage of the fully integrated analysis and simulation tools while gathering important information with visual reporting and analytics. SolidWorks 3D rendering allows our group to take full advantage of realistic details, thereby making our processes and products more efficient.

4.5.4.3 EAGLE

In short, EAGLE was free to use for students' programs that are useful for modeling electrical schematics and their breadboard counterparts. It is software that we can import into the models of the parts that we intend to buy or change an already existing model slightly to better resemble a part in which we are interested. Several models are free to use from preexisting sites, either Ultra librarian, Digi key, or the manufacturer. Small addition, for some reason none of the previous sites had one of the eagle files however a site called the component search engine which had a collection of not just the eagle library model but multiple versions for different softwares, definitely a new resource moving forward. Any or all of these will either have the models for the parts that we will be purchasing, and it has been easy to download and use the models. From there it is seamless to wire the parts in a simulated environment very close to the real-world version that we plan to do. This will be especially helpful when planning which pin to which is best to use. The only downside is that the program will not detect if there is an error, so we will also be relying on other simulation software to make up for that shortcoming. Additionally, we did not make real use of the port to beard board feature since in the current design of the Lensless Digital Holographic Microscope CMOS sensor and the LEDs will be on the exact opposite side of the device so it would not really help to see what they should look like wired side by side. Overall, a great resource that we make presenting the process that we intend to wire the Lensless Digital Holographic Microscope to look clean and legible.

4.5.4.4 Multisim Live

This is the online program that will be featured if we want to show a simulation of a certain complex point in the wiring of the Lensless Digital Holographic Microscope. This resource should be used sparingly since it is only useful for modeling the basics, but it can be helpful to show examples of our circuit design in this program so that the Lensless Digital Holographic Microscope is working on a fundamental level, and we are not just wiring things together and hoping for the best.

4.5.4.5 Mobile Application Development Software

As one of our stretch features was the creation of a mobile application for further advanced implementation of our microscope to the everyday person. The selection of the correct mobile application development software was critical to properly create a successfully working digital system fully functional for all to use via mobile device which in this case would have been an iPhone for our project. The reason being was that with the ability to have access to the internet at your fingertips through your smartphones and tablets creating and developing a mobile app has the unique ability to access many potential users of our product and would have allowed the users to do the same things we were able to do with computers. The languages that were considered to be used were Swift, Python and C as the team has background in each of these languages allowing for a collaborative effort in the creation of the app to be smooth and seamless.

In contrast to the creation of a web application that desktop computers use and we would use for the initial testing phases of our project, mobile applications allow for access to almost every type of online platform. App development would have assisted in reaching marketplaces via Blackberry, Google Play, Apple App Store, and other internet marketplaces also through social media sites which meant that we would look at our marketing requirements and narrow down the best approach for the market in which we want our product to go to. The product would have had the ability to reach a lot more customers' hands in contrast if it was used only with a desktop application. Mobile applications are usually 1.5 times faster than web applications, and they have an added advantage where developers can code various features utilizing the embedded hardware in the native device in use such as a stretch feature we wanted to do with our product such as it being bluetooth compatibility. Having a mobile application that incorporates a user-friendly interface allows for the customers to not be intimidated with the product allowing more comfortability with using the product for its intended use. The creation of a mobile application also allows for our product to stand out from the competition with our forward thinking approach. In future sections this possibility may be explored with sections displaying the mobile application development software tool that can be utilized to design, develop, and prepare the project. Sadly due to time constraints and limited resources the mobile application had to be abandoned as we focused on the completion of the project due to the project only being funded by the group was difficult to extend the project as initially planned meaning our planning was more precise so the allocation of funds was used effectively.

4.6 Photonics Equations, Methods, and Algorithms

This section discusses the background information with a deeper dive into the relevant topics and equations used within the project and the world of photonics.

For Gabor in-line holography, the approximation equation for semi-transparent objects is:

(1) $$t(x_0,y_0) = 1 + \Delta t(x_0,y_0)$$

where $\Delta t << 1$

The light will propagate coherently when the object is illuminated by plane wave A. In this scenario, z_1 is the distance between the light source and the object sample plane and z_2 is the distance between the object sample plane and the sensor plane (CMOS).

$$R[z_2]\{A \cdot t(x_0,y_0)\} = R[z_2]\{A\} + R[z_2]\{A \cdot \Delta t(x_0,y_0)\}$$

(2) $$R[z_2]\{A \cdot t(x_0,y_0)\} = A' + a(x,y)$$

A hologram is then formed at the sensor by the interference of the un-scattered reference beam (A′) and the scattered sample beam, a(x,y).

$$I(x,y) = |A' + a(x,y)|^2$$

(3) $$I(x,y) = |A'|^2 + A'^* \cdot a(x,y) + A' \cdot a^*(x,y) + |a(x,y)|^2$$

In the following equation, ★ represents a spatial convolution symbol, I^{coh} represents a hologram with perfect spatial coherence, and I^{meas} represents the measured hologram.

(4) $$I^{meas} = I^{coh} \star T[(-z_1/z_2)x, (-z_1/z_2)y]$$

The coherence length of a single point illumination source (ΔL_c) is used to determine the temporal coherence of the illumination, which is needed for the spatial resolution of an on-chip holographic microscope.

(5) $$\Delta L_c \approx \{[(2 \cdot \ln(2) /\pi\} \cdot [(\lambda^2)/(n \cdot \Delta\lambda)]$$

where λ is the illumination wavelength and n is the refractive index

For the spatial frequency component of the object to be imaged at the sensor plane, the optical path length difference (ΔOPL) of the high frequency scattered wave must **not** exceed the temporal coherence length (ΔL_c). The maximum angle for any of the plane wave components is found through:

$$\cos \Theta_{max} = (z_2 + \Delta L_c)$$

$$\Theta_{max} = \cos^{-1}(z_2 + \Delta L_c) \tag{6}$$

Cost function to solve an optimization problem:

$$x^* = \arg \min \Sigma \|W_i \cdot (x-y)\|_p^q + \alpha \cdot \gamma(x) \tag{7}$$

The next step is the reconstruction of the digital hologram. Back-propagation reconstructs the amplitude and phase and converts the object's holograms into four terms.
Phase retrieval algorithms will be described further along in the paper.

$$R[-z_2]\{A^* \cdot I(x,y)\} = |A|^2 \cdot [1 + \Delta t(x,y) + R[-2z_2]\{\Delta t^*(x,y)\}] + R[-z_2]\{|a(x,y)|^2\} \tag{8}$$

where $|A|^2$ is a DC-Background Term, $A|^2 \cdot \Delta t(x,y)$ is the reconstructed object function, $|A|^2 \cdot R[-2z_2]\{\Delta t^*(x,y)\}$ is the twin image artifact, and $R[-z_2]\{|a(x,y)|^2$ is the back-propagated self-interference term.

Allowing the first and last term to be lopped off. This is because the first term is simply the background source which we already have. And so it is easy to remove. And the last term is too small to record at this point.

$$x^* = \arg \min \|Ax-y\|_p^q + \alpha \cdot \text{card}(Cx) \tag{9}$$

The below is a convex optimization problem to solve for x*.

$$x^* = \arg \min \|Ax-y\|_p^q + \alpha \cdot \|Cx\|_1 \tag{10}$$

4.6.1 Holography and Digital In-Line Holography

In-Line Holography is a microscopic principle that was founded by David Garbor that records the entire field of information such as the amplitude and the phase other than just the intensity. Gabor developed this method in the 1940s as a way to improve the resolving power of electron microscopes. One thing to understand was that this method was built upon the idea of Holography. Holography is done within two steps: writing and reading. For the writing, we are getting a hologram. For the reading, the hologram essentially means illuminating the hologram as if it is a new object such as seen with the Figure 4.6.1-2. We are then reconstructing the hologram through two processes: the numerical one with the Kirchoffs-Helmotz Transformation equation and the physical one with the inversion of light paths. The field is then scattered from the hologram with the product being created by the illuminating plane wave and the transmission function. So from the recorded holograms, we are able to image the original object. Both its amplitude and phase images can be reconstructed digitally.

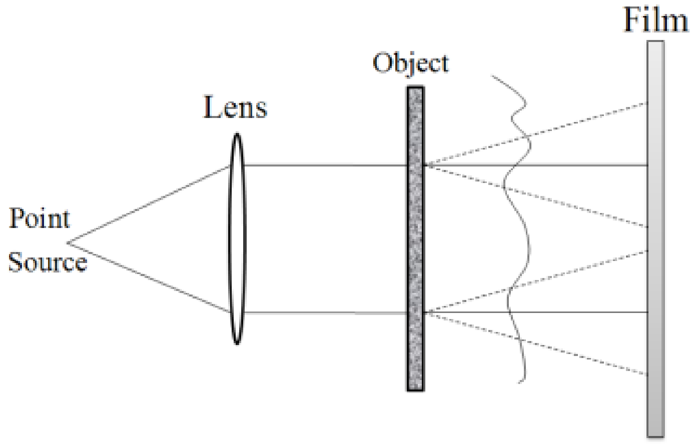


Figure 4.6.1-1. In-line optical setup for writing Fresnel holograms (Holography)

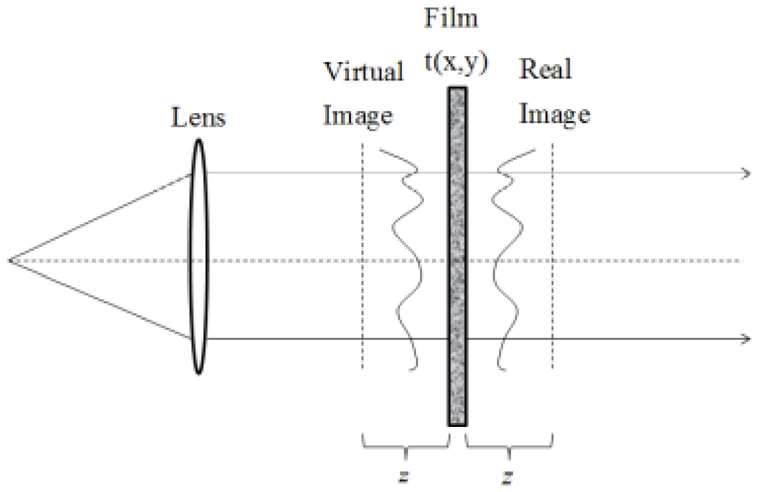


Figure 4.6.1-2. Reading an in-line hologram (Holography)

With Digital In-line holographic microscopy it was very different from other microscopy methods because with the digital inline method there was no recording of the projected image of the object. Instead, the light wavefront information originating from the object would be digitally recorded as a hologram, from which a computer calculates the object image by using a numerical reconstruction algorithm. This calculation is done through programs such as MATLAB or already created products. The image-forming lens in traditional microscopy is thus replaced by a computer algorithm. Other closely related microscopy methods to digital holographic microscopy are interferometric microscopy, optical coherence tomography, and diffraction phase microscopy. Common to all methods was the use of a reference wavefront to obtain amplitude (intensity) and phase information. The information is recorded on a digital image sensor or by a photodetector from which an image of the object is created (reconstructed) by a computer. In traditional microscopy, which did not use a reference wavefront, only intensity information was recorded and essential information about the object was lost.

4.6.2 Propagation and Back-Propagation

Propagation has two types, forward and backward, and deals with neural networks. Forward propagation is the way to move from the Input layer (left) to the Output layer (right) in the neural networks. When we move in the opposite direction from right to left meaning from backward the Output to the Input layer we are doing backpropagation. So we use backpropagation in order to traverse the network in reverse order from the forward propagation by the chain rule used in calculus. With this algorithm, we store any intermediate variables (partial derivatives) required while calculating the gradient with respect to some parameters. So first in order to backpropagate we need to do forward propagation, followed by Backward Propagation where we will propagate backward. This way we will try to reduce the error by changing the values of weights and biases. Following this, we will then put all the values together again calculating the updated weight value. This can be done too in order to update and calculate the other weight values. We must note that if the error is a minimum then we will stop. If not, we will continue to repeat the process until the error becomes minimum.

Backpropagation sequentially calculates and stores the gradients of intermediate variables and parameters within the neural network in reversed order. In our case dealing with fibers and optical components, digital backpropagation is an important method since it is a nonlinearity compensation method. We are able to use this technique in order to compensate for all fiber impairments in the optical transmission systems we may have.

4.6.2.1 Angular Spectrum Method

In this angular spectrum method, a complex wave is first transformed into spatial frequency (Fourier)domain, multiplied by a propagation kernel which was a function of the propagation distance (z2) and was finally transformed back into the spatial domain, using e.g., Fast Fourier Transforms (FFTs). This back-propagation step converted the object's hologram into four different terms that would then be reduced to the image and its twin artifact. And as mentioned before there are other ways to reduce twin image artifacts.

4.6.3 Pixel Super-Resolution

There are multiple types of pixel super resolution algorithms. This is when an image has its resolution artificially increased using computer programs. The easiest one implemented was the shift and add pixel super resolution technique where multiple images taken from slightly different locations are layered over one another with the subject lined up perfectly. This allowed the image processing software to see distinct edges and detail that each individual image provides. This was also known as sub-pixel resolution because some of the detail that spreads over multiple pixels can be brought back out and seen on the final image. Current leading-edge technology uses artificial intelligence and deep learning algorithms to deduce what the image should look like and introduce detail that would not have been there before. This was possible because any blur that occurs on the image would look similar to a human eye but in reality the blurs are unique and contain information in them themselves.

4.6.4 Spatial Convolution

Spatial convolution allowed us to digitally alter an image using convolution mathematics and a kernel image. By analyzing each individual pixel on the image taken from the camera, and running it through the convolution with the kernel. We could then highlight certain aspects of the image that we would like to emphasize. We were also able to undo certain spreads of the image if we knew how the camera will spread light. Knowing what the function going in was, it was very easy to calculate the function out and get the object representation, which will be in higher detail than the image. For example, if every "pixel-size" of light coming in, spreads and hits 92% in the pixel, but 1% in each pixel that directly borders it, we can go pixel by pixel and pull 92% from the original pixel and 1% from each pixel that borders it. This gave us the original light coming off of the object and thus created t a higher resolution than the original image. This was just one way we used spatial convolution. There were more features that were distinguished using spatial convolution. Add up enough of these feature detection convolutions and we were able to get an image of a much higher quality than the original one could have ever been.

4.6.5 Temporal and Spatial coherence

Temporal coherence was how in-tune the wavelengths were, or in other words, how monochromatic the light was. We need monochromatic light because the interference pattern we intend on imaging relied on the same wavelength and its interference pattern. Spatial coherence was how in tune the wavelengths are with each other. We needed a good amount of spatial coherence to ensure the information was properly recorded. But we can not have perfect spatial coherence because it introduced unwanted artifacts like optical blur and optical shadowing. This was why the LED was the perfect choice for us, it has high temporal coherence but not perfect spatial coherence.

4.6.6 Phase Retrieval

The information we were able to record was contained within the phase pattern and interference projected on the sensor. But all that was recorded was intensity at a certain pixel. So what we were left with was an intensity pattern and no phase pattern. Luckily, we were able to use a variety of phase retrieval algorithms. One such one was the Gerchberg–Saxton algorithm which uses two different measurements of intensity to trace back to the phase information. This was why we used multiple point sources each fired off sequentially. This allowed us to trace back the phase information using only intensity patterns measured at different locations

4.6.7 Pixel Pitch

Pixel pitch is the aspect of a CMOS sensor most closely related to the pixel size, which was naturally the most important contributor to the quality of the resulting image. The exact definition was best put by a retail site by the name of insane impact, where essentially it was the distance between the center of any two given pixels on the CMOS sensor, therefore the lowest the pixel size to pixel pitch ratio will ever be is one (4). A pixel pitch above one could result in a short gap in the resulting picture that when trying to visualize a small object, could mean that valuable information was lost. In the market research for CMOS sensors, we have found that any CMOS sensor worth a look either had a pixel pitch that equaled the pixel size or had it unlisted, therefore it would appear that a pixel size to pixel pitch ratio has an industry standard of one, additionally according to the work Wu and Ozcan if there would be an issue with any given pixel pitch using pixel super-resolution can be used as a workaround (3).

4.6.8 Gerchberg Saxton Algorithm

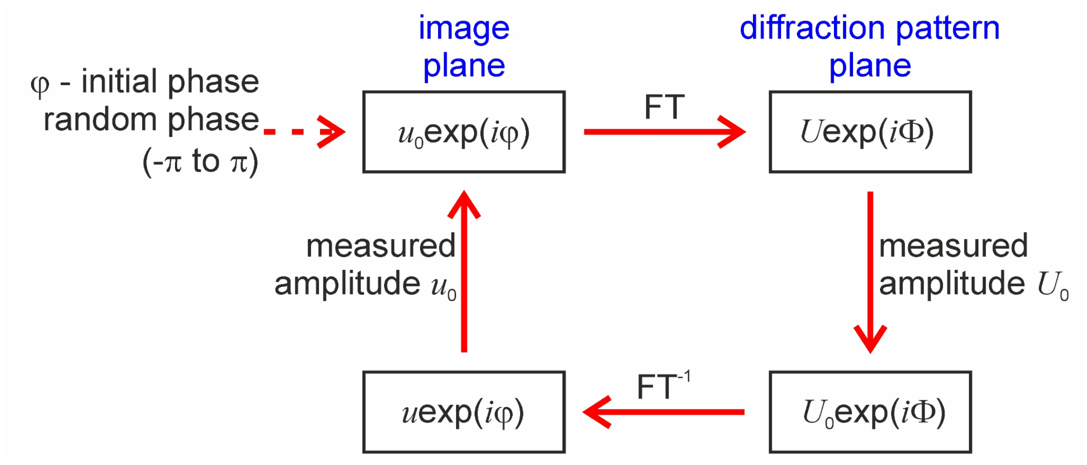


Figure 4.6.8-1. Example of how the GS Algorithm works

Gerchberg Saxton Algorithm is an iterative phase retrieval algorithm where we have only two intensity only images like the images that are captured by a camera. These two images are captured on two different planes, the image plane and the Fourier plane where the planes are separated by a Fourier transform. In the image above this can be seen. The image plane can be understood as the intensity of the near field and the Fourier plane being the corresponding intensity of its diffraction pattern. The goal of this algorithm was to extract and/or guess the phase at each point/pixel in the images used and collected in the near (image plane) and far (Fourier plane) fields. The Gerchberg Saxton Algorithm (GS Algorithm) was straightforward and it only has a few steps and with this, it was able to converge to a solution very quickly despite it having a very large solution space. The images in the rear and the far field consisted of many pixels that could any phase value which helped highlight the point that there are many unknowns at the have start of using the algorithm. Another thing to note was that the Gerchberg Saxton Algorithm did not give the global optimum solution but when used in the case of simple scenarios, the algorithm will present solutions that are pretty close. One of the most important things about this algorithm was that if we are asked to extract the phase of only one intensity only image it would be far too difficult but what the GS Algorithm has is the Fourier transform the relationship between the image and Fourier plane. One thing to note about this relationship was that each point within the Fourier Plane is the interference between every point in the image plane. What does this mean? It means that every point in the Fourier Plane gives a different interference between all the points in the image plane. In other words, every point in the Fourier Plane is the superposition of all the points in the Image Plane.

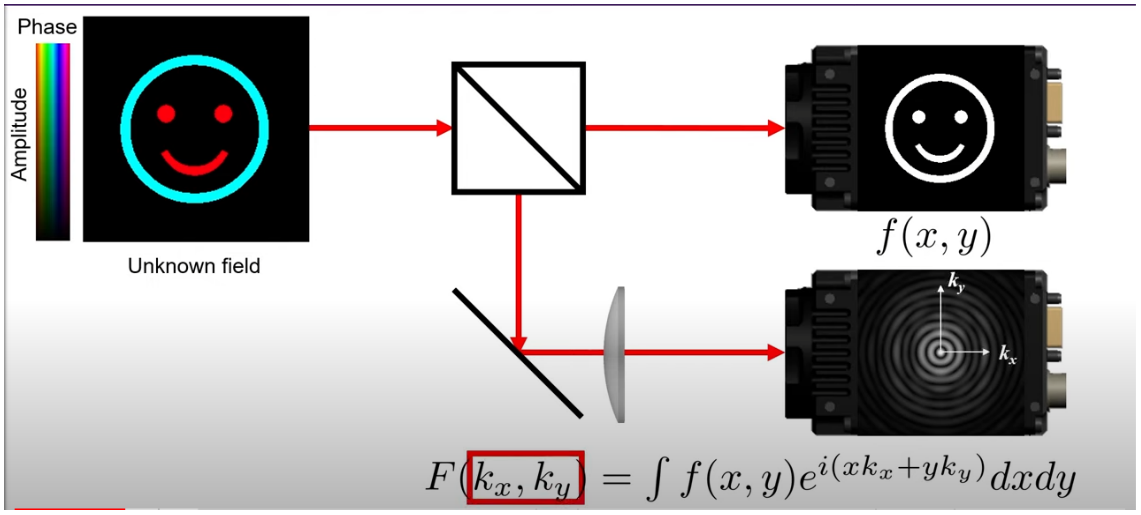


Figure 4.6.8-3. Visual representation of superposition and interference (joelacarpenter)

Image: f(x,y)
Fourier: $F(k_x, k_y) = \int f(x,y)\, e^{i\,(x*k(\text{sub } x)\ +\ y*k(\text{sub } y))}dxdy$

This interference between the two planes helps reveal the phase information. Phase information is extracted by interfering beams or other parts of beams together to then discern the phase shift between them based on the intensity created by the beams when they're interfered together. The following scenarios are when the Gerchberg Saxton Algorithm is used:

- Phase Retrieval
- Calculating a phase mask to generate a desired intensity in the Fourier plane.
- Calculating a phase mask to generate the desired field (amplitude and phase) in the Fourier plane

It was important to note that the Gerchberg Saxton Algorithm has many different variations and it was originally used for phase retrieval and x-ray diffraction rather than photonics and optics. The issue arose with x-rays that made some interference techniques that would be used in optics to retrieve phase information too difficult or impossible. X-rays would only allow for getting an intensity of an image of the image (near) and the Fourier (far) planes. It is important to note that the brightness and darkness in the algorithms are the field strength (square root intensity).

4.6.8.1 Gerchberg Saxton Algorithm: Phase Retrieval

So the way that the phase retrieval works with the Gerchberg Saxton Algorithm was very interesting. We started off with an unknown field meaning that at the beginning the phase profile was unknown. To do this we took a look at that field on a camera and we are able to see the intensity of the image which in this case was the smiley face that has no phase information present only the brightness for each pixel that the camera records/captures being shown. This was seen within the captured image plane and can be seen in the figure below. After this we then take a look at the beam/image in the Fourier plane, far field.

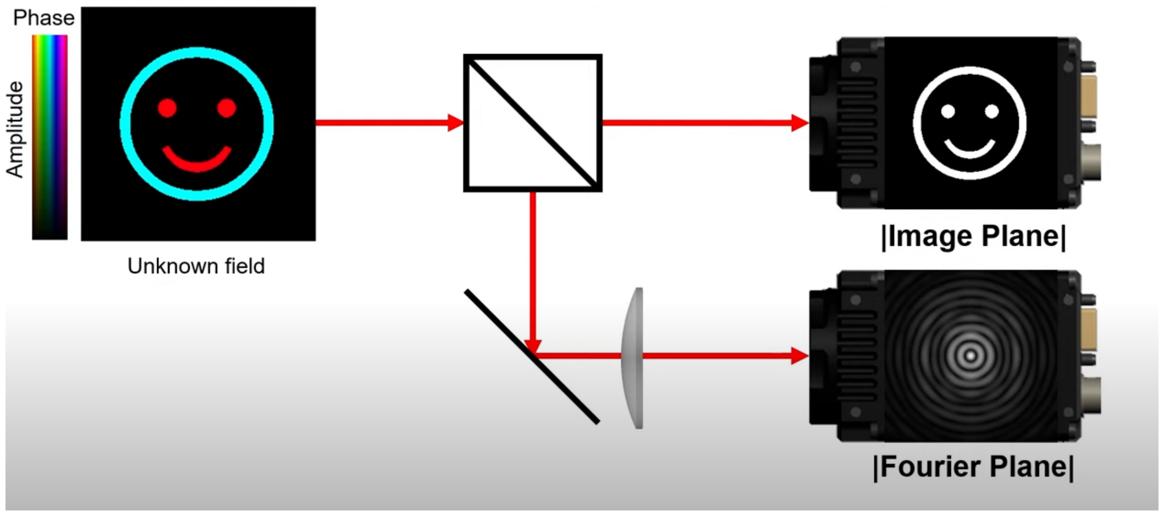


Figure 4.6.8-3. Visual representation of Phase Retrieval with GS Algorithm (joelacarpenter)

Two methods can be used to do this, the first would be to move the camera back until you reach the far field. The method which is shown in the diagram above is to split the beam going into the first camera with a beam splitter to then look at it through another camera in the Fourier plane with a camera at the focal plane of the lens. The Fourier plane is able to show us the intensity of the diffraction pattern of the image, the intensity recorded at each pixel is the information gathered at this point. This is important because every pixel and the Fourier plane is the interference between every point in the image plane, and every pixel in the second camera is the interference between all the pixels on the first camera. One thing to know is that since there are so many different phase values and camera 1 and camera 2 the values are not consistent with the intensity of camera 1 and camera 2 which would mean that the phase can’t be anything since it has to be consistent with the observed intensities. The reasoning behind this is that at random phase values there would not be any consistency with what would be measured because when you change the phase in one plane you can change the corresponding intensity formed by the interference in the next plane. There is an exception where the two beams have the same image plane and Fourier plane intensity even though their phase is different with Gaussian modes being an example of this. The two planes do not need to be related by a Fourier transformation, it can be any other unitary transformation, that is a lossless transformation cause you do not want to lose information. The interference just has to be enough with the Fourier transform is the extreme case.

For phase retrieval, we start by taking a starting guess of the phase in the image plane. This means that we will not be Fourier transforming back and forth between the image plane and the Fourier plane, this is because with each iterative step the amplitude is reinforced from the amplitude seen on the cameras so we discover the amplitude for certain. The phase during this does whatever it likes in order to accommodate for the known amplitude. So the amplitude is fixed and the phase values are the three variables that most satisfy the constraint which is the amplitude. So we must start off with the measured value which is the amplitude that is seen on the camera with the phase being unknown so we must guess or guess that it's flat. This is all done on the image plane. Now we move on to the Fourier plane. When we first view we see we get some distribution of amplitude and phase so in order to get the same values as the image plane we replaced the viewed amplitude in the Fourier plane with the measured amplitude from the image plane. So we are going to replace the amplitude in the algorithm based on the initial guess with the amplitude we actually observe because the measured amplitude is true. Just like the image plane, the phase can do whatever it likes so it can be kept as is since it is the parameter that is being found. Amplitude is enforced as a constraint and phase is a free variable within the algorithm. Then the inverse Fourier transform (FT $^{-1}$) is used back to the image plane and we follow the same process again, replacing the amplitude with the observed value and letting the phase as is in order to transform into the Fourier plane (FT) again continuing this process is iteratively constantly enforcing amplitude to conform with what was observed on the cameras and letting phase vary in order to accommodate.

So each generation nudges us closer towards a solution that satisfies the constraint. This part of the algorithm should be thought about as an oscillator. This is because it is like you have a spatial filter and with the iterative loop it allows for every iteration to have a selection based on part of the field that is being observed which aligns with the target want and it sent back through, continuing to replace the measure the amplitude with the amplitude we know to be screw and nudge it in the right direction of a better solution. So Gerchberg Saxton uses phase retrieval with two planes of known intensity and unknown face.

4.6.8.2 Gerchberg Saxton Algorithm: Calculating a phase mask to generate a desired intensity in the Fourier plane

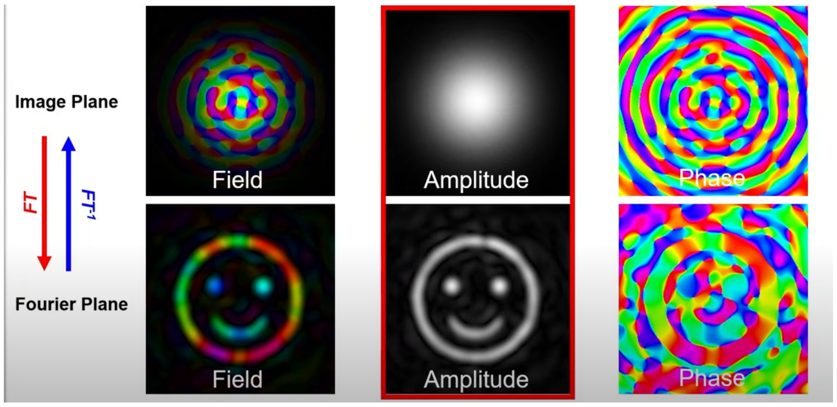


Figure 4.6.8.2 -1. Visual representation of Phase mask with GS Algorithm (joelacarpenter)

Gerchberg Saxton's algorithm was used when the calculation of an optical mask was needed for something like a hologram that will give a desired intensity pattern in the Fourier plane. Use cases are the same as the phase retrieval method discussed above, experimentally it is different. In this case, you have two known amplitudes in the image and Fourier planes respectively, and two corresponding unknown phases. The problem is the same as before, the algorithm is ran exactly the same. An example of this is Gaussian illumination beams which want to generate an image intensity in the Fourier plane, running the algorithm eventually results in the image intensity in the Fourier plane. The known Gaussian amplitude illumination is enforced on the image plane and forces the goal amplitude in the Fourier plane while the phases in both planes are free to take on whatever form as long as it's consistent with the enforced amplitudes.

4.6.8.3 Gerchberg Saxton Algorithm: Calculating a phase mask to generate the desired field (amplitude and phase) in the Fourier plane

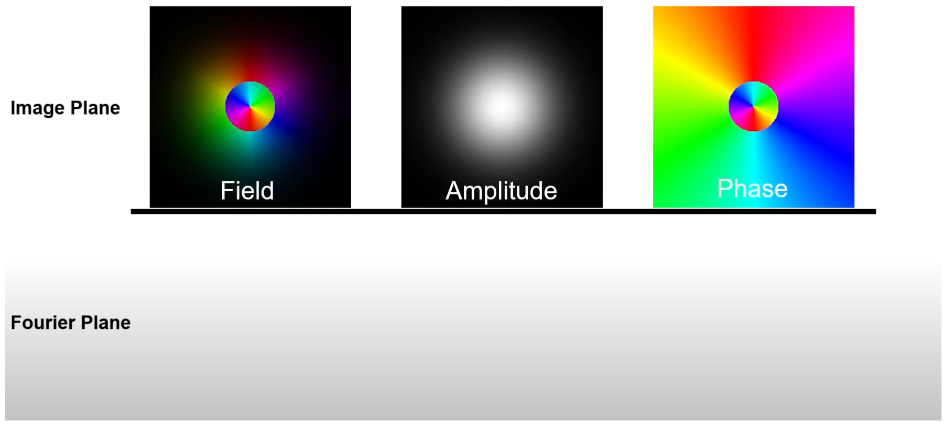


Figure 4.6.8.3-1. Visual representation of calculating amplitude and phase with GS Algorithm - image plane (joelacarpenter)

The difference in this scenario compared to the other scenarios above is the importance emphasized in the amplitude and phase with a fourier plane. In this scenario the phase can't be of any value anymore. So now instead of the phase being a free parameter it is now being enforced and controlled to conform to the requirements of the phase mask for the goal field. This means that the goal has now changed from being able to measure and calculate a complex field and not just an intensity that the previous scenarios demonstrated as amplitude and phase are now being controlled. The application of the algorithm is similar to the previous scenarios as we transform back and forth between the planes, but the parts you keep constant and reinforce and let change are different. So in the image plane we keep constant the Gaussian illuminating beam on the phase mask, the beam coming from our laser source. The phase mask that this Gaussian beam illuminates is a free parameter which means that it can either be any value or flat, but it can be any other beam type with non-uniform phase; it would just have to be extracted from the beam at the end with the calculated mask. In the Fourier plane now instead of setting the amplitude as the goal and the phase as the free parameter as we had done in previous sections, it is now set such that a field inside a certain region of the Fourier plan is now the goal which is continuously being reinforced through each iteration. The field that is outside of this region can do whatever it likes. So like before the enforced goal is the region inside of the Fourier plane with the free parameter being the area outside of that region. So the scenario splits it into an area of interest in contrast to the other scenarios that had it as the amplitude and phase. Just as in the other scenarios with the algorithm we start off with a guess of the phase mask which we then transform into the Fourier plane.

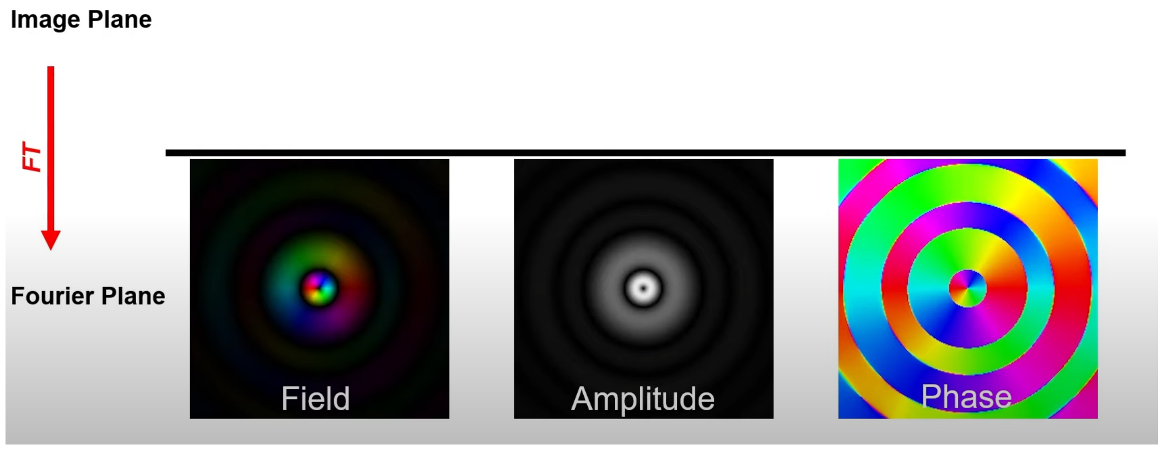


Figure 4.6.8.3-1. Visual representation of calculating amplitude and phase with GS Algorithm - Fourier plane (joelacarpenter)

Once we transform it into the Fourier plane the algorithm then replaces the central region that is observed with the goal field that is desired. We leave the outer region of the goal field untouched making that the free parameter. This scenario the weight in which the goal field takes on is very important. This is because if the goal is enforced 100% with the background power for the outer region being 0% means that there is a total emphasis on the ball field and little emphasis on the free parameter of the outside goal field causing the algorithm to stall. This means that the results we see within the first generation will always be seen. If the other extreme is used where there is more emphasis on the background and very little emphasis in the goal (goal is 1%) then the algorithm becomes unnecessarily inefficient as you're throwing away extra power in return for a better result. In order to achieve the goal beam quality in the region of interest we use the algorithm in order to put no more than the necessary power within the background region. Just as the previous scenarios we continue to do the multiple iterations of transforming it into the different planes through the Fourier transform and the Fourier inversion transform until it converges. Maximum power that we are able to deliver to the background region is going to be the amplitude overlap The Illuminating Bheem and the goal field in the plane of the phase mask.

4.6.9 Gerchberg Saxton Algorithm Things to Remember

Things to note:

- Light waves are characterized by their amplitude and phase
- Amplitude can be determined by measuring light intensity with light detectors

An iterative algorithm that is used to recover phase data

Input: two sets of magnitudes (sampled image and diffraction intensities)

- These intensities are related through the Fourier Transform. The diffraction function is the image function

Output: An estimate for the unknown phase

4.6.9.1 Gerchberg Saxton Algorithm Problems

- The error tends to decrease quickly, then remain stagnant for several iterations before dropping again
- The random initial phase causes the variability in Output. The same input initialized with different random phases can have very different outputs

4.6.9.2 Gerchberg Saxton Algorithm Error Convergence

The error will decrease or at worst remain constant with each iteration of the algorithm

4.6.9.3 Gerchberg Saxton Algorithm Random Initial Phases

- Although the random phases result in inconsistency and are not always necessary they are still used within the algorithm
- If both inputs are centrosymmetric, setting all phases to a constant will cause the algorithm to fail
- With non-centrosymmetric input in order for that input to succeed using the original algorithm, a constant initial phase estimate produces success in far fewer iterations

4.7 Fiber-Optic Cable

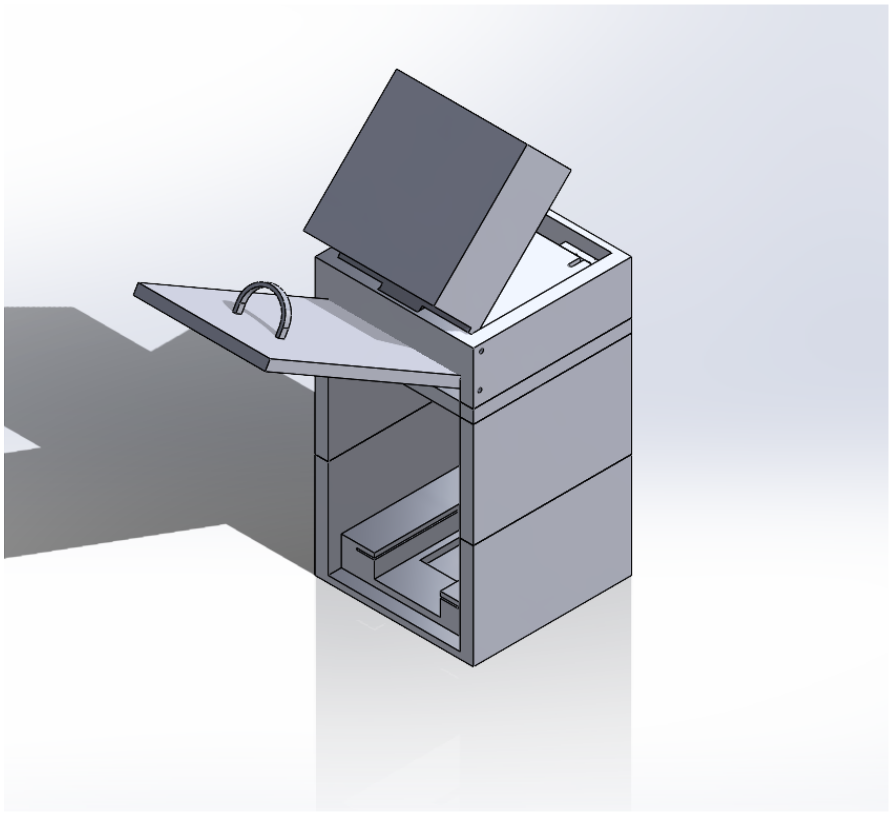
We used 15 - 25 strands of multimode fiber to be able to pump as much light as possible into the fiber. The reason we were using fiber was so that at the point of light exiting the fiber we can treat the fiber as a point source of light and properly get the hologram/phase information we are looking for.

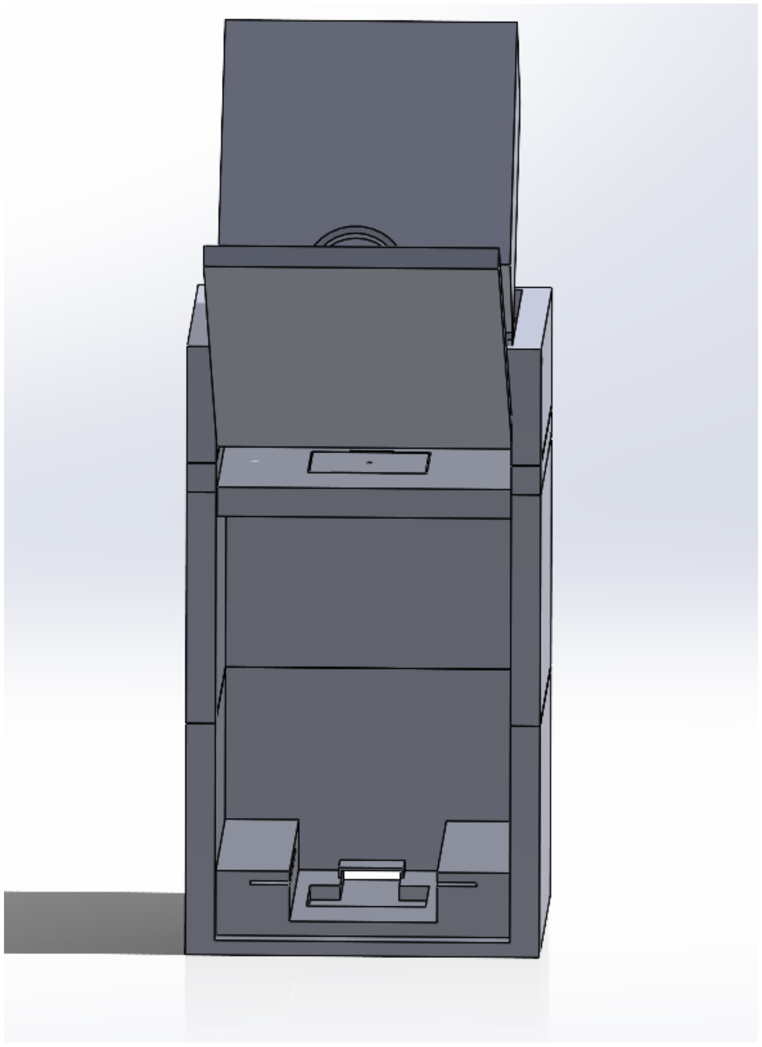
4.8 3D Printing

3D printing allows for a piece to be rapidly manufactured to exact specifications and an incredibly low cost. The 3D printers are provided to us and therefore the only cost we need to factor in is the filament cost. The beauty of the printer is not only in how fast the piece "arrives" but how quick the turnaround time is for pieces we need to re-manufacture. It allows for rapid testing and rapid adjustments to be made to find the perfect fit or make the piece necessary to turn what we already have into the perfect fit.

4.8.1 Chassis Housing Unit

We designed our model's housing unit in Solidworks and 3D printed it to ensure perfect dimensions and sturdiness to the design. Using UCF's CREOL or undergraduate senior design 3D printers, we were able to get a precise print. We did have issues with printer filament swelling and printing inaccuracies with hole sizes. Employing test prints was a large factor as well, ensuring the printer was working effectively. The housing unit we drafted is shown below. There are two arrays, high-power LED array and pinhole array, that are in their respective slots of the chassis.


4.8.1-1. Full Pinhole Chassis (Isometric View)


4.8.1-2. Full Pinhole Chassis (Front View)

4.8.2 Fiber-Optic Chucks

Fiber holders usually hold only one fiber. We tested out how this would work with two fibers per unit but the spacing between them is too varied to properly record and then use to backpropagate. Therefore, we modeled the same fiber chuck available from ThorLabs but instead of running one fiber out from the very center, We ran 5 fibers out at the same radial distance and equal angles from each other. This allows us to have a very controlled position of the five fibers that we can then call back on. The plan now is to have 3-5 of these 5-fiber chucks in a triangle or pentagon form to have a 15 - 25 fiber system. The chucks were printed using a 0.1 mm resolution because the fibers themselves have a 0.5 mm diameter. This lets them fit snugly in place and will prevent them from moving as we image our subject. Modeling our chuck after ThorLabs' chuck allows us to use our chuck in the same system as theirs. This is necessary because during the aligning process we use their rotation mounts and pitch/yaw adjustment mounts to best couple our light into the fiber.

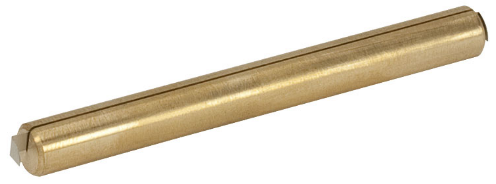


Figure 4.8.2-1. Thorlabs Side Loading Fiber Chuck

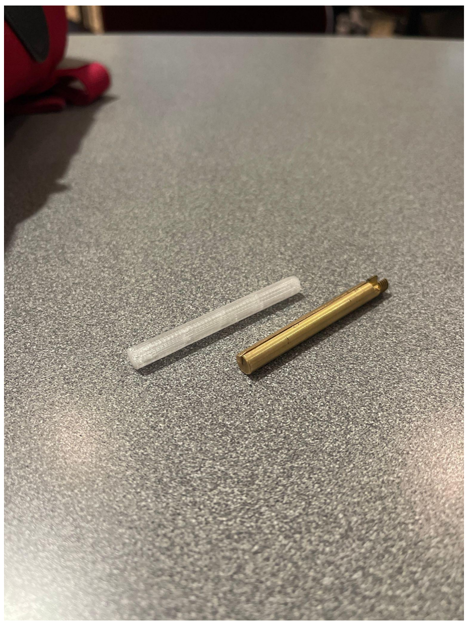
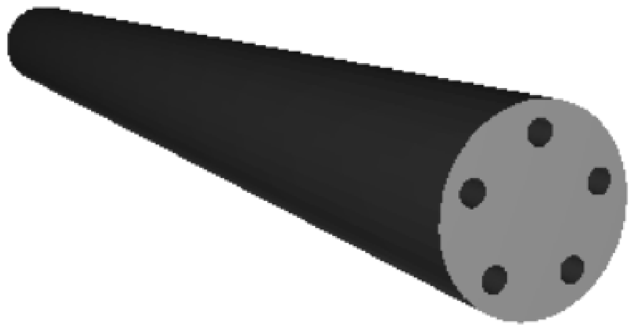
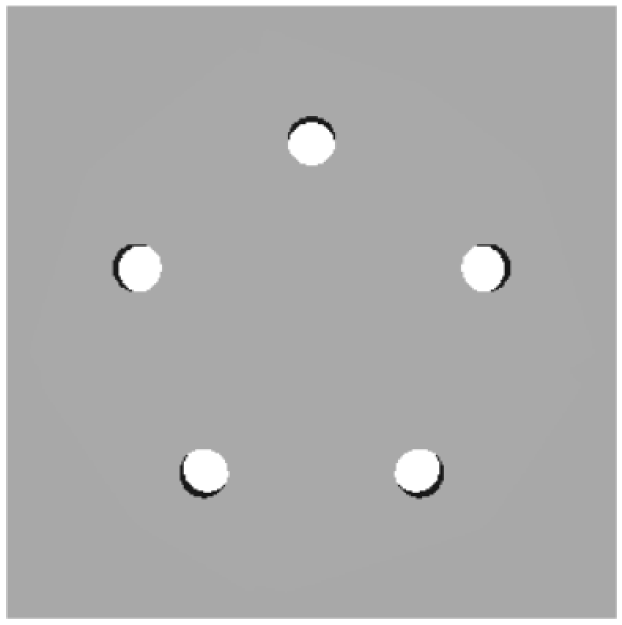


Figure 4.8.2-2. 3D Printed Side Loaded Fiber Chuck



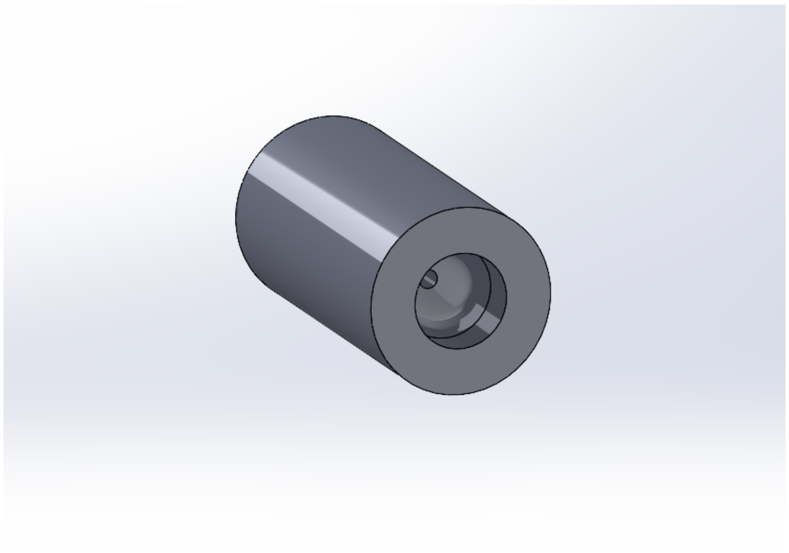
4.8.2-3. Solidworks Designed Fiber Chuck (Holds 5 Fibers)



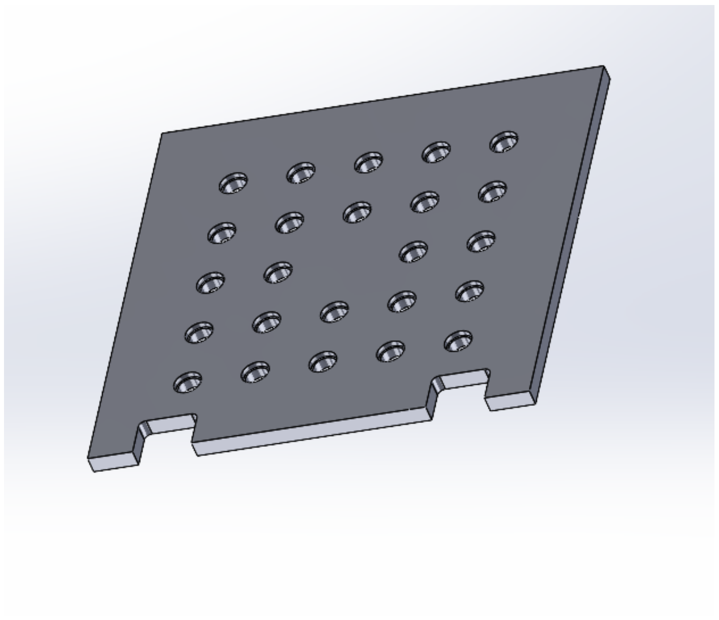
4.8.2-4. Solidworks Designed LED Array (Holds 5 LEDs).
This was an initial design used for system testing. The final LED array will be larger and contain more LED slots, as well as slightly different LED slot sizes that match the diameter of the chosen LED.

As well as these printed parts, next semester we designed and printing a fiber chuck holder that holds 5 chucks in a similar array pattern that the chuck has. The purpose of using such an array pattern is to get radiation from a wide range of angles, which better adds to our final resolution.

For our final fiber chuck design, we designed a chuck that snaps onto the LED directly, with a hole in the middle for fiber insertion and coupling. The array for the LED-Fiber design was made to house 24 LEDs and keep them tightly secured. The completed designs are shown below.



4.8.2-5. Final LED Fiber Chuck Design



4.8.2-6. Final LED-Fiber Array Design

4.8.3 Prusa i3

The Prusa i3 was provided to CREOL students by the college. The printer came with the PrusaSlicer software to properly send over gcode into the printer. The reason we chose to use this printer instead of other ones offered at UCF was because of its ease of use for beginners and extensive documentation on problem solving online. It had plenty of customizability including extrusion speed, extrusion temperature, bed temperature, infill, and resolution which allows us to tailor specifically what we need to print the small parts. Since this is a small part it means the moving of the bed won't let the inertia knock the print off the plate. And with its dimensions of 250 x 210 x 210 mm. It was clear this printer was more than sufficient to print the small parts we needed.

Figure 4.8.3-1. Example of a Prusa i3 that was used

4.8.4 Ender 3 Pro

UCF has also offered the use of other printers, the Ender 3 Pro which uses PLA, ABS, TPU that can be used to print our design through the help of ASME students as they are the ones with 3D printers.

The Ender 3 Pro offered some good advantages to its use such as low-noise, better stability, and less clogging of the extruder which was an issue for other 3D printers that can be used. The feature of having a power failure protection function was crucial when working to make the parts for our project because it would have allowed for us not only to save time if the power goes out in a circumstance but it also would have allowed for us to have the ability to save material and money meaning that we would not have to restart the build process if a power outage were to occur. With the narrower and more precise filament feed of the Ender 3 Pro this would have allowed us to potentially get a better print for the things we need such as the housing unit or the fiber couplings. This printer also had its disadvantages with the still slow printing speed and even though it was discussed of having less clogging the clogging of the extruder is still possible. The Simple first-party app offers solid print quality and the ability of having prints no stick with flexible build plate allows for this printer to be a considerable option with the only issue being the cost associated with the use of the printer and how we would use it with the UCF ASME and their rules.

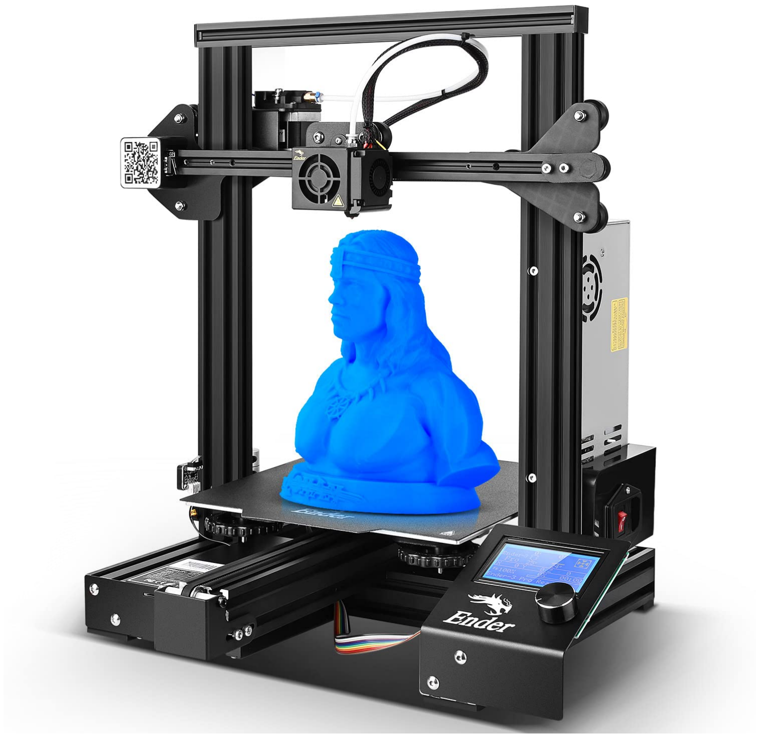


Figure 4.8.4-1. Example of Ender 3 Pro offered (Amazon)

4.8.5 CR 30

UCF has also offered the use of other printers with the CR 30 which uses PLA that can be used to print our design through the help of ASME students as they are the ones with 3D printers.

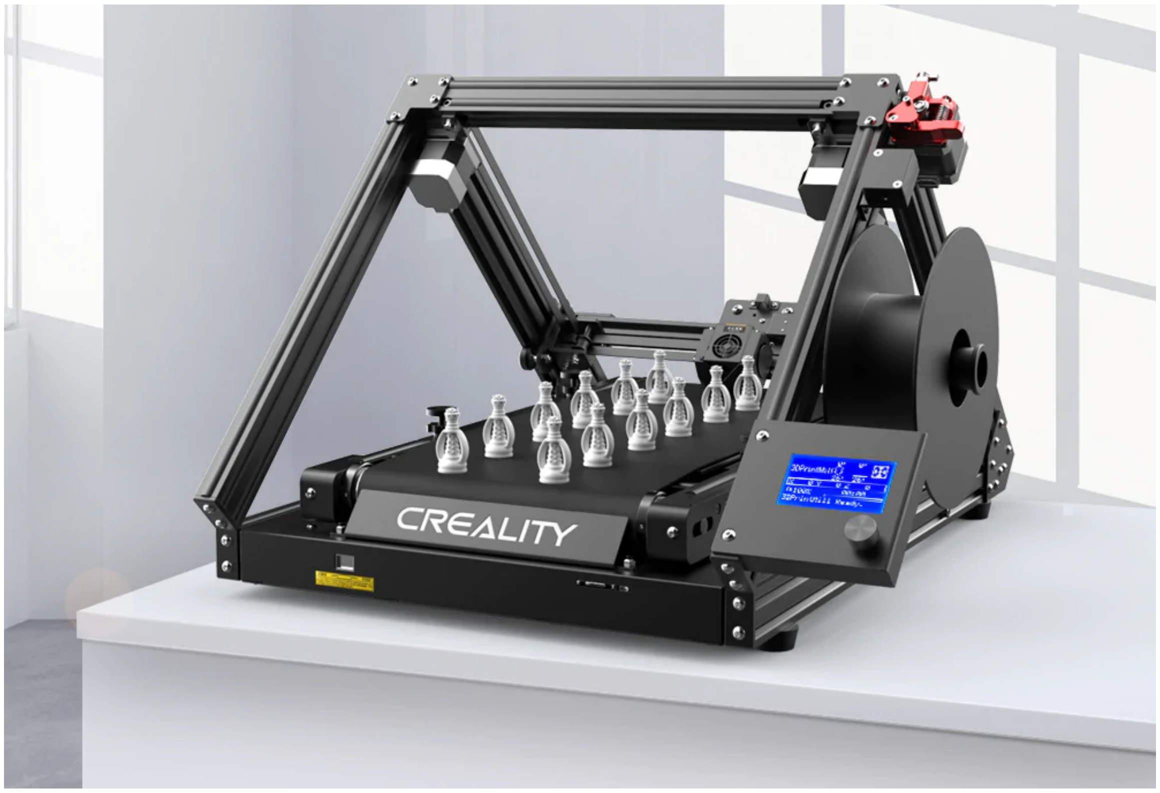


Figure 4.8.5-1. Example of a 3D printer CR 30 offered (Creality)

The CR 30 was the other printer that is offered by the UCF ASME that could be considered to use. Also known as the 3DPrintMill by Creality, the CR 30 is a cross between a traditional 3D printer and a conveyor belt. The incorporation of its unique structure allows for it to conveyor-belt style print bed and 45° printing angle, to print continuously, which makes printing long props in one-piece and batch printing possible. The process in which this printer works is similar to that of the Ender 3 Pro and the Prusa i3, CR-30, plastic filament is fed through a heated nozzle and extruded across a predetermined toolpath using a mechanical assembly capable of moving in the X, Y, and Z directions using a Cartesian coordinate system. One of the benefits that comes with this printer besides its conveyor belt style printing is the fact that its style of printing allows for you to have an infinite-Z-axis to print with, allowing for endless printing. This allows for higher productivity, time saving, and an increase in cost efficiency. This printer also comes with exclusive slice software for printing so that is a huge benefit. The other thing to note about the conveyor belt is that it allows for batch building so if we needed to construct more than one housing unit for the project this style of printer could be considered since it allows for us to save time and increase the productivity of testing our prototypes with having more than one build in the time it would take to print two or more housing units on the other printers. With a maximum printing temperature of 240 ºC and a maximum base temperature of 100 ºC, the CR 30 allows us to print in a large number of filaments such as PLA, TPU or PETG among others with the 3D printer having filament sensor with breakage detection that suspends printing allowing us to save materials.

4.8.6 3D printer Comparison

In this section we used the tables above and below to compare all the exclusive and distinct features and specs between the different 3D Printers discussed in the sections above that could be put into functional use for the Lensless Digital Holographic Microscope project. The comparison tables will help us narrow down and decide on which 3D printer that will prove to be the most beneficial for the development and creation of the parts necessary for prototype creation and testing for our project. Like the coding language comparisons, the cost comparison was taken out of the tables since for now we do not know the amount we need to print and the cost associated with UCF ASME and the use of their printers. Below we have also highlighted the printer that we ended up choosing and using for the completion of our project.

Table 4.8.6-1: 3D printer Comparison

	Prusa i3	**Ender 3 Pro**	**CR 30**
Print Volume	250 x 210 x 210 mm	8.6 x 8.6 x 9.8 inch	200 x 160 x infinity mm
Print Speed	Varies depending on material - machine is capable of 200 mm/sec movement	less than or equal to 180 mm/s, normal 30-60 mm/s	80mm/s and 40mm/s
Max Layer Resolution	minimum being 50 - 300 microns	100 microns	100 - 400 Microns
Extruder(s)	Single	Single	Single
Print Material	Up to nylon, polycarbonate, Polylactic Acid , Polyethylene Terephthalate Glycol , Acrylonitrile Styrene Acrylate , PVB , Acrylonitrile Butadiene Styrene	PLA, ABS, TPU,wood, Copper, gradient, etc	PLA, TPU, PETG
OS Supported	Windows and Mac	Windows and Mac	Windows and Mac

Table 4.8.6-2: 3D printer Comparison cont.

	Prusa i3	**Ender 3 Pro**	**CR 30**
Print without a Computer?	SD, USB Type-B	USB SD-Card	USB SD
File Types	STL, STEP, 3MF, OBJ and AMF	STL, OBJ, AMF	STL, OBJ
Max temperatures	300 °C	255°C	240 °C
Heated Bed	Yes (max 120 ℃)	Yes	Yes (max 100°C)
Best features	SuperPINDA probe , MISUMI bearings , Power loss recovery , Open source hardware , Removable print sheets	Compact design with decent print volume, Prints can be high-quality, Tight filament path improves compatibility with flexible filaments, Magnetic Printing Bed, Aluminum Extrusion for Y-axis, Resume Print Feature (power save)	Heated conveyor belt, conveyor-belt style print bed, Filament out detection Power out recovery, Build volume, infinite Z-axis, Kinematics, Print head, Printing modes

5.0 Design

This section will describe the part chosen based on the research done in section 5.0. Each part will be decided based on the best fit for our Lensless Digital Holographic Microscope as well as describe some of the methods we decided to use with our project.

5.1 LED and Fiber Coupling Design

We had two options for coupling light into the multi-mode fiber. The first option was using an LED array and aligning each fiber until obtaining maximum power efficiency per fiber. We would create a 5x5 output array as well for the fibers, where the ends of the fiber would all be even, and create the light source plane (A). The second option was having the same fiber output setup, but using laser diodes to power the system instead of LEDs. This option would have provide a much greater power over LEDs, but came at a dramatically increased cost. Another benefit of using laser diodes instead is the ease of coupling. Coupling a laser beam into a fiber is dramatically easier than with LEDs, because LEDs have a greater radiation angle than that of a laser with a constant beam diameter.

We used most likely use an alternative to standard LEDs, which is parabolic LEDs or LEDs with a much smaller radiation angle. Standard LED radiation angles tend to be within 120 to 160 degrees. The fiber diameter is much smaller than the LED's coating diameter, so we needed to use an LED with a radiation angle as small as possible. This will help the couple as much light directly into each fiber as possible.

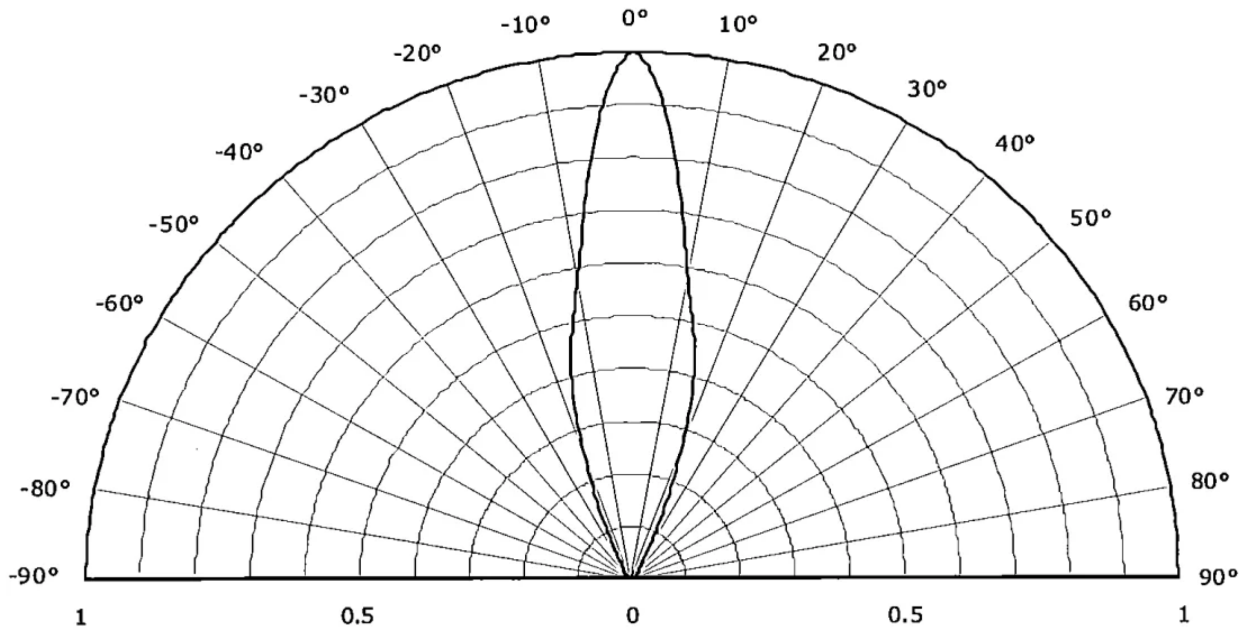


Figure 6.1-1. LED Radiation Angle Diagram

This diagram (figure 6.1-1) depicts an LEDs radiation angle and how we want the angle to be as small as possible to couple the maximum amount of light into the small diameter multi mode fiber.

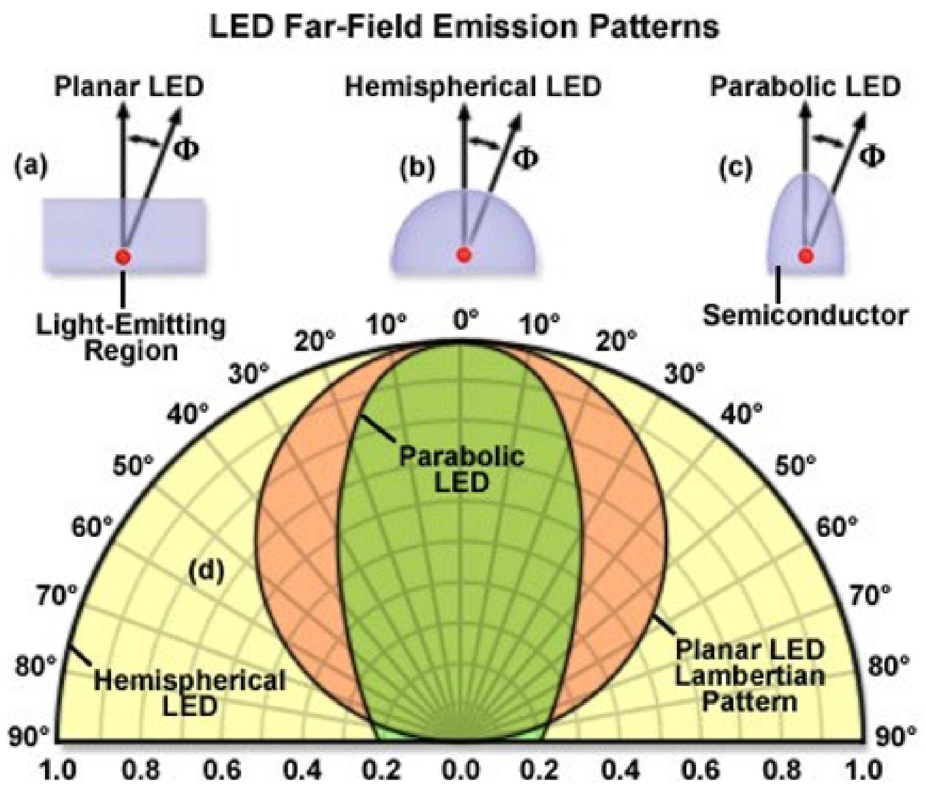


Figure 6.2-2. LED Radiation Types Diagram

5.2 Microcontroller and Electrical Design

After some research, many of the options for both the microcontroller and CMOS sensor do not appear to have accessible files containing their symbols, and footprints for easy use in EAGLE, since the performance of these two parts of the device, need to meet certain criteria and the cost of these parts are a significant factor of why we pick a certain part over another, whether or not it has a symbol and a footprint should not be a limiting factor. It was difficult to get an exact symbol and footprint working from scratch, but it is within reason that the part did fit our needs, it doesn't need to have a readily available symbol and footprint. We were able to find a similar part and edit it a bit to more closely match what the symbol was, all within EAGLE, and be able to present the official schematic with the parts selected. Also, since the Lensless Digital Holographic Microscope will have an array of 25 LEDs, the concept of each of them is the same, there will be only two LEDs wired into the Schematics, to keep the design simple and readable.

5.2.1 Electrical Schematics

After some research conducted the final schematic might change a bit to accommodate the restrictions of the microcontroller model that we went with due to the different microcontroller’s power methods that were available. The schematic shown below is more for the theory behind how we want to wire the Lensless Digital Holographic Microscope, so if there is an additional step to the process of a certain microcontroller it won’t be seen in the schematic unless we choose that microcontroller, after which the schematic will show the full design. Also, CMOS sensor developers don’t seem to upload CAD models to the degree that microcontrollers do so a schematic shown in EAGLE is more likely to use a stand-in part for the CMOS sensor than any other part.

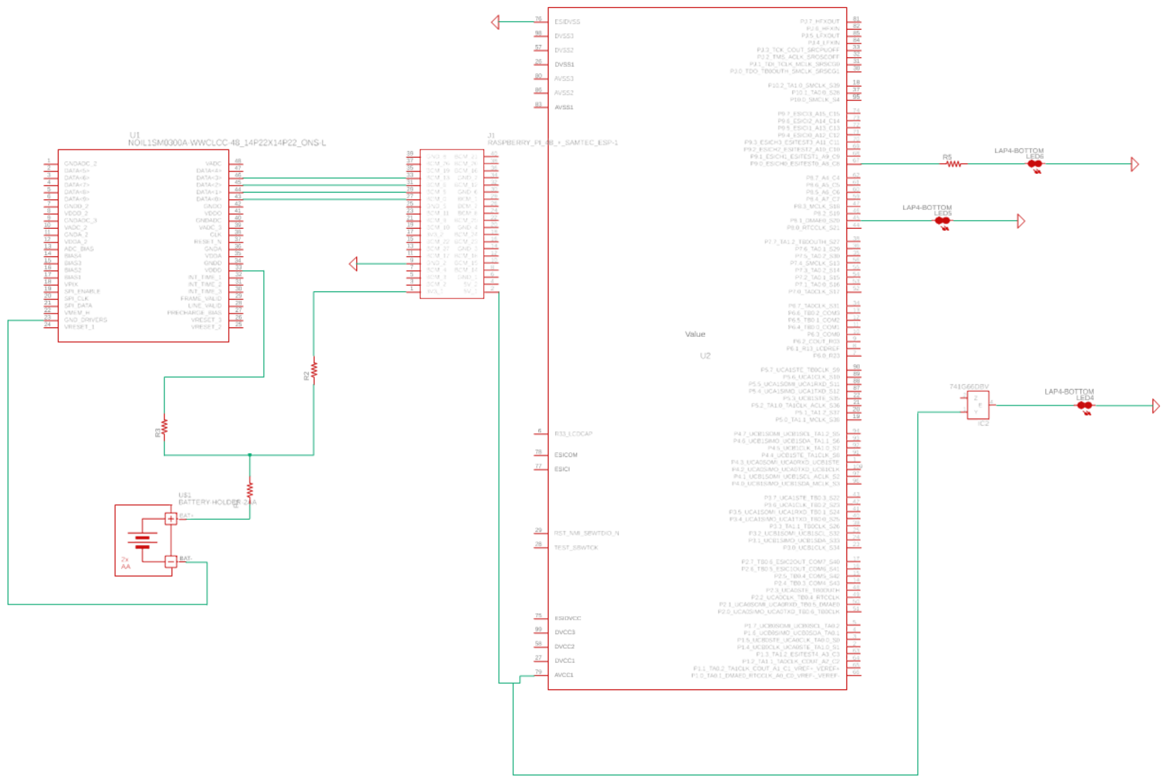


Figure 5.2.1-1. First Eagle CAD schematic

The following section, until the end of this part of the paper will be in part analysis and explanation of the schematic found in Figure 5.2.2-1. For a quick introduction of the parts shown here, the model on the bottom left is the main power source of the Lensless Digital Holographic Microscope, for this example it will be a simple two AA battery pack, the final model will need more power than this, however this is a good stand in for now. The part on the far left is a CMOS sensor stand in, the NOIL1SM0300A if curious, the part in the center is the Raspberry PI 4, the 2 wasn’t available at the time, however their pin count and pin map are the same and are about the same for this schematic. And lastly on the right that is the MSP430, with a couple LEDs to demonstrate the ways we could handle the LEDs, once testing is done all 25 will be wired the same way. As a side note pretty much all the pins on the right side of the MSP430 is a pin that can be easily toggled to turn the desired LED on and off, this is the reason why we are confident that this microcontroller can handle the LED array.

So, the CAD model for the CMOS sensor doesn't really matter for two good reasons, the model shown in the schematic is just a CMOS sensor that was available on ultra-librarian, there are several thousands of CMOS sensors available on digikey, and about six of them at the time of writing have a CAD model ready to be used in EAGLE. One of the reasons why we are comfortable with just using a stand-in CAD model is because we know that how the CMOS will be implemented into the design of the Lensless Digital Holographic Microscope doesn't translate well to the schematic. The only thing that would show on the schematic is that power is going to the correct pin and the ground pin is going to ground, this is the only pin to pin connections, and they are aspects that wouldn't just be on any CMOS sensor but are basically on any electrical component. Additionally, the other reason is that the CMOS sensor is going to be connected to the microcontroller via a USB port which won't be shown accurately on the schematic, but for the sake of example there are a few pins connected to the microcontroller to show that there will be a connection between these two parts.

Lastly the connection between the Raspberry PI and the MSP430 with the LEDs, demonstrates how the MSP430's power depended on the testing with the Raspberry PI, the connection is meant to show that the Raspberry PI can reset the MSP430, in the current schematic, effectively turning it on and off will reset the code, which will make the LEDs turn on and off in sequence. The MSP430 was connected to the Raspberry PI via a USB connection in the final build if we are confident that the main microcontroller has enough power to do the image rendering at that stage while powering the other microcontroller, if that is not the case, we can just get a larger power source, and do a three-way voltage divider to power all the components individually. Next on the far-right side there are multiple LEDs, they are best represented by just a basic diode, configured in a few different ways, we will decide on which to do after some testing. For now, the from top to bottom we have a resistor in series with the LED, to show that if the voltage is too intense for the LED we can lower the voltage to the acceptable value, the middle set up is for if the voltage is roughly sufficient, we can simply wire the pin directly to the LED. Lastly if we find that the MSP430 cannot provide sufficient voltage since we need the LEDs at maximum brightness a switch gate can be used, a pin from the microcontroller will be enough to serve at the activator, and source of the power for the LED in this case would just be the base power source, again we can just have a three or four way voltage divider in the Lensless Digital Holographic Microscope. In this final build the operating of the LEDs would still be the same, and likely the same code can be used, meaning only one LED would be on at a time so the effective voltage demand of the entire LED array would be the voltage requirement of a single LED, and we wire all the LEDs in parallel. In summary the schematic is meant to show the basic circuit theory, and our intent to wire the electrical system needed to implement our chosen components into a final design, there is still a lot of small work that needs to be done however once we have the parts, we need then we can work out how exactly the entire system will work.

5.3 Software Design

The research portion of the project life cycle is has been conducted. As we tested samples and objects in order to test the CMOS sensors resolution we emphasized and concluded the application of programming languages. The languages were narrowed down to Python, C, and MATLAB as the MATLAB helps with the strenuous calculations that took place with the photonic equations and theories. We narrowed down to Python because of the ease of syntax and programming as well as the simplicity that the language offers over languages such as C and C++. With the CMOS sensor languages such as C and C++ were still viable options especially due to the fact that the CMOS is written in C. Since both the development and target machines support C-based code, it was also feasible to do some initial debugging in the time-shared environment. The thought of handling debugging was not the top priority at that moment in time as selecting the correct CMOS sensor for our project was top priority and it ensured that the features we want to tackle could be tackled. With this in mind the conclusion of using Visual Studio Code is one that was much easier to come to than the programming language. The features that Visual Studio Code offered was unmatched and allowed for maximum efficiency when it came to programming. With its lightning fast features as a source code editor it was perfect for the days in which coding was just not a one day process but an entire trial and error cycle. Eclipse was good option but since our target languages fall outside of the languages for Eclipse, VS Code was the obvious choice and it helped with instant productivity with syntax highlighting, bracket-matching, auto-indentation, box-selection, snippets, and more. Another reason VS Code was the front runner was with debugging. With VS Code there was an interactive debugger that helps with source code, inspect variables, view call stacks, and execute commands in the console. VS Code's ability to also integrate with builds and scripting tools to perform common tasks helped make our tasks in the project's software development cycle became much faster. VS Code had support for Git so you we could work with source control without having to leave the editor, including viewing all the pending changes diff. This not only allowed for branching which was mentioned in the Software tools section, but meant that we could use GitHub as well for version control and keeping track of our projects progression throughout its life cycle. Once the domino of the CMOS sensor was chosen, the Software Design followed along and the project took off allowing the computer engineer to have an understanding of the photonic equations and how they were implemented when the software cycle began.

5.4 Mobile Application Design

At the moment how we would have liked to deliver the end result of the Lensless Digital Holographic Microscope, to the device of the end user's choice, so either desktop, for efficiency or mobile for accessibility. This such feature was a feature of our stretch goals which required us to accomplish a mobile distribution aspect for the project. Seeing as portability is one of our main concerns it is clear to us that different locations may have different requirements. Wi-Fi is not global. Bluetooth and subsequently AirDrop can be. This is why we believe that it was important to be able to implement Bluetooth communication into our microscope but in the end due to time and limited funds and resources the mobile application was not able to be completed.

6.0 Project Testing and Prototype Construction

To test our design, we compared our prototype's functionality of the working designs with the project objectives and specifications that were discussed within the design specifications for the Electrical , Photonics, and Programming side.

A demonstration of the optical components used for the first optical demonstration are discussed below. We also discuss how we planned to test our project as a whole and its individual component in regards to our teams make up to ensure that it worked properly. The object or topic of what we wanted to image was still in the research process for the optical demonstrations but we concluded of what we may be testing. The ideas include: Oil monitoring in Ocean Water (Tracking runoffs, seeking oil sources) Air health (Bioaerosols, Pollution, Mold Spores, Bacteria) Plant health (Cell burst, Cell Hyponatremia). First step was to communicate the CMOS with the computer. Second step was to image the interference patterns of a Hologram. Third step was to back propagate the hologram to demonstrate viability. Fourth step was to implement more light sources to add Pixel Super Resolution capabilities into our microscope. Fifth and final step was to demonstrate the communication between the microscope and the smartphone if we could get to that stretch feature.

6.1 Optical Demonstration

This demonstration is not to demonstrate a complete working product but to show that the optics that make this project work can function as intended. The systems being tested must also adhere to the design specifications listed in section 2.4.

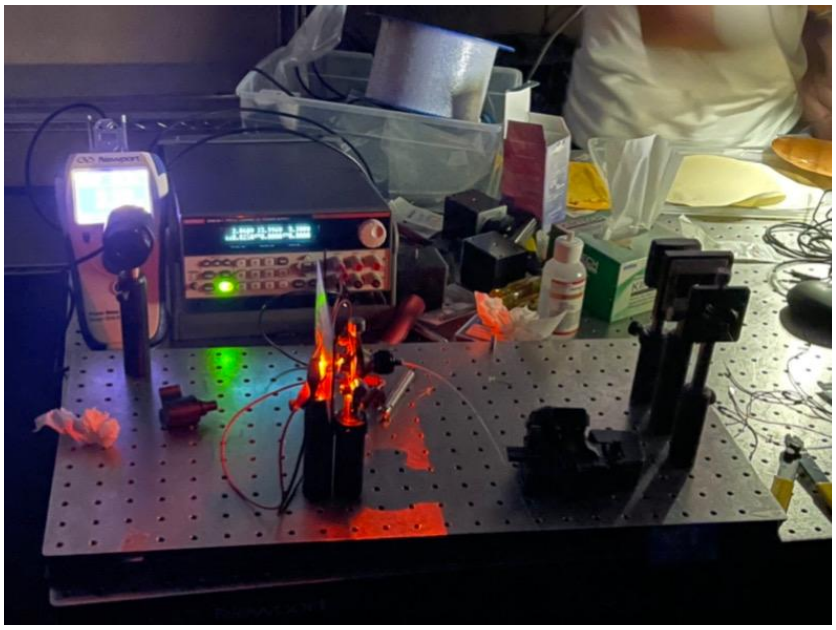


Figure 6.1-1. Initial Demonstration of LED-Fiber Coupling

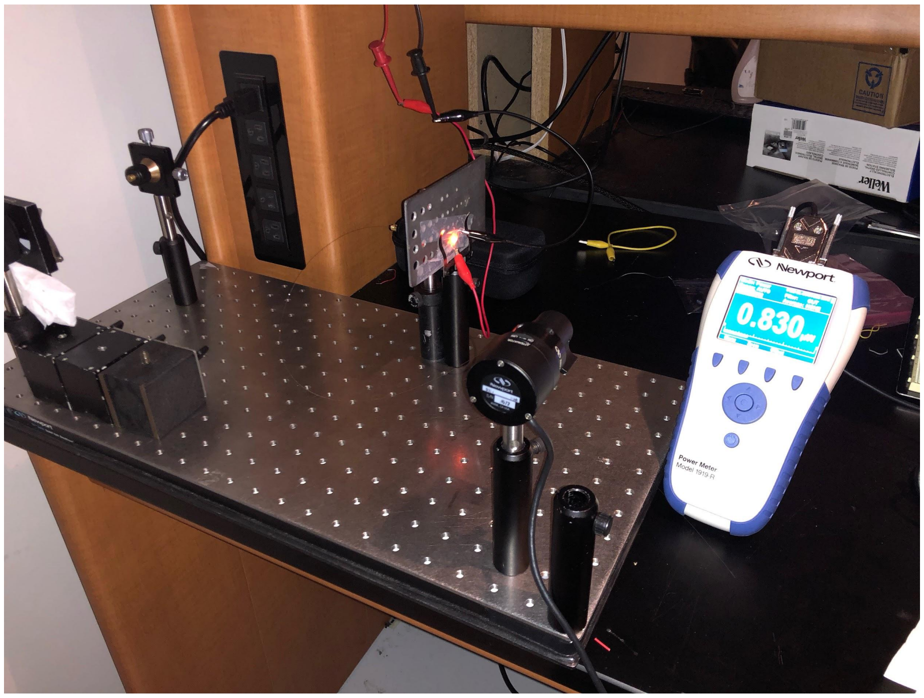


Figure 6.1-2. Part 2 of Initial Demonstration of LED-Fiber Coupling

In the above figures is the first creation of an optical demo we had created this far into the semester. Above you can see the components discussed in the 3.0 section.

Objective: Couple a significant amount of light into the fiber and image an interference pattern on the CMOS

Environment: This was done at the Center for Research and Exploration in Optics and Lasers.

Conclusion: Fiber Coupling was successful.

6.1.1 Optical Demonstration Components

The following sections will explain the optical components that would be used to demonstrate our design process. In our demonstration we were able to demonstrate how the Lensless Digital Holographic Microscope would work.

6.1.2 In Class Optical Demonstration One

During our Optical Discussions demonstration, we were able to demonstrate how the optical components worked and how we were able to get them to work towards getting them to function. We were able to test both the camera system as well as the IR LED portion of our system.

Procedure for Demo 1:

1. Sanding Down LEDS.
   a. This was done because the LEDS we used for the optical demo were for temporary use. The curvature of the housing unit made the light spread. By sanding the LEDS down with 30 micrometer, then 9 micrometer, then 3 micrometer sanding pads, and finishing it with a polishing cloth, we were able to have a flat surface where the most light could propagate out without the effect of the led curvature.

2. Fiber Positioning.
   a. Once we had the flat LED, we connected it into our system and held the fiber close to it using a fiber chuck holder on a translation stage in the z-direction (towards or away from the LED). The fiber chuck holder can move in the x and y directions. This allowed us to gently tune the position of the fiber to be able to couple the most power into a power meter using the fiber holder adapter for the power meter.

3. Resin Glue the Fiber to the LED.
   a. Using the UV Resin picture in Fig 6.1-3 we unified our LED-Fiber system. We observed a drop in power. While this drop was initially attributed to a refractive index issue, in the demo 2 we explained a new hypothesis.

4. Position other Fiber End pointing towards the subject plane and imaging plane.
   a. Now that the fiber is coupling the most power. Point it towards the object that will be imaged. The position of the Fiber matters. A large Z1 distance (Fiber to subject) needs to be much greater than Z2 (the distance from the subject to sensor) to properly image the hologram along the sensor plane.

5. Image using CMOS and tune
   a. Image what you now see on screen. Tune the imaging software if needed.
   b. We used an ND filter on our CMOS, this is not always necessary.

6. Pixel Super Resolution
   a. Using pixel super resolution algorithms we can increase the resolution of the holograms imaged.
   b. We only used shift-and-add for this demo but there exist plenty of other options.

This was all we did for the first optical demo.

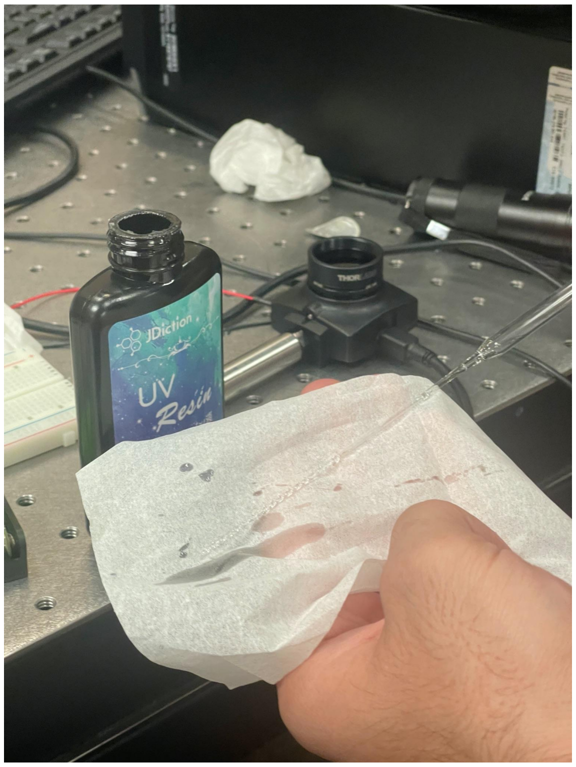


Figure 6.1-3. UV Resin being used to secure fiber to LED after coupling.

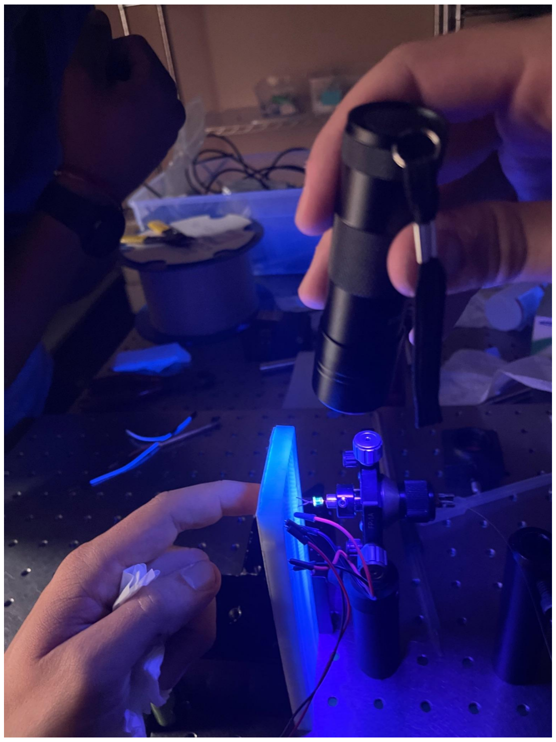


Figure 6.1-4. UV Flashlight Curing UV Resin

6.1.2 In Class Optical Demonstration Two

During our In Class Optical Discussions demonstration, we were able to demonstrate our optical components and how we have worked towards getting them to function. We were able to test both the camera system as well as the IR LED portion of our system.

Disclaimer: Our optical demo two was riddled with technical errors and setbacks. While we learned many new techniques the procedure will be detailed as it worked with us for clarity, candor, openness of communication and reproducibility.

Procedure:

1. Measure Fiber Chuck

2. Model and Print New Fiber Chuck
    a. Using the model of the original fiber chuck, recreate it in solidworks. Instead of having one slit, create five slits all equidistant from each other and from the center of the chuck. Print this chuck. Realize that the slits are failing to hold your fiber. Make the slits into holes instead. The slits are there to hold the fiber, if you make a hole, it will not be held as securely. Realize that the holes can be filled with the same UV resin to hold it in the chuck. Continue with the printing of the hole'd chuck instead of the slit one. (This is all after many many issues with printing and slicing, one issue was the holes would get covered by the other print lines if it was extruded too hot.)

3. Insert New Fiber Chuck into holder
    a. Ensure the chuck fits.
    b. Resin glue the fiber into the chuck
    c. Try and decipher why the chuck no longer fits. Surmise that it was because the Resin expanded within the fiber and now no longer fits.

4. Measure Correct Spacing
    a. 15 cm to 2cm.

5. Image a Hologram
    a. This step was not reached on account of the fiber chuck no longer fitting once we used resin.

Solutions:

1. We struggled with printing.
    a. Solution: look into using a smaller extrusion nozzle size.
2. We struggled with fitting the fiber chuck into the holder.

    a. Solution: print a new fiber holder. Which we were going to do anyways on account of the fact we need three to five chucks in one holder.

3. Problem chuck holder now no longer supports fine x and y tuning.
    a. Solution: No longer a problem because at Z1 >> Z2 and using our fiber, the fiber acts as a point source and does not need fine tuning that way. Also, we had the distance from point to point measured in our STL file regardless.

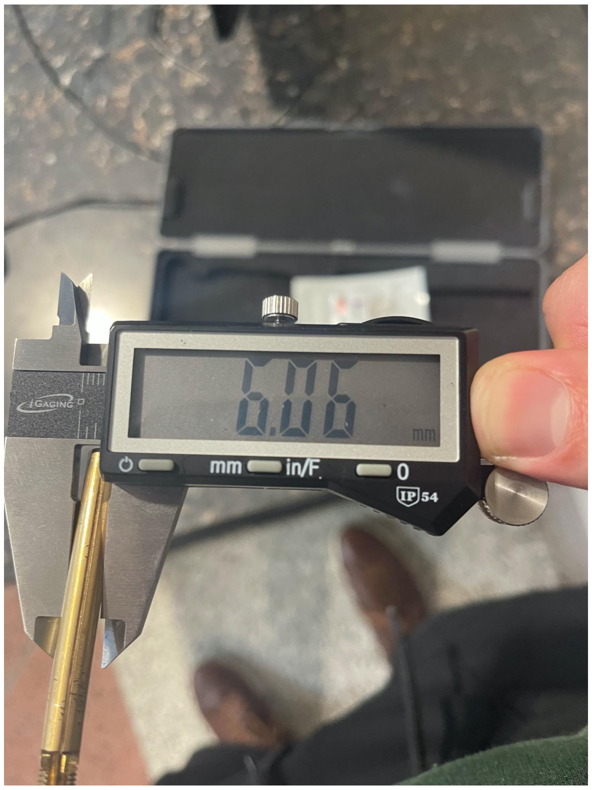


Figure 6.1.2 Chuck diameter, needed to fit printed chuck into chuck holder.

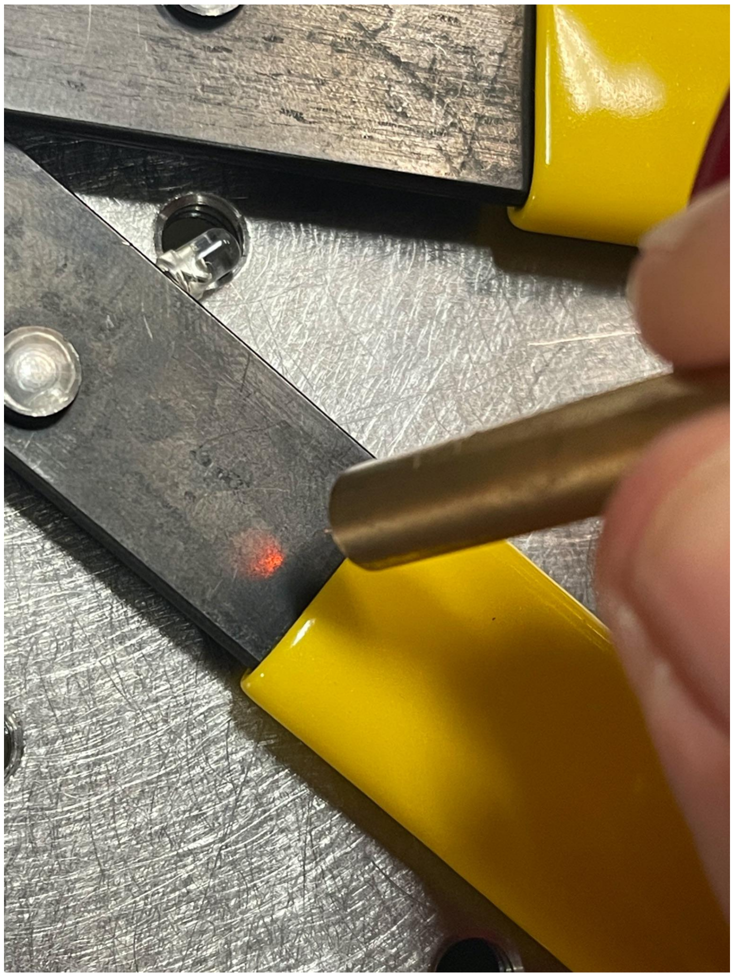


Figure 6.1.2-1 Spatially Coherent Light Exiting Fiber as a Point Source

6.1.3 Further optical Tests

Further optical tests were to include a reconstruction of the hologram to indicate our math and our spacing has been correct.

6.2 Microcontroller Testing

Since there are two microcontrollers to be used in the Lensless Digital Holographic Microscope there are going to be two simple tests to make sure that we are safe to move forward with the design. There was a delay in the delivery of the Raspberry PI 2 so this will be, in short, a plan to do testing.

6.2.1 Raspberry PI Testing

Since this part is known for being a computer you can use as a microcontroller, we were confident that it can handle the processing needed to accomplish our goals, so it did not need traditional testing. What we would have liked to test was the connections of this part to any CMOS sensor that we can borrow with the other microcontroller to see how they are all powered. In short, whether we could use a simple connection to power the Raspberry PI via its' micro-USB port, which is how it is recreationally powered, or through a technical pin-to-pin connection. A pin-to-pin connection would be more efficient, and for now we were not planning on using any other pins so it would be easy to wire and solder on if that were necessary, however we believe that a proper connection via a micro-USB at the voltage and current listed would enable one or both other components to just be wired through the Raspberry PI. Again, if it was clear that that is not the case, we would just need to invest in a larger power source and make the proper voltage division between all the components. In summary, we would have liked to begin this testing but without the requisite components, it would have just led to planning using figurative values that might not have any basis in reality, so we just wanted to share our insight with how we plan to test with the Raspberry PI to show that we don't plan to just drop it into the design and hope for the best.

6.2.2 MSP430 Testing

In short this is a check that the MSP430 can fully power a simple LED in an array, the LEDs used aren't the LEDs that were to be used in the final build, to be safe, however the voltage requirements for both LEDs are about the same therefore it was still beneficial to do this kind of test. At the time of writing this served more as a reminder to remember how to program the microcontroller and turn on an LED which worked almost immediately, and for the upcoming demo we were now confident that we can just plug in the MSP430 into a nearby computer and it would work. This test would need to be reconstructed, once we came to the conclusion about how much voltage we needed, and how all the components are wired. As mentioned before there are a few different ways to wire the LEDs if there is an issue with powering the LEDs in this device. If we end up in a situation where the MSP430 can just be connected to the Raspberry PI via a USB cable, and both microcontrollers are operating at the appropriate voltage, with the LEDs operating at maximum brightness, the MSP430 would be essentially working in the manner that it was working for the demo. Additionally in the following figure 6.2.2-1 there is a quick picture showing which pins were used for the demo.

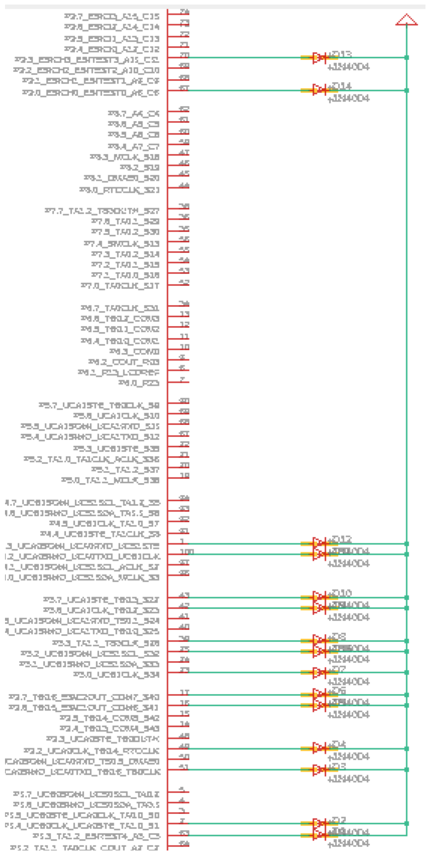


Figure 6.2.2-1 Demo Microcontroller Setup

The inclusion of this Eagle picture in 6.2.2-1 is intended to show that we have tested these few pins and now are confident in the MSP430 to control the LED array in the final build, and surprisingly has enough voltage for each pin to make each individual LED to be at maximum brightness. There is one small problem that occurred during testing, and might cause a change to the pins in the final build. A few of the pins that were selected aren't GPIO pins so they aren't intended for this role, however there are simply not enough GPIO dedicated pins without using the boosterpack which would add an additional cost of fifteen dollars to the bill of materials. These pins would still be used in this manner because we don't need to use them for their intended purpose, so the issue is that some pins are set to 1 at the beginning of the sequence so at the beginning a few LEDs would light up at the same time before all LED pins are masked off. This wasn't an issue since it is only on powering on the device, but a simple work around would be possible if we can find 25 pins where this isn't an issue. To clarify all the LEDs turning on at the beginning should affect the final product, and won't cause any electrical problems, this is closer to a visual quirk than a design concern. Also in figure 6.2.2-1 you can see that all the LEDs are going to the ground, it is completely fine for this to be wired in this way since only one LED will be powered at a time. The reason we wanted to repeat this is that going forward we are going to test if it is ok for the system to run that ground through the MSP430 or if it needs to lead back to the on-board battery.

6.3 Software Testing

Testing the software was crucial to ensure that the system would perform and be able to be measured properly. The system would need to be able to power on correctly, be able to read in the sensor signal, and precisely give the delta time between each sensor signal. The processor would also need to be able to correctly display the calculated speed on the LCD display, and the MCU would need to correctly generate a PWM signal that matches the desired frequency. In addition, the PBITs would need to correctly display any errors that occur, if they were to, on the LCD display.
Objective:

Environment: The development environment was fully contained within the raspberry pi and additionally microcontroller. Wherein, the synchronization of the screen capture and LED flashing was done within the microcontroller, the setting off by the signal was done by the raspberry pi. The raspberry pi would then process the image and provide what we so that we could transfer the data capture to the MATLAB software on a pc. The production environment will be the button to turn on the microscope, as well as the email/airdrop/bluetooth image the smart phone will receive.

Procedure: The procedure would implement the matlab code fully on a pc for its heavy CPU processing and graphics processing. The microcontroller will send a simple on off signal to the LED array and camera at the same time. LED[z] will go off with CAMERA, and then the image capture will be PICTUREx-y where x-y specifies the two dimensional plane the array is in.

Conclusion: The conclusion of the software was that the system would be able to communicate with the computer and presenting the image of the object we are imaging.

7.0 Design Constraints and Standards

In this section we are highlighting the standards and general limitations we need to account for during the project life cycle of Lensless Digital Holographic Microscope. These constraints may be financial, ethical, technological, or legal constraints. These must be noted and observed accordingly for this product to be successful in meeting the product's applications.

7.1 Standards and other Safety Concerns

In this section, the project cycle shifts its focus to various ethical and safety standards that are addressed and examined. This section has separate discussions pertaining to each required standard and how it relates to our design. This section outlines the ethics of working with photonics, LEDs, electrical components, safety protocols of working with resins, UV rays, and many more highlighted within this section.

7.1.1 Standards for Working with Optics and Photonics

We followed the standards demonstrated here at UCF and the standards created by the IEEE Photonics Society which covers the standards in areas that are: lasers, optical devices, optical fibers, and associated lightwave technology and their applications in systems and subsystems, in which the quantum electronic devices are key elements. We also followed that Laser safety standards set by OSHA and ANSI standards for laser use.

7.1.2 LED Standards

LEDs have radiation emission limits as per the FCC. Between 30MHz and 10000 MHz.
LED suppliers must also fill out FCC Supplier Declaration of Conformity (SDoC)
Heavy metal limits. Florida is one state that *does* regulate the amount of heavy metals in LEDs

7.1.3 Programming Standards

The programming languages were narrowed down to four languages: C, C++, Python, and MATLAB. Since we were able to use all the languages all the standards will be covered for all four of them. For the C programming language we used the ISO/IEC 9899:2011 which specified the form, structure, and established the interpretation of programs that were written with the C programming language. It also defined the representation of the C language within the programs following the syntax and constraints of the C language, covering the semantic rules that we took in mind when we were writing and interpreting C programs. This also covered how we would want the input data to be processed and how the output data was to be produced by C programs, and the restrictions and limitations imposed by a conforming implementation of C within our programming. Since C is a programming language that is prone to risk and the code can be affected by vulnerabilities and defects with its unpredictability ported between different hardware it is key that code we made followed this coding standard in order to help prevent undefined and unpredictable behavior. For C++ it was a different and more complex language than C so the standards were similar in some aspects but were vastly different in others. Like C, C++ was very flexible and its ability to be used for high performance. It wascrucial to maintain a codebase and have it be maintainable while using the C++ coding rules. This allowed for the code created toensure that undefined and unpredictable behavior stays out of the code. There were also several established standards for C++. Some are specific to embedded industries concerned about functional safety — including MISRA, AUTOSAR, and JSF AV C++. Others were designed for secure coding, such as CERT. These standards help us understand how Names were to be used, how the code and syntax was to be formatted for C++ programs, how we were to cover the Classes used in the programs, documentation within the program through headers and more, and how was follow along with the things of the program covering Complexity Management. As for MATLAB the coding standard has only slightly changed over the past 12 years with the priority of the standard being concerned more with the correctness, clarity and generality that the code was written in. The goal of the standards was to allow for the code to be clear and correct and allow others to understand the processes that were written. These MATLAB coding recommendations are unchanging with the best practices in the software development community and the standards that were to be used within MATLAB are generally the same as those for C, C++ and Java, with modifications for MATLAB features, syntax, and history. Finally the coding standard for Python was very similar to that of C and C++ with regards to how the coding language was to be used in order to limit and eliminate the amount of undefined and unpredictable behavior and making sure they stay out of the code.

Any use of the C, C++, Python, and MATLAB programming language to develop software for the Lensless Digital Holographic Microscope conformed to the programming language standards above in order to help the code be consistent and easily maintained while allowing for the development process of the software programas to be less complex and thereby reduce the errors that were encountered. Following these standards allows anyone to understand it and can modify it at any point in time.

7.2 Realistic Design Constraints

The Lensless Digital Holographic Microscope has realistic design restraints as listed below in sections 7.2.1 through 7.2.11. During the development and brainstorming of the project, various constraints were being considered during the project life cycle in order to create an optimal product that was price effective, ethical, and safe. Each constraint has been carefully considered in regard to its realism and how it will be applied to the design and our final product.

7.2.1 Economic Constraints

After having conducted research into parts like CMOS sensors it was clear to us this was an expensive project.We knew as a group that there was always going to be concern for purchasing certain parts would make us uncomfortable because we wanted to use parts that we would be confident in using. This constraint did not stop us from trying to deliver the best product that we would be able to make given our time limit, however we tried to find a sponsor to alleviate our financial concerns, or at least reimburse us in some way but in the end we did not find one. The remainder of the parts were not cheap comparatively, however throughout the completion of the project we continued to look for ways to reduce the cost such as using our school’s available 3D printers, in a sense we would aim at creating the most affordable version of our project that we were confident can deliver a good viable end product.

7.2.2 Time Constraints

Now that the lingering effects of the pandemic are mostly behind us, there shouldn't be a time constraint on us simply getting the parts we need to complete the product. Therefore the only implicit time constraint on us would be the deadlines our advisors put on us to deliver progress. However, that is a constraint imposed on the entirety of our graduating class, so there shouldn't be too much of an issue here.

7.2.3 Environmental Constraints

Unless otherwise specified all the microcontrollers and CMOS sensors discussed here were all RoHS (Restriction on the use of Hazardous Substances) compliant. Therefore we feel that using any of these parts would meet any environmental standards that have been set for us.

7.2.4 Ethical Constraints

For the time being we haven't suffered from an ethical dilemma during the initial research of the Lensless Digital Holographic Microscope, so we feel that as long as we attempt to deliver the best product we can, we should be able to avoid any ethical constraints.

7.2.5 Sustainability Constraints

Our sustainability constraints were similar to our environmental constraints, since we wer using RoHS approved parts, we do not have a relevant constraint.

7.2.6 Social Constraints

The closest thing to a social constraint that we have, is the social pressure to try to make the Lensless Digital Holographic Microscope a successful senior design project, which is a constraint that is helpful to have.

7.2.7 Political Constraints

After examining the project carefully, it was determined that any potential political constraint was not relevant to the Lensless Digital Holographic Microscope product.

7.2.8 Health and Safety Constraints

The Lensless Digital Holographic Microscope would be producing images for the customer, and it was a consideration for us that the image was presentable, so that it wouldn't hurt the end user's eyes. This was a constraint that was worked out during the prototype phase since we wanted what was best for our health as well. Additionally since we are using electrical components we were going to have to make sure that all of the connections were safe, and resistant to active use, to ensure that our customer that the product that they have purchased was safe to use.

7.2.9 Programming Constraints

When programming for Lensless Digital Holographic Microscope, one must consider the constraints that could come along with programming and creating software for its application. When doing this we are able to come to think of a feasible solution to common issues. By keeping this in mind we were able to make the project and frame the algorithms for the many different problems that came up. Analyzing the constraints can give us crucial information like the possible edge cases and the expected time complexity of the solution.

Some of the constraints discovered at the time of the early design process and final product were:

- understanding of what language would be needed for coding with the microcontroller and sensor
- the planning for the coding of the project understanding that the code software will have multiple iterations
- time crunch with the coding and communication with fellow group mates to make the code work along with all other components. The ability to stay up to date and record the many project changes
- resource constraints with the potential application of a phone application with the programming of Bluetooth and the app itself

7.2.10 Electronic Constraints

For the electrical side of Lensless Digital Holographic Microscope it should be clear that we have deliberately kept the circuit design simple to avoid the implicit and explicit constraints that we could face when implementing our chosen design. In short, the core concept of what this product needs to do electrically has been nearly unchanged from the first conception and the design rhetoric we used, was simply the best way we could think of to wire this project's individual parts together without breaking the laws of circuit theory.

Some of the constraints discovered at the time of writing this paper and the early design process are the:

- To have a sufficient power source that can provide enough energy for the microcontroller and the about 20 LEDs

- The device is expected to be able to be used multiple times in a single session so the battery life must be able to do go for a couple of hours at least
- The microcontroller needs to at least send the images via Bluetooth to the device for process, or hopefully be able to do as much computation on the embedded system as possible

This section is quite old versus the rest of the document, so it would be helpful to leave this for prosperity and loosely revisit these with the knowledge that we have now. As with most things in this project's life cycle we have thought a bit too far ahead, trying to think about what experience we want for the end user, trying to think about how a microscope is used by a professional sounded like a good idea and trying to fill out a list of what they would want from their microscope is what we tried to do, and then convert that into what would we need to make it work electrically. However, when we think about the constraints we immediately face, they somehow seem simpler, to the point where we prefer the list of assumed constraints that we made months ago. Even though the electrical design we decided on ended up being slightly more complex than we initially thought, with two microcontrollers instead of one the most impactful electrical restraint will probably be when we go to wire the final build what is going to be the most accessible, if we need to ground the LED array into the microcontroller or the on-board battery. Other than that we have been free to design this with the only real restraints being the nature of the components that we need to work with to get the project done, which have felt more like design restrictions than constraints, we didn't have a preference when beginning the market analysis and the circuit theory of this project so this design restrictions just made it felt less daunting to begin the EAGLE process of the design.

7.2.11 Photonics Constraints

When using and working with Photonic elements for Lensless Digital Holographic Microscope, one must consider the constraints that could come along with handling and creation of the physical optical product. When doing this we are able to come to think of a feasible solution to common issues. By keeping this in mind as we make the project, we are able to tackle the different problems that arise from making the housing unit, creating the optical components for demos, and more. By analyzing the constraints we are able to gain crucial information such as the expected time complexity of finding a solution to a problem allowing us to create the best product possible.

Some of the constraints discovered at the time of writing this paper and the early design process are the:

- Optical Shadowing: Blur caused by anything, not on the imaging plane
- Twin image artifacts: Unavoidable extraneous data, can be mitigated by spacing out twin image and image.
- CMOS CCD Pixel Size restrictions: Limited by current hardware limits and finances.
- Speckled Interference Pattern: Caused by monochromatic light, will be using partially chromatic light.

- Multiple Reflection Interference: Can be mitigated using partially chromatic light.
- Structure around 16 cm or less. (soft restriction)
- Image subject must be semi-transparent

7.2.12 Testing/Presentation Constraints

The constraints that we faced during our testings, in either our independent testings or our live demos were for the most part self restricted. We needed to test our components, both on the optical front and the electrical front both of which can be done at either the CREOL building or the senior design lab respectively, the equipment there proved to be enough to test out the parts we have chosen to use in our design of the Lensless Digital Holographic Microscope. The largest constraint we suffered was our accessibility to our parts. Since we were working with safe components from good retailers there wasn't a reason we would have an issue with testing our design as long as we are careful doing so. Additionally for the demo we made use of a nearby computer to power the microcontroller that runs the LED array, delaying the time we need to set up an on board battery to begin the testing of the optical side of this device. This is a rather large example of the lack of constraints that we have had our testing and our presentations.

7.3 Project Design Problems

As with nearly all design processes, several problems occurred during the creation of the Lensless Digital Holographic Microscope. Problems which have become part of the project has been recorded in order to understand the issues which created these problems and learn from previous mistakes.

7.3.1 Photonics Problems

Initially we struggled to have the LED and fiber remain connected after we used the resin, this was solved by printing a snap away piece that lets us use more surface area of the LED to connect it to the fiber. Then we ran into the issue of how to best couple light into the fiber, but we found that parabolic LEDS work best. This is further explained in section 5.1. Other issues we had were the inability to have the fibers form an array using the chucks provided, this is why we printed our own.

7.3.2 Microcontroller Problems

Other than the delay of the Raspberry PI 2, which set back our testing schedule the only problems with the microcontrollers have been in finding candidate models for our market analysis. It is a bit of a shame that we initially was grouping on board computers with microcontrollers since after a good amount of research further we realized that basically the choice was clear a Raspberry PI was needed, and the role of the microcontroller changed drastically. Since it went from a component that could both process the image while controlling the LEDs we see now that a two microcontroller design is better moving forward. Problems are meant to be more of a learning experience than a set back and we were happy that we made this realization now rather than next semester. There will likely be loads of problems once we begin designing the power flow of the Lensless Digital Holographic Microscope, however since we are working with relatively few complex components we can afford to take our time and be safe with our testing. Since we need to do voltage calculations that could result in frying the Raspberry PI if we are not careful enough. Additionally there is a strange time pressure on getting the microcontrollers working, electrically speaking, since the longer this takes the less time we have to actually code on the component that will have the image processing on it, which is a whole complicated matter in its own right. Thankfully the second microcontroller, the MSP430 is basically done with testing for the demo at the end of the semester already, with the code basically needing to go from two LEDs to fifteen which shouldn't be an issue.

A small problem that did come up was, procuring a CAD model for a Raspberry PI, ultra librarian did not have one despite the digikey listing showing that there should be one, which isn't surprising after going for CMOS sensors where we believe 95% of the ones that we looked at didn't have a CAD model despite advertising one. Regardless, in the case of the Raspberry PI, another site by the name of the component search engine, did have a Raspberry PI and all related CAD files. Additionally the download method was better than Ultra librarian in my professional opinion.

Another small issue that appeared during the testing of the MSP430 was that since a few of the LEDs aren't designated as GPIO pins, meaning they have a different intended use, there are a lot of them so we won't go into detail. However, because of this some of them are preset to being a 1 at the beginning of the program before they are masked down, for the rest of the process. This will overwrite their intended function so we can use them as a GPIO pin after this point since we wouldn't need a pin to do anything else, unless we accidentally used the reset pin, but that is unlikely, we would have chosen to avoid that one if we could. Anyway, this small issue doesn't damage any part of the design since the power runs for so short of a time, and the voltage is quite low, before the program returns to operating as normal, so this small problem ends up feeling like a small visual quirk, there is no harm in keeping this practice up as far as I'm aware. However there might be a work around however that would come from just using different pins or the boosterpack for the MSP430, the latter would add another $15 to the build cost, and the former is doable if tedious at best. It is worth remembering that this is a hardware issue, not software so there isn't some way we can code it out.

7.3.3 CMOS sensor Problems

The primary problem with the CMOS sensor is that they are way too expensive for our budget, as a group we have a way around needing one for our demos, but this has made it so we haven't been able to purchase one yet, this on top of finding a good CMOS sensor is hard anyway. We know that this part needs to work out of the box so we are going to eventually have to buy an expensive one, especially since we need one with a built-in

USB cable. This is because after a lot of research into how the CMOS sensor transmits the data, we went from planning a simple pin to pin connection between a simple CMOS sensor and a simple microcontroller, to requiring an on board computer like a Raspberry PI and a CMOS sensor that meet all the requirements with a USB plug-in. While we understand how a single instance of a transmission works we would probably not be able to make code efficient enough to even properly store the information in a coherent way without outside help. In short, in a single transmission a multi-bit integer is sent so a 1 or a 0 to each pin, and while we could store each in an array, and the overall total in an array of arrays, each input is very unique. Also the multi-bit integer refers to how some CMOS sensor can have either an eight bit or twelve bit integer, the difference between the two doesn't matter. The issue is after we somehow sync the microcontroller to the speed at which the CMOS sensor is transmitting, storing them shouldn't be the biggest concern, the problem would be the computation afterwards, we aren't working with effectively RGB values yet, each pixel is transmitting one at a time and each is looking for a certain thing. This creates the issue with once we have one stored we don't know where in the loop that one pixel is. Even if that issue can be resolved we then would need to recompile that back into a picture that we can reasonably work with, which once that is somehow done we would end at the same step a CMOS sensor with a USB connection to a on board computer starts at. Making it so the first major issue of the CMOS sensor can be resolved and for the good of this project and our sanity will be resolved by getting the better, more expensive parts.

For an additional side note, many of the most expensive CMOS sensors are basically a camera, with a lens, which is conflicting with the core design philosophy of the Lensless Digital Holographic Microscope. While the core CMOS sensor on paper is able to do what we need it to do, it would require additional work to remove a feature that we effectively paid for to get a CMOS sensor. Which could prove to be risky and possible damage the most expensive part of the device. Another small issue that came up during market research on CMOS sensors, was the pixel pitch, it initially sounded like it was a large issue that could severely impact the quality of the final results, however for most of the CMOS sensors that we have already picked to be candidate parts did not even have a pixel pitch, and those that did would always have it equal to the pixel size. Leading to a stressful point where we thought that we might end up in a situation where we just have to accept a pixel pitch that impacts the quality. However, two revelations happened, it turns out that the market standard for pixel pitch to pixel size ratio has always been one so we should never have a problem since the rest of the industry is working with the same restraints that we are, also in the paper by Ozcan and Wu they detailed a method that makes it so we don't have to worry about the pixel pitch called Pixel Super Resolution (3).

7.3.4 Software and Coding Problems

Some of the software and coding problems of the Lensless Digital Holographic Microscope project cycle were trying to get the CMOS sensor to capture images through the raspberry pi terminal. It took about two to three weeks to get this issue out of the way and it took us buying a new camera module to see that the first camera module we had purchased had been defective which was a cost that hurt us economically. Other constraints were the calculation of the necessary values that needed to be input into the matlab code so the images that the CMOS captured would be reconstructed and then placed into the Multi-Frame Pixel Super Resolution program. The only other time problem was time as the computer programmer needed to understand the photonic methods and how it was planning to be implemented so that the program would work as the photonic engineers wanted.

8.0 Administrative Content

This section of the document will be to show how well the team can manage their time and budget on this project. The time and due dates for each milestone will be spread from the Initial Document to the Final Presentation time. The budget will be decided by our sponsor and will be listed to show each expense that makes up the Solar bike.

8.1 Schedule

This section includes a table outlining the project milestones and schedule for the Lensless Digital Holographic Microscope project ranging from August 2022 to May 2023 the entirety of the Senior design capstone project.

Table Project Senior Design I Milestones and Schedule

Senior Design I	**Who is completing the task**	**Start Date**	**Due Date**	**Status**
Understanding the scope of the project	Group 7	08/22/22	08/26/22	Completed
Role Assignment	Group 7	08/25/22	08/25/22	Completed
Identify parts	Group 7	09/02/22	09/03/22	Completed
Project Reports				
Divide and Conquer	Group 7	08/29/22	09/16/22	Completed
Updated Divide and Conquer document (D&C V2)	Group 7	09/17/22	10/07/22	Completed
First draft	Group 7	09/26/22	11/4/22	Completed
Final draft	Group 7	11/4/22	11/18/22	Completed
Final document	Group 7	11/18/22	12/0/22	Completed
Optical Demos				
Midterm Demo	Parker and Nico	10/1/22	10/11/22	Completed
Creation of Optical Device for Demo	Group 7	10/1/22	10/11/22	Completed
Demo #2	Parker and Nico	10/12/22	11/22/22	Completed

Table Project Senior Design II Milestones and Schedule Continued

Project Life Cycle: Reseach, Documentation, and Design	Who is completing the task	Start Date	Due Date	Status
Microcontroller	Julian and Nick	09/02/22	12/06/22	Completed
LED	Parker	09/02/22	12/06/22	Completed
LED structure	Parker	09/02/22	12/06/22	Completed
LED manufacturing	Parker	09/02/22	12/06/22	Completed
Fiber coupling	Parker and Nico	09/02/22	12/06/22	Completed
Hardware design	Parker, Nico, and Nick	09/02/22	12/06/22	Completed
3D printing - of LED Fiber holder	Nico	09/02/22	12/06/22	Completed
Fiber Design	Parker and Nico	09/02/22	12/06/22	Completed
Fiber holder (5 lines per cylinder)	Nico	09/02/22	12/06/22	Completed
Distance between sample - fiber	Nico	09/02/22	12/06/22	Completed
Distance between sample - CMOS	Nico	09/02/22	12/06/22	Completed
Calculations behind holography	Julian	09/02/22	12/06/22	Completed
Theory behind reverse propagation algorithms	Julian	09/02/22	12/06/22	Completed
Circuitry	Nick	09/02/22	12/06/22	Completed
CMOS specifications	Nick	09/02/22	12/06/22	Completed
Software Tool	Julian	09/02/22	12/06/22	Completed
Schematics	Nick	09/02/22	12/06/22	Completed
Electrical supply	Nick	09/02/22	12/06/22	Completed
Holography theory	Julian and Nick	09/02/22	12/06/22	Completed
Housing Unit	Group 7	09/02/22	12/06/22	Completed
Pixel size/pitch calculation	Nick	09/02/22	12/06/22	Completed
Software Design	Julian and Nick	09/02/22	12/06/22	Completed
Code used for project	Julian and Nick	09/02/22	12/06/22	Completed

For the table below many of the testing and dates are to be announced or determined due to the fact that the due dates are unknown at this time but are within the table to allow us to be organized and prepared for the material that is to come in Spring 2023.

Table Project Milestones and Schedule for Senior Design 2

Senior Design II				
Testing and Redesign	Group 7	01/20/23	04/10/2023	Completed
Finialize Prototype	Group 7	01/20/23	04/12/2023	Completed
Practice Presentation	Group 7	01/20/23	04/12/2023	Completed
Final Report	Group 7	01/20/23	04/25/2023	Completed
Final Presentation	Group 7	01/20/23	04/19/2023	Completed

8.2 Financial Considerations

As of the writing of this paper we still need to find a sponsor so we do not have a current budget to work with in order to build and make our project from a concept to a reality. The prices listed below in the table are estimated using the research and parts used for the first optical demonstration conducted. As time continues to move the price, part choice and quantity are subject to change as prototyping is being conducted and the project gets closer to being produced. The final cost of the product will be evenly split between all the group members and if a sponsor is found.

Table 8.2-1: Project Parts Cost

Component Quantity	Quantity	Total Cost
LEDs	25	TBD
Small amount of multimode fiber	TBD	TBD
Fiber Chucks	Approx. 8	3D Printed
Fiber Clamper	1	TBD
Fiber Cleaning Wipes and Alcohol	1	TBD
UV Resin	2	$19.99
UV Flashlight	1	$6.99
Total ($)		TBD

**Note: that for the table above we are using the financial considerations for this submission of the first optical demo made for as seen within the schedule in section 10.1. The final Financial table will be present within the final draft and paper submission as the

CMOS sensor selection is still in the research stage in order to not spend funds carelessly. **

Electrical side:

Since a large part of our time was spent on market analysis on these parts, we feel like we should separate the components that we know we need to buy at some point, at the value and quantity that we need at the time, and the components that we have already done a lot of research on. Therefore there is now a table 8.2-2 which displays the parts that we have selected to be in the final design, unless there is a major problem with one of these parts and their costs won't change. The expectation is the power supply which requires some additional testing which is why we priced it a bit higher than we expect to pay. Additionally it is worth mentioning that small costs such as resistors, wires, soldering material, and etc, will not be on any list since it is hard and useless to estimate the value of those components, however there should be a full bill of materials including those ready for the documentation next semester.

Table 8.2-2 Electrical Component Cost

Part Name	**Cost**
Raspberry PI 2	$35
MSP430F5519 - PCB	$8.00 w/Build
AR0234	$109.99
Power Supply	$43
Total	**$195.99**

9.0 Conclusion

By bringing together Electrical and Computer Engineering with Optics and Photonics the Lensless Digital Holographic Microscope is able to go from thought to reality. The Lensless Digital Holographic Microscope helps bring forth a challenge to the issues of current Digital Holographic Microscopes and make a product that is simple, cost-effective, and portable that can be constructed at home or mass manufactured with smartphone compatibility (Bluetooth or direct lens imaging). Through deep research and extensive testing into the different fiber arrangements, the different LEDs we use. The spacing distances we settle on, the mat behind the holograms, pixel super resolution, back-propagation, and all the computer software we put into it, we have a perfect blend of our programs and our strengths. We brought this technology to UCF and ensure we do a deep dive into each concept and come out stronger and smarter than before.

10.0 Appendix

10.1 Acronyms

RoHS = Restriction on the use of Hazardous Substances

LED = light emitting diode

CMOS = Complementary metal–oxide–semiconductor

VS Code = Visual Studio Code

10.2 Work Cited

1.
LED Radiation Angle diagram,
https://www.ledsupply.com/blog/how-does-a-5mm-led-work/graph/

2.
LED Radiation Angle Differences diagram,
https://www.physicsforums.com/threads/difference-between-frensel-reflection-and-total-internal-reflection.670362/

3. Research on related works
https://innovate.ee.ucla.edu/wp-content/uploads/2010/03/Lensless-Microscopy-Ozcan-Group-Methods-2017.pdf

4.
https://www.insaneimpact.com/pixel-pitch/

5.
Fiber Chuck Reference Part
https://www.thorlabs.com/thorproduct.cfm?partnumber=HFC007

6.
"5.3. Forward Propagation, Backward Propagation, and Computational Graphs." *Dive into Deep Learning*, https://d2l.ai/chapter_multilayer-perceptrons/backprop.html.

7.
Admin. "Image Processing 101 Chapter 2.3: Spatial Filters (Convolution)." *Dynamsoft*, 2 Aug. 2019,https://www.dynamsoft.com/blog/insights/image-processing/image-processing-101-spatial-filters-convolution/.
8.
"Angular Spectrum Approach." *Focus*,
https://www.egr.msu.edu/~fultras-web/background/asa.php.

9.
BluEntCAD. “Top Advantages of SolidWorks Software.” BluEntCAD, 20 Feb. 2022, https://www.bluentcad.com/blog/solidworks-software-advantages/.

10.
Defienne, Hugo, et al. “Pixel Super-Resolution with Spatially Entangled Photons.” *Nature Communications*, vol. 13, no. 1, 2022, https://doi.org/10.1038/s41467-022-31052-6.

11.
Dorado, Ignacio Garcia, director. *Handheld Multi-Frame Super-Resolution (TOG 2019 & SIGGRAPH 2019)*, YouTube, 3 June 2019, https://www.youtube.com/watch?v=iDn5HXMQNzE. Accessed 3 Nov. 2022.

12.
“Difference between C and C++.” *GeeksforGeeks*, 25 Oct. 2022, https://www.geeksforgeeks.org/difference-between-c-and-c/.

13.
“Digital In-Line Holography for Multiphase Flows.” *Sensing Technologies Laboratory*, https://sites.gatech.edu/chen-mazumdar/digital-holography/.

14.
Eldar, Yonina, director. *Phase Retrieval with Application to Optical Imaging*, IEEETV, 29 Nov. 2015, https://ieeetv.ieee.org/phase-retrieval-with-application-to-optical-imaging. Accessed 3 Nov. 2022.
15.
Elkaseer, A., Schneider, S., & Scholz, S. G. (2020). Experiment-based process modeling and optimization for high-quality and resource-efficient FFF 3D printing. *Applied Sciences*, *10*(8), 2899. https://doi.org/10.3390/app10082899

16.
Garcia-Sucerquia, Jorge, et al. “Digital in-Line Holographic Microscopy.” *Applied Optics*, vol. 45, no. 5, 2006, p. 836., https://doi.org/10.1364/ao.45.000836.
17.
Holography - University of Illinois Urbana-Champaign. http://light.ece.illinois.edu/ECE460/PDF/Holography.pdf.
18.
Insane Impact. “Pixel Pitch Defined and Why It Matters.” *Insane Impact*, 13 July 2022, https://www.insaneimpact.com/pixel-pitch/.
19.
Jaganathan, Kishore, et al. “Phase Retrieval: An Overview of Recent Developments.” ArXiv.org, 26 Oct. 2015, https://arxiv.org/abs/1510.07713.
20.

J. Garcia-Sucerquia, W. Xu, S. Jericho, P. Klages, M. Jericho, and H. Kreuzer, "Digital in-line holographic microscopy," Appl. Opt. 45, 836-850 (2006).

21.
joelacarpenter. (2021, September 30). Gerchberg-Saxton Algorithm (tutorial). YouTube. Retrieved December 5, 2022, from https://www.youtube.com/watch?v=momXpSbOqMQ

J. Luo, Zhenxiang, et al. “Pixel Super-Resolution for Lens-Free Holographic Microscopy Using Deep Learning Neural Networks.” Optics Express, vol. 27, no. 10, 2019, p. 13581., https://doi.org/10.1364/oe.27.013581.

22.
Microsoft. “Visual Studio Code - Code Editing. Redefined.” RSS, Microsoft, 3 Nov. 2021, https://code.visualstudio.com/.

23.
Nasrollahi, Kamal, and Thomas B. Moeslund. “Super-Resolution: A Comprehensive Survey.” Machine Vision and Applications, vol. 25, no. 6, 2014, pp. 1423–1468., https://doi.org/10.1007/s00138-014-0623-4.

24.
Rehm, Lars. “Video: Google's Super Resolution Algorithm Explained in Three Minutes.” *DPReview*, DPReview, 30 May 2019, https://www.dpreview.com/news/6126296505/video-google-s-super-resolution-algorithm-explained-in-three-minutes.

25.
Rivenson, Yair, et al. “Phase Recovery and Holographic Image Reconstruction Using Deep Learning in Neural Networks.” ArXiv.org, 10 May 2017, https://arxiv.org/abs/1705.04286.

26.
Saavedra, Gabriel, et al. “Digital Back-Propagation for Nonlinearity Mitigation in Distributed Raman Amplified Links.” Optics Express, vol. 25, no. 5, 2017, p. 5431., https://doi.org/10.1364/oe.25.005431.

27.
Seth, Neha. “Backward Propagation: Backward Propagation Working in Neural Network.” *Analytics Vidhya*, 8 June 2021, https://www.analyticsvidhya.com/blog/2021/06/how-does-backward-propagation-work-in-neural-networks/.

28.
Velaquez, Daniel, and J. Garcia-Sucerquia. “Multi-Wavelength Digital in-Line Holographic Microscopy.” Biomedical Optics and 3-D Imaging, 2012, https://doi.org/10.1364/dh.2012.dtu1c.4.

29.
“What Is Eclipse Ide?: Importance and Features of Eclipse IDE.” EDUCBA, 12 July 2021, https://www.educba.com/what-is-eclipse-ide/.

30.
“What Is MATLAB?” *Mathworks*, https://www.mathworks.com/discovery/what-is-matlab.html.

31.
Xu, Wenbo, et al. “Digital in-Line Holography for Biological Applications.” *Proceedings of the National Academy of Sciences*, vol. 98, no. 20, 2001, pp. 11301–11305., https://doi.org/10.1073/pnas.191361398.

32.
https://www.britannica.com/technology/microscope/Confocal-microscopes

33.
https://www.thermofisher.com/blog/atomic-resolution/seeing-with-electrons-the-anatomy-of-an-electron-microscope/

34.