Nanophotonics for Ultrafast Optical Modulation, Ocean, and Energy Applications

by

Md Zubair Ebne Rafique

A Dissertation Presented in Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy

Approved August 2022 by the
Graduate Supervisory Committee:

Yu Yao, Chair
Joseph Palais
Yong-Hang Zhang
Maxim Sukharev

ARIZONA STATE UNIVERSITY

December 2022

ABSTRACT

Nanophotonics studies the interaction of light with nanoscale devices and nanostructures. This thesis focuses on developing nanoscale devices for optical modulation (saturable absorber and all-optical modulator) and investigating light scattering from nanoparticles for underwater navigation and energy sector application.

Saturable absorbers and all-optical modulators are essential to generate ultrashort high-power laser pulses and high-speed communications. Graphene-based devices are broadband, ultrafast, and compatible with different substrates and fibers. Nevertheless, the required fluence to saturate or modulate the optical signal with graphene is still high to realize low-threshold, compact broadband devices, which are essential for many applications. This dissertation emphasizes that the strong light-matter interaction in graphene-plasmonic hybrid metasurface greatly enhances monolayer graphene's saturable absorption and optical signal modulation effect while maintaining graphene's ultrafast carrier dynamics. Furthermore, based on this concept, simulation models and experimental demonstrations are presented in this dissertation to demonstrate both subwavelength (~λ/5 in near-infrared and ~λ/10 in mid-infrared) thick graphene-based saturable absorber (with record-low saturation fluence (~0.1μJ/cm2), and ultrashort recovery time (~60fs) at near-infrared wavelengths) and all-optical modulators (with 40% reflection modulation at 6.5μm with ~55μJ/cm^2 pump fluence and ultrafast relaxation time of ~1ps at 1.56μm with less than 8μJ/cm^2 pump fluence).

Underwater navigation is essential for various underwater vehicles. However, there is no reliable method for underwater navigation. This dissertation presents a numerical simulation model and algorithm for navigation based on underwater polarization mapping

data. With the methods developed, for clear water in the swimming pool, it is possible to achieve a sun position error of 0.35˚ azimuth and 0.03˚ zenith angle, and the corresponding location prediction error is ~23Km. For turbid lake water, a location determination error of ~100Km is achieved. Furthermore, maintenance of heliostat mirrors and receiver tubes is essential for properly operating concentrated solar power (CSP) plants. This dissertation demonstrates a fast and field deployable inspection method to measure the heliostat mirror soiling levels and receiver tube defect detection based on polarization images. Under sunny and clear sky conditions, accurate reflection efficiency (error ~1%) measurement for mirrors with different soiling levels is achieved, and detection of receiver tube defects is demonstrated.

DEDICATION

To my family, who never stopped believing in me!

ACKNOWLEDGEMENTS

First and foremost, I am thankful to almighty Allah for everything in my life. I am thankful to my beloved parents, who went through many hardships to bring me to this position in life. I could not have achieved anything if not for their unconditional support, love, and belief in me.

I am forever in debt to my supervisor Professor Yu Yao for her graceful guidance throughout my Ph.D. life. I humbly extend my gratitude toward Professor Chao Wang, Professor David Ferry, Dr. Ali Basiri, Dr. Jing Bai, Jiawei Zuo, Dr. Xiahui Chen, Hossain Resalat, Ahmed Hasssan, Mo Tian, and all my current and former lab mates for their support, discussions, and collaboration throughout my Ph.D. I am honored by the continuous support and comments from my committee members Professor Joseph Palasi, Professor Yong-Hang Zhang, and Professor Maxim Sukharev.

I am grateful to my wife, Tajrin Akter, for her love and support throughout the years of my Ph.D.; without her, I would be lost in this world. Furthermore, I would like to thank my Father-in-law (late) and Mother-in-law, who never forgot me in their prayers and always treated me like their own son. Finally, I want to thank all my family members back in Bangladesh who anxiously prayed for me so that I could be the first person in my family to achieve a Ph.D. degree.

TABLE OF CONTENTS

Page

CHAPTER Page

CHAPTER Page

CHAPTER Page

CHAPTER Page

APPENDIX Page

LIST OF TABLES

Table Page

LIST OF FIGURES

Figure Page

Figure Page

Figure Page

Figure Page

Figure Page

Figure Page

Figure Page

CHAPTER 1

INTRODUCTION

Nanophotonics is the study of light behavior after interacting with nanoscale devices and nanoparticles. Nanoscale devices include different types of subwavelength optoelectronic devices such as optical modulators, photodetectors, light polarization detectors, LASER, etc. Moreover, nanoparticles can change light properties after interacting with them, which involves different types of light scattering such as Rayleigh scattering, Mie scattering, Raman scattering etc. During my Ph.D., I have developed nanoscale devices for optical modulation (graphene and metasurface-based saturable absorber in near-infrared and all-optical modulator in mid-infrared wavelengths) and investigated changes in visible light polarization after it is scattered from nanoparticles for application in underwater navigation and concentrated solar power (CSP) plant's maintenance. In this chapter, I have tried to introduce all these topics briefly.

1.1 Graphene and Hybrid Metasurface

Graphene is a carbon-based 2D material, often called a wonder material. It has a single layer of carbon atoms arranged in a hexagonal honeycomb lattice[1–4]. Graphene has a linear dispersion relation (energy vs. wavevector or E-k) and a zero-band gap at the Dirac point or K-point[5]. It is one of the unique characteristics of graphene. This property allows the subatomic particles in graphene to act and behave as massless Dirac fermions (a type of quasi-particles)[6]. The consequence of this is tremendous, as it results in very high mobility of the carriers due to less back scattering; it introduces a constant optical conductivity and

constant optical absorption in graphene[6–8]. The zero band gap in graphene can be changed if we have pristine or doped bilayer graphene[2]. It has extraordinary electrical, optical, and mechanical properties. It is 100-300 times stronger than steel and is a very good heat conductor[4]. Graphene has a broadband response to optical signals and can be used to scale devices to work in different wavelength ranges[3,9,10]. Also, the carrier dynamics of graphene are ultrafast and can relax any excided carriers within 10fs to 1ps time scale[11–14]. The properties of graphene can be changed by applying an electric field or optical signals[7,10,15,16]. These properties allow graphene to be used in various applications in many different fields, namely in electronics (Graphene FETs[17–20]), optics (modulator[1,21–24], sensor[25–28], photodetectors[29–33], ultrafast switches[24] etc.), mechanical (graphene-enhanced composite materials[34,35]), energy storage, thermal engineering, and many more applications[36].

Metamaterials are artificially engineered materials that have properties that cannot be found in naturally occurring materials. To obtain these features, it is required to stack a lot of layers of materials or pattern the bulk material in a certain way. This can complicate the fabrication process and also makes the device bulky. Metasurface, on the other hand, is the subwavelength (i.e., ultrathin) version of metamaterial, which produces enhanced properties such as high nonlinearity, enhanced light-matter interaction, wavelength, direction-dependent refractive index, and so on. Therefore, metasurfaces are easier to fabricate and control the process better than metamaterial fabrication. Metasurfaces are mainly periodic arrays of thin layers of materials such as noble materials, semiconductors, ferromagnetic materials etc. Changing the size, gap, period, and thickness of these materials makes it possible to control the properties of metasurfaces. Metasurfaces can be

separated into two main categories: passive and active metasurfaces. Passive metasurfaces are most common, and the properties of this type of metasurfaces cannot be changed after its fabrication, i.e., the properties are fixed. In contrast, active metasurfaces are those whose properties can be changed even after fabrication by applying an electric field or optical signals. Plasmonic metasurfaces such as those made of gold nanobars are very common but once fabricated, the properties of these metasurfaces are fixed and work as a passive metasurface. Hybrid metasurfaces are mainly a combination of different materials or metasurfaces that can essentially enhance or change the properties of another or both. One possible combination is graphene and plasmonic metasurface (nanobar). We know that it is possible to change the properties of graphene by applying an electric field or optical signal. So, the properties of hybrid metasurfaces combining graphene and plasmonic metasurface can also change by applying electric field[37] and optical signals[38].

1.2 Saturable Absorption

Saturable absorber refers to a material or device that, if it gets hit by enough photons, the stimulated emission and absorption process inside the material becomes equal. In this situation, we can say that the material or the device is saturated, and seemingly no photons are being or will be absorbed anymore at this state (Figure 1). So, it is transparent to the incoming light after it gets saturated. Here, we only consider the passive saturable absorber where the output pulse gets narrower. It is also possible to actively pump a saturable absorber and make it an amplifier (Figure 1). However, we will only talk about the saturable absorber here.

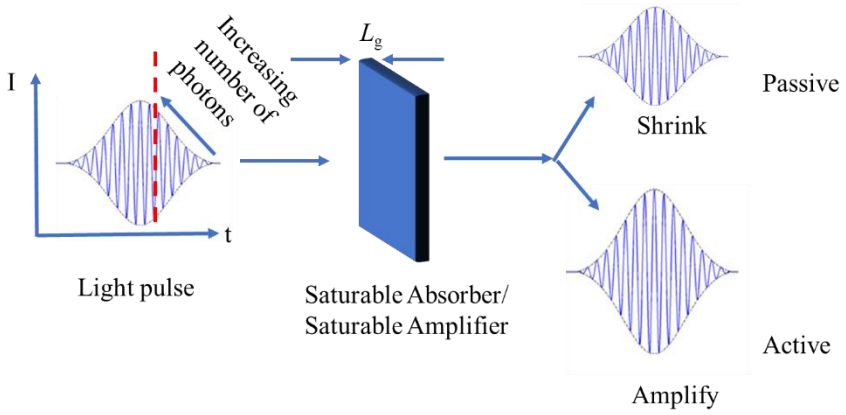


Figure 1: Saturable Absorber Working Principle. Light Pulse Comes in and Interacts With the Material. As the Number of Photons Interacting With the Material Increases, the Material Starts to Saturate, and After It Saturates, the Rest of the Pulse Goes out as If the Material was Transparent.

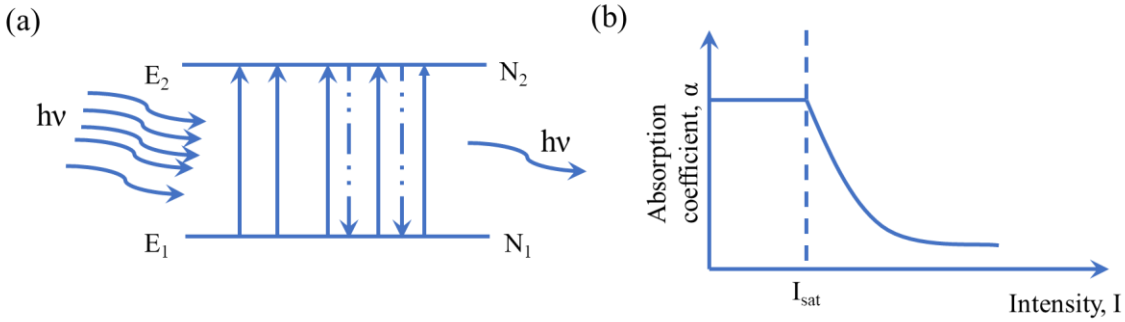


Figure 2: Two Energy State Representation of Saturable Absorber. (a) Incident Light Pulse Having a High Number of Photons Will Saturate the Absorber (b) the Absorption Coefficient of the Material/Device Decreases after the Light Pulse Reaches to Saturation Intensity.

Let us consider a material having two energy states (E_1 and E_2), and the energy difference is hν (E_1-E_2) (Figure 2(b)). Furthermore, initially, the number of carriers in E_1 and E_2 is N_{10} and N_{20}. If a light pulse with the energy of hν is incident on the material, then the population

in each energy state will change. Moreover, if the light pulse has a very high intensity and the number of photons is such that the material reaches the point where the number of absorption and stimulated emission has become equal, or the number of carriers in E_1 has become equal to the number of carriers in E_2. Then, it will seem that material has been saturated, and the net absorption is zero. This saturation intensity of the light pulse is called saturation fluence of the device, I_{sat}. After passing this saturation intensity, the absorption coefficient of the material or device will drop (Figure 2(b)).

The main characteristics of this type of material or device that we need to consider are the saturation fluence (energy per unit area needed to make the material/device saturated), speed (how fast it can be saturated), and also the relaxation time (the time it takes to regain the absorbing characteristics of the material or device after getting saturated).

Application of this type of device is to do passive mode locking and getting high power (currently, femto-second (fs)) lasers output. There is other type of applications of saturable absorber, such as pulse width measurement technique, controlling the pulse width of laser, optical signal processing etc.

For mode-locked lasers, locking different modes inside the laser medium creates very high-intensity short pulses. Usually, inside the laser medium, there is only one fundamental mode. If we introduce an optical modulator inside the laser cavity, with each pass of the laser pulses (fundamental mode) through the modulator, it distributes the power to the side modes, and eventually, all the modes supported by the gain medium are activated, and all of them are locked together to produce the ultra-short pulse. The pulse width is related to the number of modes locked. So, the higher the number of locked modes, the shorter the

pulse (eq. 1). Total electric field, assuming all the modes have the same amplitude and have a certain phase relation,

$$e(t) = E_o e^{j\omega_o t} \frac{\mathrm{Sin}\frac{N\Delta\omega t}{2}}{\sin\frac{\Delta\omega t}{2}} \quad (1)$$

Where E_o is the amplitude, N is the number of modes, ω_o is the central mode frequency, and $\Delta\omega$ is the free spectral range of the laser medium.

Now, the modulator that we introduced inside the mode-locked laser cavity can be controlled in two different ways. One, from outside with an electrical signal (active), and two, it can be self-modulating (free running or passive). Depending on that, mode-locked lasers are of two kinds, (a) Active mode-locked lasers and (b) passive mode-locked lasers. For active, the modulator is controlled electrically from the outside. And for passive, the modulator works by itself or is self-modulated by the laser pulse inside the cavity. So, as saturable absorbers are self-modulating passive modulator, it is used in passive mode-locked lasers. In passive mode locking, the mode-locking process starts due to one random noise or spontaneous emission large enough to saturate the saturable absorber and builds up from there. For pulse reshaping, the input and output relationship of the pulse is given in[39],

$$I_2(t) = I_1(t) \frac{G_0 e^{u_1(t)}}{1 + G_0[e^{u_1(t)} - 1]} \quad (2)$$

Where the input pulse $I_1(t)$ is given by, $I_1(t) = \frac{\omega_0}{T}\sin^2(\frac{\pi t}{2T})$,u(t) is given by, $u(z,t) = \frac{\omega(z,t)}{\omega_s}$ and G_o is the gain of the laser medium. The low-intensity part of the pulse gets absorbed, and the high-intensity part of the pulse are transmitted through the saturable absorber. So, it is possible to change the laser's pulse shape with a saturable absorber. That means if we

have a saturable absorber and hit it with a pulsed laser with certain pulse width, the out coming laser pulse from the SA (saturable absorber) will have smaller pulse width. In the later sections, we will see some implementation of these characteristics.

Graphene, as we have discussed previously, is a material that, among other properties, shows nonlinear response when absorbing light pulses with a large number of photons[40–42]. But absorption in graphene is very small (~2.3%). So, to increase the light-matter interaction and increase the absorption of light in graphene, a hybrid metasurface is designed, consisting of plasmonic nano-antenna and graphene. The plasmonic metasurface is designed in such a way that it can efficiently confine the light in the nano gap and increase the light-matter interaction with graphene. So, saturation can be obtained with lower pump intensity or pump fluence. In this study, ultrafast nonlinear absorption characteristics of the graphene-plasmonic hybrid metasurface are demonstrated through saturable absorption measurement, pump-probe measurement to measure the recovery time of the device, and the capability of pulse width narrowing is also demonstrated. We have achieved ~0.1uJ/cm2 saturation fluence from the experiments, which is two to three orders of magnitude smaller than graphene saturable absorbers reported in the literature. Furthermore, the recovery time achieved by this device is around 60fs obtained by degenerate pump-probe measurements.

1.3 Reverse Saturable Absorption

A reverse saturable absorber (RSA) can reduce the intensity of incoming light by increasing its absorption at higher intensities. Therefore, this nonlinear optical phenomenon can be used to realize optical limiters, switches, and pulse shapers. So far, different types of RSA

have been demonstrated with solution-based polymers [43], organic dyes [44], nanoparticles [45], functionalized or solution dispersed graphene [46], and so on. However, they are bulky, require unique preparation methods, and exhibit a large optical limiting threshold and long response time (>ns), which are unsuitable if we want to combine them in a compact photonic circuit to limit, modulate or shape ultrafast signal. On the other hand, pure graphene has an ultrafast response time, which can be utilized to realize fast devices, but without altering or functionalization, it does not work as a reverse saturable absorber, and functionalization tends to slow down the response time of graphene [3]. In this dissertation, I have shown that combining graphene monolayer with plasmonic metasurface makes it possible to achieve reverse saturable absorption with the help of hot carriers generated by the plasmonic metasurface while maintaining ultrafast response time. Using the reverse saturable absorption property of our subwavelength hybrid planar metasurface design, it is possible to develop ultrafast low-threshold optical limiters, switch or pulse shapers for ultrafast-photonic circuits where fast response time and the low-optical limiting threshold are essential.

1.4 All-Optical Modulation

Modulating signals, either electrical or optical, is essential for data processing. Nowadays, optical modulators[47] are getting much attention because of their capability of achieving high speed and low power requirements for high-speed optical signal processing[48] and communication[49], optical interconnects[3], ultrafast pulse generation[50], quantum information, ultrafast spectroscopy[51] etc. There are many different methods to modulate optical signals, such as electro-optic modulators, acousto-optic modulators etc. For electro-optic modulators, the optical signal is modulated utilizing traditional techniques such as

refractive index modulation of the material by applying an electric field (Pockels effect). However, this technique depends on an electrical circuit that usually determines the speed of the modulation of the optical signal. So, the optical signal modulation speed is limited by the capacitive effect of the electrical circuit (time constant). Furthermore, another novel optical modulation technique type is known as the all-optical modulator. For all-optical modulators, the optical signal is modulated by another optical signal, not electrical. So, the latency imposed by the speed of the electrical circuit used to control the electro-optic modulator is not present here. Consequently, it is possible to operate all-optical modulators in ultrafast manner in THz or higher speed instead of the GHz or less speed achieved by electrically controlled optical modulators.

Many all-optical modulators are reported in the literature in near-infrared (NIR) wavelength ranges[22,24,49,50,52–67]. However, due to insufficient nonlinearity in the available materials and lack of proper methods, there are a few demonstrations of all-optical modulation in mid-infrared wavelengths larger than 2μm, all of which are less than 5μm. So, there is no demonstration of all-optical modulation in wavelengths larger than 6μm wavelengths. Most of the all-optical modulators demonstrated are based on 2D materials such as graphene[23,65–67], two-photon absorption[54,59], silicon nanoantenna[52], Tamm-plasmon resonance[68], plasmonic semiconductor nanocrystals[50], 2D array of holes in Si membrane[55] etc. It is challenging to obtain ultra-compact, low pump fluence, and ultrafast all-optical modulation in MIR due to inherent optical absorption and small modulation of conventional materials.

Graphene is a novel 2D material with broadband nonlinear optical behavior and ultrafast carrier dynamics, and it is also compatible with many different types of substrates, including fiber optics. However, the demonstrated all-optical modulators utilizing only graphene in the literature require a large optical pump fluence (>1mJ/cm^2)[49,65–67] primarily due to low optical abruption of light (~2.3%) and ultrashort carrier life time. There are different methods to increase the optical absorption, such as increasing the optical interaction length by combining it with waveguide structure[66], increasing light-graphene interaction utilizing metasurface (plasmonic and semiconductor)[22,69], putting graphene inside a cavity[70] etc.

In this part of my dissertation, it is demonstrated that with the help of a graphene plasmonic hybrid metasurface design, it is possible to achieve high all-optical modulation depth and ultrafast speed in wavelengths larger than 6μm[38] with very low pump fluence of about 75μJ/cm^2. The hybrid metasurface consists of a novel plasmonic metasurface of closely coupled pi (π) shaped antenna and monolayer graphene. The plasmonic metasurface creates nanoscale hot spots with high near-field enhancements when light is incident on the device. The hybrid metasurface is a double resonant device having one resonance in the MIR and the second resonance in NIR wavelengths. Due to this double resonance behavior of this device, it is possible to enhance the near field for both pump (NIR) and probe (MIR) wavelengths by a few orders of magnitude compared to the incident light electric field. Due to this effect, the pump photon absorption is highly boosted, and we can achieve larger modulation in the MIR wavelengths (probe beam) with very low pump fluence in the NIR wavelengths. NIR all-optical modulation is also demonstrated with this device design with a very low pump fluence of 7.9μJ/cm^2. For optical signal modulation in MIR and NIR

wavelengths, the required pump fluence with our device is more than two orders of magnitude lower than all-optical modulators reported in the literature. Ultrafast pump-probe measurements in the NIR wavelengths (780 pumps and 1560 probe) illustrated that the device conserves its ultrafast behavior of the graphene carrier dynamics, thus resulting in an ultrafast response of our device in the order of 1-2ps range. In this project, I helped build the optical setup, participated in the fabrication of the device, did measurements and data analysis.

1.5 Light Scattering by Nanoparticles

Many natural phenomena such as the blue color of the sky, the rainbow, the red color in the sky during sunsets, the white color of cotton clouds or the dark color of clouds that will make rain, the blue color of the sea, and so on are happening because of light scattering. Even the law of reflection at an interface (Snell's law, Fresnel's equations etc.) is a simplification of the scattering phenomenon[71]. So, we can derive the equations of reflection from Maxwell's equations considering the boundary conditions as well as from the theory of light scattering[71]. For large objects, many times larger than the wavelength, we tend to use laws of reflection instead of scattering because they are easier to calculate and can show us the direction of light and its properties after reflection quite easily. However, for small particles, we have to use the theory of scattering to understand the changes in the properties of light, such as polarization, because of the comparable size of the wavelength and particles.

Two different processes happen when light is incident on a small particle or nanoparticle. One, light energy is absorbed in the particle (Excitation); two, re-distribution of the absorbed energy we see as scattered light (Re-radiation). To go into details, all particles

are composed of electric charges such as electrons and protons. When light, which is just the oscillation of electric and magnetic fields, falls on the particle, the charges start to oscillate at the same frequency as the electric field of light. These oscillating electric charges radiate electromagnetic energy in all directions with the same or different intensities. This is called the secondary radiation or re-radiation of energy or radiation scattered by the particle. Also, some parts of the incident light energy can be converted to other forms of energy, such as heat, by the oscillating electric charges; this is called absorption. So, absorption and scattering go hand in hand for lights interacting with particles.

There are different types of scattering. Two major categories are elastic scattering and inelastic scattering. For elastic scattering, the frequency or wavelength of light is unchanged after scattering from the nanoparticles, such as Rayleigh scattering, Mie scattering etc. For inelastic scattering, the frequency or wavelength of light changes after scattering, such as Raman scattering, stokes scattering etc.

When we consider a particle interacting with light electromagnetic (EM) waves, we can divide the particle into several different regions with different oscillating dipole moments. These dipole oscillators produce secondary radiation or scattered light in all directions. The intensity of the re-radiated energy from the particle, in a certain direction, depends on all the electric fields radiated by all those dipoles in that direction and, most importantly, their phase differences. For Rayleigh scattering, the redistribution of energy after interacting with the particles is quite uniform, meaning the scattered light intensity is distributed equally in all directions. This happens because the particles involved in Rayleigh scattering are very small spherical particles compared to the wavelength of light, and the phase

differences between all the re-radiated EM fields are pretty small. However, for Mie scattering, the re-radiated filed intensity or scattered light intensity is a function of position because Mie scattering happens for larger spherical particles, larger or comparable to the wavelength of light. As the particles become larger, the number of dipole moments also becomes larger, and the phase relationship between all re-radiated EM fields becomes quite different. So, we see some ups and downs in the re-radiated intensity for Mie scattering in large particles. Usually, for larger particles, Mie scattering, light is mostly scattered in the forward direction. The phase relationship between the re-radiated EM fields also depends on the shape of the particles. However, here in my dissertation, I am considering only spherical particles involved in Rayleigh and Mie scattering of light.

So, Rayleigh scattering happens for small particles smaller than the wavelength of light (in my case, <100nm), and this is an approximation of Mie scattering theory which is the more general consideration. On the other hand, Mie scattering happens for particles comparable to or larger than the wavelength of light. The scattering cross section is highly dependent on the wavelength of light due to different phase relationships for different wavelengths in different scattering angles. And the light polarization is also changed due to scattering, which is considered extensively in this part of my dissertation with the help of Muller matrix. Muller matrices are like transfer matrices which can tell us how light will change its intensity and polarization in different scattering angles after scattering.

Mie scattering theory is the Mie solution to Maxwell's equations, which converges to the geometric optics limit for large particles. The scattering pattern depends highly on the particles size (radius, a), index of the particle (n_p), index of the surrounding medium (n_{medium}) and the wavelength (λ) of light.

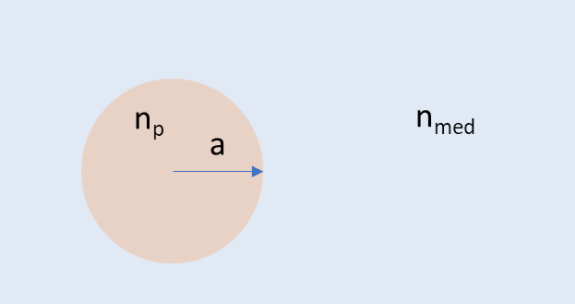


Figure 3: Representative Diagram Shown a Particle with Radius an Inside a Medium with Refractive Index n_{medium}.

Some very useful metrics to consider are provided below.

Size parameter,

$$x = \frac{2\pi a}{\lambda / n_{medium}} \tag{3}$$

Refractive index mismatch,

$$n_r = \frac{n_p}{n_{medium}} \tag{4}$$

We can consider that if x<<1, then we can expect to see Rayleigh scattering, and if x≈>1, then we can expect to see Mie scattering. Also, if we consider the refractive index mismatch for the same particle size and light, we can have different types of scattering for different surrounding mediums. For instance, in air ($n_{medium} = n_{air} = 1$), for 100nm particles, scattering will probably be Rayleigh scattering. But for the same 100nm particle when inside water ($n_{medium} = n_{water} = 1.33$) the most probable scattering is Mie scattering, just because of the changes in the surrounding medium.

In my dissertation, I have discussed three different projects for light scattering applications. These are underwater navigation, mirror soiling, and receiver tube defect detection. Underwater navigation is an application for underwater vehicles. The mirror soiling and

receiver tube defect detection are some applications related to concentrated solar power (CSP) plant maintenance.

1.6 Geolocation Detection Using Polarization

Polarization of light, if appropriately manipulated, can be used as a tool that allows us access to different applications such as polarimetric imaging[72], data communication[73,74], optical image processing[75], quantum computing[76], and our topic of interest 'navigation'[77]. We live under an abundance of light that comes from the sun. Sunlight is unpolarized outside our atmosphere. Once it enters our atmosphere, it gets scattered by different scatters and becomes polarized[78]. Moreover, the polarization of this skylight introduced by scattering inside the atmosphere has a unique way of showing us the way! Literally speaking. This unique application is to navigate using the polarization of skylight[77]. Navigation with the help of skylight polarization has been used from ancient times by Vikings to find their way in their vast sea explorations[79–81]; insects use it to find their way back to their home[82,83] and so on.

As the light enters the earth's atmosphere, it goes through multiple scattering events and eventually reaches the earth's surface. All the light path goes through different scattering events; thus, the ultimate polarization state of the two light beams reaching the earth's surface is not the same. Similarly, countless photons are going through countless scattering events; ultimately, they all have some uniqueness compared to others. That means if we map the polarization at one point on earth, it should be a unique representation of the polarization states of the skylight at that position on earth. In other words, at each position on earth, we should have unique polarization mapping depending on the sun's position.

This is the main idea of our work which utilizes the skylight polarization mapping to determine the position on earth.

Inside the atmosphere, there are a lot of particles of different types[84], which introduces different polarization components to the scattered light. There are particles that are small compared to the wavelength of light; they are known as Rayleigh scatter, which introduces Rayleigh scattering[85]; particles that are larger than the wavelength of light and introduce Mie scattering[86], also there are non-spherical cloud particles, aerosol particles, and all of them contribute to the polarization state of the scattered light. From the literature, we know that clear skylight is mostly linearly polarized, and the circular polarized and unpolarized light is very small compared to the linear polarized part[87,88]. Also, we know that the particles smaller than the wavelength of light or Rayleigh scatters introduce most of the linear polarization that we see in skylight[85]. So, Rayleigh scattering of light is responsible for skylight polarization formation.

Now, to see the polarization distribution from underwater, we need to consider different contributions from different sources. Sunlight and skylight both go inside water through refraction. Due to different scatterers in the water, sunlight and skylight both can get scattered and change the polarization state of the incoming light. As water has refractive index (n_{water}=1.333) larger than air (n_{air}=1), overall probability of mie scattering increases. Also, many more scatterers are suspended in water than in the air. So, scattering inside water is mainly due to Mie scattering. In this part of the dissertation, I have demonstrated geolocation capability from under the water utilizing the underwater light polarization map.

1.7 Mirror Soiling Detection

Heliostat mirrors are used in concentrated solar power (CSP) plants to redirect and focus sunlight on receiver tubes to produce electricity. Due to wind, dust storms, and many other natural environmental effects, these mirrors can become soiled with dust particles. As a result, sequential cleaning is required to maintain the maximum reflectivity of the mirrors. We demonstrated a fast and field deployable inspection method to measure the heliostat mirror soiling levels based on polarization images in regular daylight settings. Under sunny and clear sky conditions, we have achieved accurate measurement of reflection efficiency (error ~1%) for mirrors with different soiling levels.

Mirror soiling could significantly reduce the optical reflection efficiency (below 90%) of heliostats. The heliostat mirror cleaning cost is ~25% of the total Operation and Maintenance cost of the Concentrated Solar Power (CSP) plant [89]. Currently, the soiling levels are usually determined by manual measurement using hand-held reflectometers, which are slow and only cover a small portion of the mirror surface. A fast and field-deployable method for detecting the soiling levels of heliostat mirrors is highly desirable for planning CSP field collector cleaning to maintain low optical loss with minimal cleaning cost. Moreover, it can also facilitate the study of mirror soiling patterns in CSP fields. Here we show that polarization images of heliostat mirrors, esp. the degree of linear polarization (DoLP) images, can be used under sunny and clear sky conditions to measure the mirror soiling levels. We performed outdoor tests and determined mirror soling levels up to reflection efficiencies of 98%, which is deemed challenging by conventional imaging-based methods [90,91] without needing high accuracy sensors [92]. The polarization

imaging method can be deployed in CSP collector fields for in-situ characterization of the heliostat mirror soiling levels with high throughput and low cost.

1.8 Receiver Tube Defect Detection

Receiver tubes[93–96] are used in concentrated solar power plant[97,98] to collect the heat generated by the sunlight. The sunlight is focused on the receiver tubes by heliostats or mirrors[98]. There are different types of receiver tubes, and the design can be different[95], but it is usually a hollow rod where the energy transfer material can flow. The receiver tubes usually contain molten salts that can store the heat or energy for a long time and can be used later when there is no sun shining in the sky (cloudy weather) or at night. The receiver tubes are usually painted with heat-resistant paints such as carbon black or pyromark[99,100]. They can withstand thousands of degrees Celsius temperatures. However, with time the paint on the receiver tubes can degrade, fall off, or it can be scratched due to storms or wind. Also, it can cause leakage of molten salt. All these defects in receiver tubes can cause a significant decrease in the efficiency of the power plant and cost a lot of money for repairs and maintenance. So, it would be beneficial if we could detect these defects earlier and take precautions.

People usually use thermal camera[101,102], different types of sensors and signal processing[103,104], visual inspection[104], and intensity imaging[104] to detect these defects. However, they are slow, depend on good lighting conditions, costly equipment and so on. We propose improving the defect detection and inspection technique of receiver tubes by utilizing light scattering by the nanoparticles on the receiver tubes and the polarization information of the scattered light. The nanoparticles on the receiver tubes scatter light, and

comparing the scattered light polarization from a perfect receiver tube with the receiver tube under inspection, and we can determine the defects in the receiver tube.

1.9 Coordination of Different Chapters in the Dissertation

In this dissertation, in chapters 2 and 3, I have talked about the design, fabrication, and characterizations of ultrafast low saturation fluence saturable absorber based on graphene plasmonic hybrid Metasurface. I have discussed in detail the structures I have used, the changes in characteristics with different structure parameters, numerical model to investigate properties of graphene under intense laser light excitation and measurements of different parameters such as saturation fluence and response time.

In chapter 4, I have talked about the all-optical modulation in mid-infrared wavelengths achieved by pumping graphene plasmonic hybrid Metasurface with near-infrared pump light and its characterization of different properties such as response time with the help of pump-probe measurements.

In chapter 5, I discussed our ongoing project about a mid-infrared saturable absorber utilizing the graphene plasmonic hybrid Metasurface and some preliminary results.

In chapter 6, I have discussed skylight polarization and navigation and underwater light polarization and navigation. I have also discussed in detail the need for a new simulation model to describe underwater light polarization for turbid waters.

In chapter 7, I discussed how we could utilize polarimetric imaging to determine soiling levels of heliostat mirrors in CSP plants. Detail discussion about the simulation model and detection process is included here. This is a proof-of-concept demonstration, so I have used our own sample that we made in our lab.

In chapter 8, I discussed the receiver tube polarization pattern and how we use light polarization to detect different defects in them. Understanding the polarization pattern and how it is obtained is also discussed with the help of a mathematical model developed here.

In chapter 9, I summarized all the discussions in this dissertation and provided an outlook for future research in each of these aspects.

CHAPTER 2

DESIGN AND ANALYSIS OF GRAPHENE PLASMONIC HYBRID METASURFACE SATURABLE ABSORBER (GPMSA) IN NIR WAVELENGTH

Use of nonlinear optical component- saturable absorber is ubiquitous to generate ultrashort high-power laser pulses with passive mode-locking. Graphene-based saturable absorbers are broadband, ultrafast, and compatible with different substrates and fibers. Nevertheless, the required saturation fluence is still high to generate low-threshold, compact, self-starting mode-locked lasers. This dissertation emphasizes that the strong light-matter interaction in graphene-plasmonic hybrid metasurface greatly enhances monolayer graphene's saturable absorption effect. Furthermore, based on this concept, a subwavelength (~λ/5) thick graphene-based saturable absorber with record-low saturation fluence (~0.1μJ/cm2) and ultrashort recovery time (~60fs) at infrared wavelengths is experimentally demonstrated.

2.1 INTRODUCTION

Saturable absorbers are nonlinear optical components with reduced optical absorption loss at high optical intensities. Saturable absorbers with low saturation fluence and ultrafast recovery time are highly desirable for low threshold, compact, and stable self-starting high power mode-locked lasers, efficient laser pulse shaping, and high-speed optical signal processing.

Different saturable absorbers are reported in literature utilizing different materials and approaches, such as graphene[40–42], transition metal dichalcogenide (TMDC)[105–108], carbon

nanotube (CNT)[109–111], topological insulator[112,113], waveguide structures[21,24,114], semiconductor metasurface[115], plasmonic metasurfaces[116], perovskites[117], black phosphorous[118,119], and so many others. However, we are mainly interested in graphene saturable absorbers because of graphene's broadband optical properties[7,8], nonlinear absorption[3,120], and ultrafast recovery time[11]. A comparison between saturable absorbers utilizing graphene and other materials is presented in Table 1 in terms of different performance parameters such as saturation fluence and recovery time. If we look at only saturation fluence of different approaches, they are comparable, but if we combine recovery time with saturation fluence, we can see that graphene saturable absorbers perform better than the other reported works. But the problem is that the weak absorption of graphene is only 2.3%, which is very low, and the coupling of light with free-standing graphene is very weak, which results in a large saturation fluence around mJ/cm2. There are different methods to reduce the saturation fluence; one of them is to incorporate waveguide structure to increase the light-matter interaction with graphene[24,114]. This method can reduce the saturation fluence to μJ/cm2 level with a comparable recovery time around 100fs. But the problem with this approach is that it increases the insertion loss of the device[24]. So, the goal of our project is to have low saturation fluence and ultrafast recovery time while maintaining low insertion loss of the device.

Table 1: Some Reported SAs in Literature

	Wavelength (nm)	Saturation fluence	Saturation intensity	Recovery time	Modulation Depth	Insertion Loss
SESAM[121]	1314	1.1 uJ/cm^2	-	-	4%	-10.2119dB
Monolayer Graphene[122]	750	5.3 mJ/cm^2	5300 MW/m^2	100fs	65.9%	4.9592 dB
Multilayer Graphene[41]	630	7.13 mJ/cm^2 (3 layer)	7100 MW/m^2 (3 layers)	0.21ps	~65% (3 layer)	4.34 dB
MoS_2 Nanoplatelets[105]	400, 800 and 1064	0.124 mJ/cm^2	136.1+/- 20GW/cm2, 279.6+/- 60 GW/cm2 and 8.7 MW/m^2	-	10% (N), 34% (N) and 4.6%	-, - and 0.34 dB
WS_2 film[107]	1560	1.1 J/cm^2	250 GW/m^2	-	1.20%	0.18 dB

Plasmonic-Metasurface[116]	1555	0.64 J/cm^2	-	-	60%	8.42 dB
Waveguide integrated with Graphene[24]	1550	0.17uJ/cm^2 (PS Laser) 0.04uJ/cm^2 (FS Laser)	-	~100fs	1.5dB	19 dB
Graphene-Bi2Te3 heterostructure[123]	1550	0.82-1.23 J/cm^2	72.9 to 109.5 GW/m^2	191-296fs	11.5-41.88% (depending on coverage of Bi_2Te_3)	2.5 to 5 dB
Few Layer $PtSe_2$ [106]	1064	-	0.346 GW/cm^2	-	26% (ΔT/T)	-
CMOS-compatible Gr-Si Hybrid WG[21]	1565	1.353 mJ/cm^2	-	1.65 ps	22.7% (ΔT/T)	-
Large-Diameter SWCNT[109]	2300	18 uJ/cm^2	-	220fs	30% (ΔT/T)	-
Carbon-nanotubes[110]	1500-1630	0.56 J/cm^2	-	~1ps	9.182%(ΔT)	-
Ionic Liquid Gated CNT[111]	1540	-	-	80 to 190fs (At different Applied voltages)	0.7 to 3.2 % (ΔT/T)	-

In this paper, we demonstrate an ultra-compact graphene-plasmonic hybrid metasurface saturable absorber with ultralow saturation fluence (~0.15 μJ/cm2) and ultrafast recovery time (<60fs) in the infrared wavelength. The subwavelength-thick metasurface device confines light efficiently in a diffraction-limited nanogap and increase the light-matter interaction[37] with graphene. As a result, absorption of light increases, and a sizeable instantaneous change is observed in graphene's conductivity due to an increase in photogenerated hot carriers in the conduction band (CB). As a result, the number of photoexcited carriers in CB saturates at saturation intensity, and the photoexcited carriers relax due to intraband carrier-carrier and carrier-optical phonon scattering. Furthermore, the photoexcited carriers' rapid rise and subsequent ultrafast decay introduce ultrafast temporal modulation of our device's reflection spectra, magnifying the saturable absorption (self-modulation) effect in the hybrid-metasurface. As a result, the saturation fluence of the device becomes lower. Besides, the ultrafast carrier dynamics of graphene help achieve ultrashort recovery time with our device. To the best of our knowledge, this

SA device has the lowest saturation fluence and the fastest recovery time demonstrated until now.

2.2 DESIGN CONCEPT

2.2.1 Plasmonic Metasurface Design

To increase the absorption in monolayer graphene, we are using a plasmonic metasurface composed of a gold (Au) nanobar antenna array underneath the monolayer graphene. To enhance the near-field inside the metasurface nanogaps, we have used a dielectric layer (Al_2O_3) and a back reflector (Al) to emulate the Fabri-Perot cavity effect (metasurface on top works as a partial reflector). So, when the light pulse falls on the device, multiple reflections in the dielectric layer introduce interference on top of the metasurface and satisfy the perfect absorption condition[37] to cancel out most of the reflection, which increases the absorption. The multiple reflections in the Fabri-Perot cavity can be modeled with the modified Fresnel equations presented below[37],

$$r_{12} = \frac{n_1 - n_2 - \sigma_M Z_o}{n_1 + n_2 + \sigma_M Z_o} \tag{5}$$

$$r_{21} = \frac{n_2 - n_1 - \sigma_M Z_o}{n_2 + n_1 + \sigma_M Z_o} \tag{6}$$

$$t_{21} = \frac{2n_1}{n_1 + n_2 + \sigma_M Z_o} \tag{7}$$

$$t_{21} = \frac{2n_2}{n_1 + n_2 + \sigma_M Z_o} \tag{8}$$

$$r_A = \frac{E_R}{E_i} = \frac{r_{12} + (r_{12}r_{21} - r_{12}r_{21})r_{23}e^{i2kd}}{1 - r_{21}r_{23}e^{i2kd}} \tag{9}$$

$$r_{12} + (1 + r_{12} + r_{21})r_{23}e^{i2kd} = 0 \tag{10}$$

Equations (5) - (8) are the modified Fresnel equations. Furthermore, equation (9) represents the overall reflection from the device. If we set equation (9) to zero, which means no reflection and all the light is absorbed, we can obtain equation (10). This equation is the perfect absorption condition.

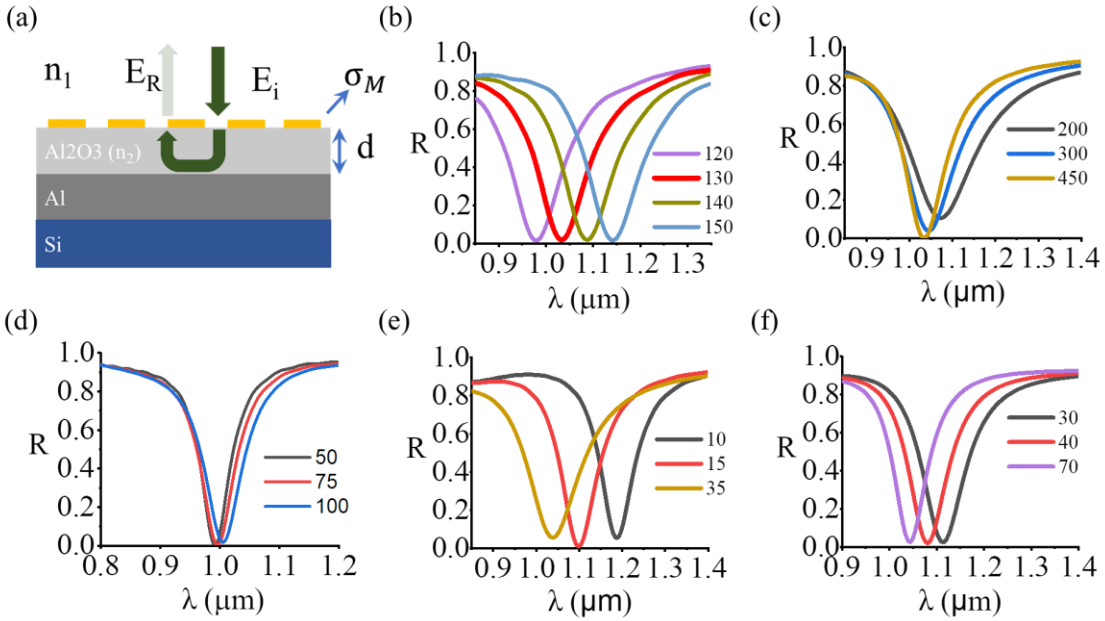


Figure 4: FDTD Simulation to Obtain the Best Graphene-Plasmonic Hybrid Metasurface (a) Plasmonic Metasurface Design with Fabry-Perot Cavity Mode. The Surface Conductivity of the Metasurface (σ_M) Can be Changed By Changing Different Structural Parameters. (b) Changing Au Antenna Length Changes the Resonance of the Device. Can be Effective in Find Resonance Around 1.035μm. (c) The Y-Period is Changed to Find a Period That Does Not Introduce the Grating Effect. (d) Au Metasurface Width Change. It Does not Affect the Device's Working Wavelength Much as only the Carrier Oscillation On the Short Axis is Changed, Whose Resonance Does Not Fall Around this Wavelength Range. (e) Al_2O_3 Thickness Change Can Change the Fabri-Perot Cavity Mode By

Changing the Phase Accumulation. Thus, It Can Be Used to Obtain Different Absorption At Working Wavelength. (f) The Gap Between Two Nanoantenna Can Change the Working Device Wavelength by Affecting the Near-Field Enhancement Between Them. All Parameters Shown in the Figure Are in Nanometer(nm).

The perfect absorption condition (eq. (9)) depends on the metasurface conductivity (σ_M), dielectric layer thickness (d), complex dispersive refractive index (n_2) of the dielectric layer, and wavelength of operation. We can modify the metasurface conductivity by changing the Au- nanoantenna length, width, thickness, and gap between. Adding graphene on top will also change the metasurface conductivity depending on the quality of graphene, chemical doping, number of layers and so on. Furthermore, we can change the dielectric layer thickness to obtain perfect absorption in the desired wavelength, which in this case is the wavelength of our laser in the lab at 1.035μm.

We do all the FDTD simulations with commercially available Lumerical Inc. FDTD software. Here, we try to put together the best possible structure with large absorption at the resonance and high nearfield enhancement in the gap. Figure 4(b)-(f) highlights this optimization process with some selected plots. Figure *5*(b) presents the near-field enhancement inside the gap. It shows a large enhancement of over 1000 at the laser wavelength. Also, at the laser wavelength, the absorption of the final device is around 99% (shown in Figure 4). Therefore, the best possible structure parameters range that we find is as follows: length of Au antenna, L = 130nm; width of Au antenna, w = 75nm to 100nm; the gap between nanogap, G=30 to 70nm; thickness of Au, t_{AU}= 40nm; Al_2O_3 dielectric layer thickness, t_{ox}= 20 to 30nm.

2.2.2 Hybrid Metasurface Design

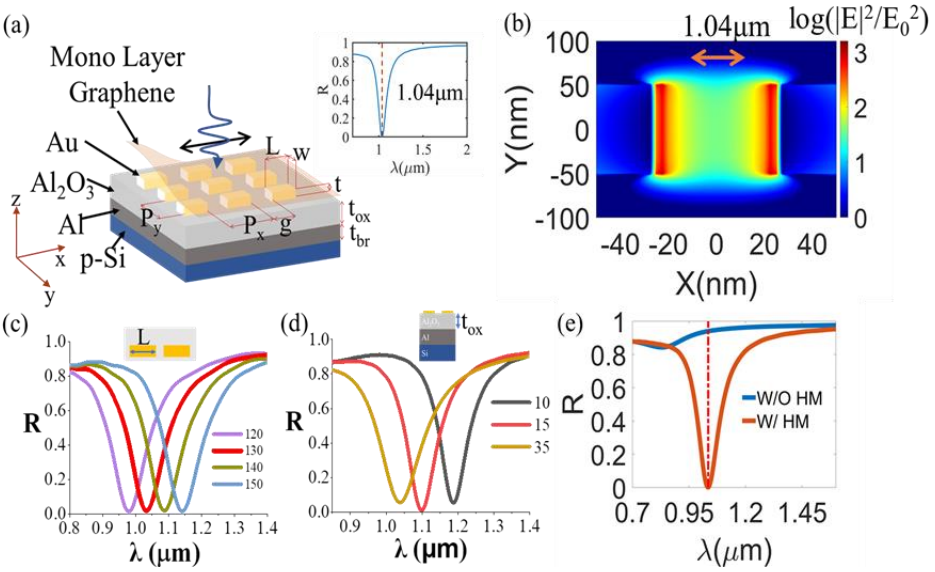


Figure 5. Graphene-plasmonic Hybrid Metasurface Design and Analysis. (a) Graphene-plasmonic Hybrid Metasurface (HM) Device Schematic. Inset Shows the Typical Reflection Spectra of the Device with Perfect Absorption at 1.04μm. (b) Near-field Enhancement Inside the Nanogap of the Nanobar Antenna at the Working Wavelength (1.04μm). GPMSA Device Parameter: Au Nanobar Length, L=130nm, Width, W=60nm, Gap, G=30nm, Thickness, t_{Au}=40nm, X-period, Px=160nm, Y-period, Py=450nm and Al_2O_3 Thickness, t_{AlO}=20nm. (c) the Working Wavelength of the Hybrid Metasurface Can Be Tailored by Changing the Length of the Nanobar. (d) the Absorption of Incident Light Can Be Controlled by Changing the Device's Different Parameters, Such as the Oxide Thickness. (e) Comparing the Effect on the Reflection (or Absorption) of Light When We Have Only Graphene Without Metasurface and With Metasurface.

After designing the plasmonic metasurface, we incorporate a 2D graphene layer on top of it. The device schematic is illustrated in Figure 5(a). A typical reflection spectrum of the

device, having perfect absorption at 1.035μm, is presented in the inset of Figure 5(a). With this device design, we can achieve a very large near-field enhancement inside the nanogaps of the nanobar array. The field enhancement at the working wavelength is presented in Figure 5(b). This large enhancement increases the light-matter interaction in graphene and absorption, reducing the device's saturation fluence. Another advantage of using this device design is that we can modify different parameters of the plasmonic metasurface (such as antenna length (L), gap (g), width (w), oxide thickness (t_{ox})) to change the working wavelength of the device (Figure 5(c)) and also the absorption at the working wavelength of the device (Figure 5(d)). This flexibility allows us to operate at a wavelength of our choice with the absorption we want. Moreover, these devices with different absorption can achieve different saturation fluences per our needs.

After the device fabrication, the structural parameters were changed, and consequently, the spectral response of the devices deviated from the ideal condition that we simulated here. For fabricated GPMSA devices, the best minimum reflection at the resonance (at around 1.035μm) we could achieve is around ~20% (Figure 13). Therefore, to recreate this in the simulation we also changed the device parameters to achieve ~20% minimum reflection at the resonance (as shown in Figure 11a and all other simulations presented after that).

So, the GPMSA device parameters used in all simulations presented later in this dissertation are Au nanobar length, L=120nm, width, W=100nm, gap, g=30nm, Au nanobar thickness, t_{Au}=40nm, x-period, P_x=150nm, y-period, P_y=450nm and Al_2O_3 thickness, t_{AlO}=40nm.

2.2.3 Comparison of Saturable Absorption of Our Hybrid Device and Graphene

We can compare the absorption of saturable absorber devices when only graphene is used and when we are using the hybrid metasurface design to obtain a glimpse of the improved performance of our device. A detailed description of the comparison is presented in terms of one of the device's most important performance parameters, saturation fluence, given below.

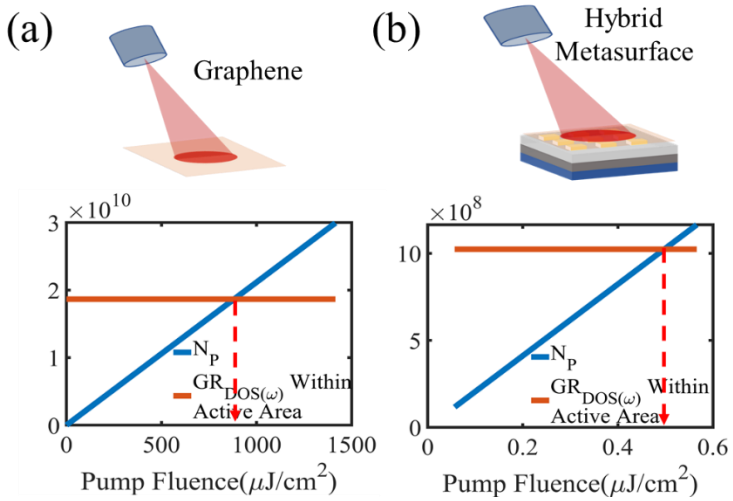


Figure 6: (a) Saturation Fluence of Only Graphene When There is No Enhancement. (b) Saturation Fluence of Our Hybrid Metasurface

We can compare the performance of the hybrid metasurface device in terms of saturation fluence with only graphene. For this purpose, we can set up two parameters, density of states of graphene inside the active area, $GR_{DOS(\omega)}$And the total number of absorbed photons, N_P. Given as follows,

$$GR_{DOS(\omega)} = DOS(\omega)\, E_{ph}\, ActiveArea \tag{11}$$

Here, $DOS(\omega) = \frac{2E_{ph}}{\pi\hbar^2 {V_F}^2}$

$$N_P = A\sqrt{|E|^2}\, n_p = A\sqrt{|E|^2}\, \frac{E_p}{\hbar\omega} = A\sqrt{|E|^2}\, \frac{P_{inc}\, T_p}{\hbar\omega} \tag{12}$$

Here, DOS ($J^{-1}m^{-2}$) is the density of states of graphene. E_{ph} is the photon energy ($\hbar\omega$) of the incident light. V_F is the fermi velocity of graphene ($\approx 10^6 ms^{-1}$). $\hbar$ is the reduced plank's constant ($h/2\pi$). A is the absorption of light. $|E|^2$ is the near field enhancement. $n_p$is the total number of photons in one pulse. E_p is the engy of the incident laser pulse. ω ($=2\pi f$) is the laser pulse frequency at the laser wavelength (1.035μm). P_{inc} is the incident average power of the laser. T_p is the pulse period of the laser (10ns). Let, Incident beam is circular-shaped at the focus point with a diameter (2r) 150μm. So, the beam area ($A_{beam} = \pi r^2$) is 1.7671e-8 m^2.

Only Graphene:

If we consider only graphene, there is no enhancement ($|E|^2=1$) and the absorption, A, equals its weak absorption (≈2.3%). And, as we are considering continuous graphene, the active area is equal to the beam area. So, the number of states available to fill up, inside the active area of graphene is

$GR_{DOS(\omega)} = DOS(\omega)\ E_{ph}\ (\text{Active Area}) = DOS(\omega)\ E_{ph}\ A_{beam} = 1.865e10$

And the number of photons absorbed by graphene after laser light is incident on it,

$$N_P = A\sqrt{|E|^2}\ \frac{T_p}{\hbar\omega}\ P_{inc} = 1.1975e9\ P_{inc} \qquad (13)$$

For different incident average power, P_{inc}, from 0.1mW to 25000mW, we can calculate the N_P (provided in Figure 6(a)). We can see that the available states in graphene ($GR_{DOS(\omega)}$) fills up around 900μJ/cm^2 or ≈ 1mJ/cm^2 pump fluence. This means the saturation fluence of the device with only graphene is around 1mJ/cm^2. This is very close to the reported saturation fluence in literature for saturable absorbers with only graphene[41,122].

Graphene-plasmonic Hybrid Metasurface:

Now for our hybrid metasurface, let's consider a near-field enhancement, $|E|^2$, of 500 and very low absorption, A, around 10%. As we are using nano-bar antenna array, the enhancement happens mostly inside the nanogaps. So, we can consider the hot spots or the nanogaps where the most of the light is absorbed.

Active area= total area of all the hot spots under the beam = $A_{hotspot}$.

So, the number of states available to fill up, inside the hotspots of graphene is

$GR_{DOS(\omega)} = DOS(\omega)\ E_{ph}$ (Active Area) = $DOS(\omega)\ E_{ph}\ A_{hotspot}$ = 8.29E8

And the number of photons absorbed by graphene after laser light is incident on it,

$$N_P = A\sqrt{|E|^2}\ \frac{T_p}{\hbar\omega}\ P_{inc} = 1.16e11\ P_{inc} \quad (14)$$

For different incident power, P_{inc}, from 0.1mW to 25000mW, N_P is calculated and plotted in Figure 6(b). From Figure 6(b), we can see that to fill up the available states we only need 0.5μJ/cm^2 pump fluence. So, the saturation fluence of our device at this condition is 0.5μJ/cm^2. This is already a thousand times smaller than the case for only graphene. If we can achieve 50% absorption with the hybrid metasurface, the achievable saturation fluence is 0.08μJ/cm^2. For, 80% absorption, achievable saturation fluence is 0.05μJ/cm^2. So, we should be able to achieve less saturation fluence for larger absorptions of the hybrid metasurface.

This is just a rough calculation. To produce a more stringent model of the device we need to investigate the carrier dynamics in Graphene after laser light falls on it. Carriers will be excited from valence band to conduction band and the excited carriers will go through some thermalization processes which is not considered here. The next section discusses about this in detail.

2.3 Numerical Modeling of Graphene and Theoretical Analysis of GPMSA

2.3.1 Ultrafast Transient Behavior of Graphene

Saturable absorbers based on graphene can utilize the ultrafast carrier dynamics of graphene, so it can have a very fast recovery of the saturable absorption effect.

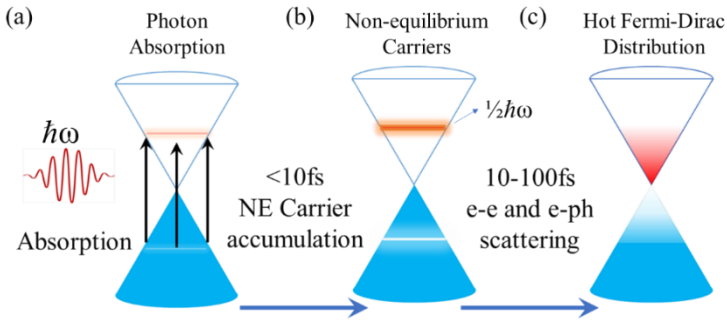


Figure 7: Light with Photon Energy $\hbar\omega$ Is Incident on Graphene. (a) Excitation of Carriers from Valence Band (VB) to Conduction Band (CB) under High Light Intensity (b) Nonequilibrium Carrier Concentration Produced from the Excitation (c) Within 10-100fs, the Excited Non-equilibrium Carriers Will Thermalize to Hot Fermi-Dirac Distribution Due to Carrier-carrier and Carrier-Optical Phonon Scattering.

When intense laser light hits graphene, a large number of ultrafast carriers are excited from the valence band to the conduction band. These large numbers of excited carriers in CB (and VB) form a non-equilibrium carrier distribution within the energy band covered by the laser bandwidth. Within 10-100fs, these non-equilibrium carriers in CB experience electron-electron and electron-optical phonon scattering and relaxes to a hot equilibrium Fermi-Dirac distribution[11]. Figure 7 illustrates the excitation of carriers from valence band (VB) to conduction band (CB) and the subsequent processes upon laser light incidence on the device. To model these processes, we can consult the dynamic interplay between the

occupation probability of electrons in the conduction band (f_C (t, ω)) and the valence band (f_V (t, ω)) upon laser excitation. These dynamics of the carrier occupation probabilities are explained sufficiently by two semi-empirical coupled differential equations relating the density of states of graphene at the excitation photon energy (D(E)), incident light intensity (I (t, ω)), and the associated relaxation process, τ_1, which are provided in equations (15) and (16) as shown below [124],

$$\frac{\partial f_V(t)}{\partial t} = -f_V(t)\,\frac{\pi\alpha}{D(E)\,\hbar\omega}\,I_0 exp(-\frac{t^2}{\tau^2}) + f_C(t)\,\frac{\pi\alpha}{D(E)\,\hbar\omega}\,I_0 exp(-\frac{t^2}{\tau^2}) + \frac{1-f_V(t)}{\tau_1} \quad (15)$$

$$\frac{\partial f_C(t)}{\partial t} = -f_C(t)\,\frac{\pi\alpha}{D(E)\,\hbar\omega}\,I_0 exp(-\frac{t^2}{\tau^2}) + f_V(t)\,\frac{\pi\alpha}{D(E)\,\hbar\omega}\,I_0 exp(-\frac{t^2}{\tau^2}) - \frac{f_C(t)}{\tau_1} \quad (16)$$

Here, $I(t) = I_0 \exp(-t^2/\tau^2)$ and $D(E) = D(\hbar\omega/2) = \hbar\omega/(\pi\hbar^2 v_F^2)$. These two coupled differential equations explain the absorption, stimulated emission, and the subsequent relaxation of the excited non-equilibrium carriers. τ_1 constitutes the relaxation time corresponding to the carrier -carrier, and carrier -phonon scattering rate, and it times the thermalization from non-equilibrium carrier accumulation to the hot fermi Dirac distribution indicated in Figure 7[125,126]. For saturable absorption, we are primarily interested in the ultrafast accumulation of non-equilibrium photoexcited carriers from VB to CB and the subsequent relaxation of the carriers from the non-equilibrium states to the equilibrium Fermi-Dirac distribution near the Dirac point. The most contributing relaxation channels in this process are the carrier-carrier and carrier-optical phonon scattering, which happens within 10-100fs. So,

the scattering contributions from other sources such as carrier-acoustic phonon and phonon-phonon scattering are not considered here as they have a pico-second or larger time constant[11]. So, here, the relaxation time, τ_1, only consists of the excited carriers' intraband electron-electron and electron-optical phonon scattering rates. Furthermore, as τ_1 is ultrafast, it is only valid within the energy band of half of the photon energy of the incident laser light, which is the energy band of our interest.

So, after finding the time-dependent occupation probability from these two equations, we can calculate how many electrons are excited into the conduction band after the excitation process with the following equation,

$$N(t) = \big(fv(t) - fc(t)\big)\, D(E) \tag{17}$$

Graphene has a constant absorption of $\pi\alpha_f$ or 2.3% at all wavelengths. As the carriers accumulate in the conduction band from the valence band, the absorption decreases from 2.3%. We can account for the changing absorption, $A(t,\omega)$, by considering the time-dependent occupation probability of electrons in CB ($f_C(t, \omega)$) and VB ($f_V(t, \omega)$),

$$A(t,\omega) = \pi\alpha_f\big(f_V(t,\omega) - f_C(t,\omega)\big) \tag{18}$$

Here, $A(t,\omega)$ is the time-dependent absorption of graphene. We can convert this absorption into an absorption coefficient, $\alpha(t,\omega)$, by considering a thickness (d) of 0.3nm for the graphene monolayer. Please see the discussion in Appendix A for the absorption coefficient calculation of monolayer graphene. For these simulations, we take the graphene absorption coefficient long before pumping, $\alpha(-\infty,\omega)$, as the initial or base value, and then we try to calculate the changes in the absorption coefficient due to laser pumping, $\Delta\alpha(t,\omega) = \alpha(t,\omega) - \alpha(-\infty,\omega)$. From the changes in time-dependent absorption coefficient,

$\Delta\alpha(t,\omega)$, the change in complex refractive index, $\Delta n_T(t,\omega) = \Delta n(t,\omega) + i\Delta k(t,\omega)$, of graphene can be calculated with the help of the Kramers-kroning model[127],

$$\Delta k(t,\omega) = c\frac{\Delta\alpha(t,\omega)}{2\omega} \tag{19}$$

$$\Delta n(t,\omega) = \frac{c}{\pi}\, p.v \int_{\Omega_l}^{\Omega_m} \frac{\Delta\alpha(t,\Omega)}{\Omega^2-\omega^2}\, d\Omega$$

$$\Delta n(t,\omega) = \frac{c}{\pi}\left[\int_{\Omega_l}^{\omega_-} \frac{\Delta\alpha(t,\Omega)}{\Omega^2-\omega^2}\, d\Omega + \int_{\omega_+}^{\Omega_l} \frac{\Delta\alpha(t,\Omega)}{\Omega^2-\omega^2}\, d\Omega\right] \tag{20}$$

Here, KK model relates the change in the imaginary part ($\Delta k(t,\omega)$) and the real part ($\Delta n(t,\omega)$) of graphene's complex refractive index with its time-dependent change in absorption coefficient, $\Delta\alpha(t,\omega)$. To calculate the effective total complex refractive index upon laser excitation, we extract the optical surface conductivity, $\sigma_S(-\infty,\omega)$, of non-pumped graphene from FDTD numerical model in Lumerical Inc. FDTD simulation software. From $\sigma_S(-\infty,\omega)$, we can calculate in-plane ($\|$) complex permittivity, $\varepsilon(-\infty,\omega) = 1 + \frac{i\sigma_S(-\infty,\omega)}{\varepsilon_0\omega d}$And then complex refractive index, $n_T(-\infty,\omega) = \sqrt{\varepsilon(-\infty,\omega)}$Of non-pumped graphene. Then we add the time-dependent change in the complex refractive index, $\Delta n_T(t,\omega)$= $\Delta n(t,\omega) + i\Delta k(t,\omega)$, to obtain the total complex refractive index of graphene,

$$n_T(t,\omega) = n_T(-\infty,\omega) + \Delta n_T(t,\omega) \tag{21}$$

$n_T(t,\omega)$ includes the instantaneous time-dependent index change due to laser pumping. The complex permittivity, $\varepsilon(t,\omega)$, of the graphene is then calculated from the complex index, $\varepsilon(t,\omega) = n_T(t,\omega)^2 = (n(t,\omega) + ik(t,\omega))^2$. Time-dependent optical surface

conductivity, $\sigma_S(t,\omega)$, of the graphene can be calculated from the in-plane (||) complex permittivity of graphene as follows,

$$\varepsilon(t,\omega) = 1 + \frac{i\sigma_S(t,\omega)}{\varepsilon_0 \omega d} \tag{22}$$

Using this instantaneous change in graphene's optical surface conductivity, $\sigma_S(t,\omega)$, we can perform a simulation with our hybrid metasurface using the 2D surface conductivity model in Lumerical FDTD and find the time-dependent reflection spectra of our device. The simulation is done in a self-consistent way, as explained in Appendix A.

When we shine laser light on graphene, the simulated instantaneous change in the carrier concentration (non-equilibrium carrier accumulation) in the conduction band around the laser bandwidth is illustrated in Figure 8. Due to this large instantaneous change in carrier concentration, we observe an instantaneous change in the optical conductivity of graphene. Figure 9(a, b) illustrates this large instantaneous change in the real and imaginary part of the graphene's optical conductivity (σ_S(μS)). And, due to this large instantaneous change in σ_S(μS), the absorption of graphene reduces instantaneously from its typical weak absorption value. Figure 9(c, d) illustrates the instantaneous change in graphene's absorption, from 2.3% to 1.4%, at a pumping fluence of 5.66μJ/cm^2. Although it is instantaneous, the absorption change is less than 1% for only graphene.

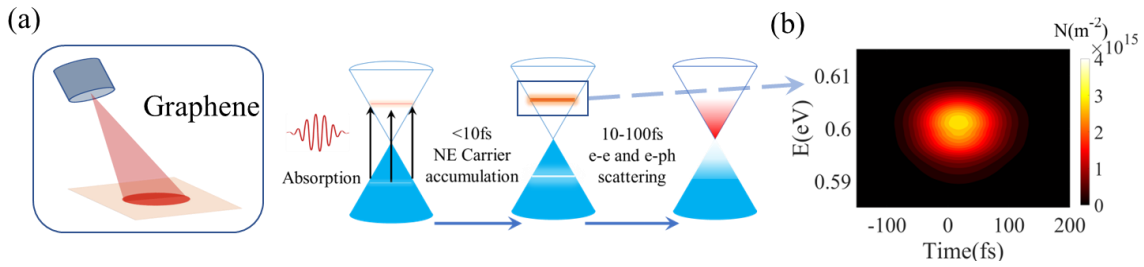


Figure 8: (a) Shining an Ultrafast Laser on Graphene. The Electrons in Valence Band (VB) Absorb the Incoming Photon, Get Excited to the Conduction Band (CB), and Form a Non-equilibrium (NE) Carrier Distribution, in CB, Around the Energy Band Covered by the Laser Wavelength Bandwidth. Due to Electron-electron (e-e) and Electron-optical Phonon (e-ph) Scattering, the Non-equilibrium Carriers Relax to a Hot Fermi-Dirac Distribution Within 10 – 100fs (Relaxation Time, τ_1). (b) A large Number of Excited Non-equilibrium Carriers Gets Distributed Around the Laser Wavelength (Energy Band), Covering the Finite Bandwidth of the Laser.

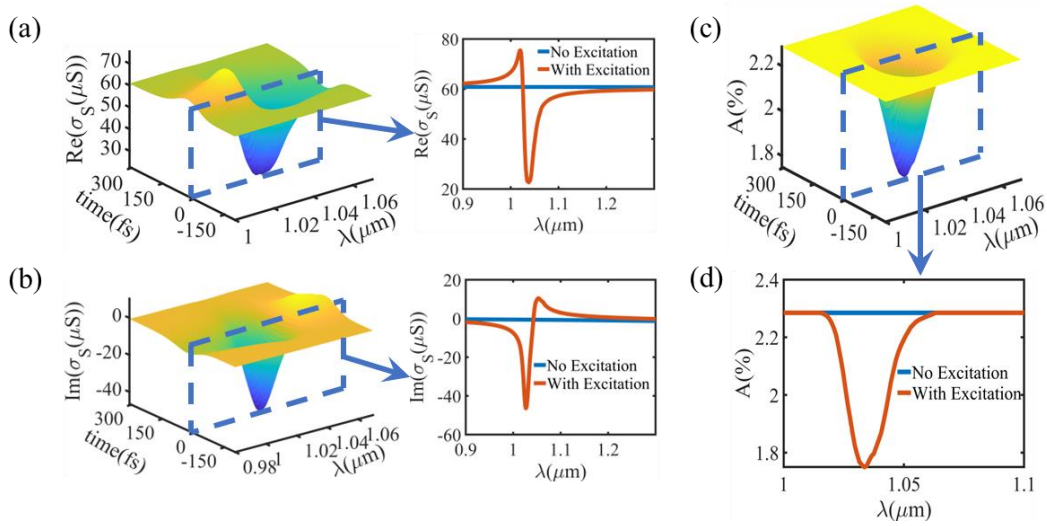


Figure 9: (a) The Real Part of Optical Conductivity (Re(σ_S(μS))) and (b) The Imaginary Part of Optical Conductivity (Im(σ_S(μS))) of Graphene Experience Instantaneous Changes Due to the Non-equilibrium Carrier Accumulation in CB. Inset Shows the Changes in

Re($\sigma_S(\mu S)$) and Im($\sigma_S(\mu S)$) Around the Laser Bandwidth. (c) Monolayer Graphene Absorption Changes Instantaneously as the Optical Conductivity of Graphene ($\sigma_S(\mu S)$) Is Instantaneously Changed. (d) Closer Look at the Change in Monolayer Graphene Absorption with and Without Laser Excitation.

2.3.4 Ultrafast Transient Behavior of GPMSA

The transient behavior of GPMSA device is directly related to the ultrafast behavior of the excited carriers in graphene. When the intense laser light hits our device, the near field inside the nanogaps is greatly enhanced, as shown in Figure 5a, increasing the light-matter interaction with graphene.

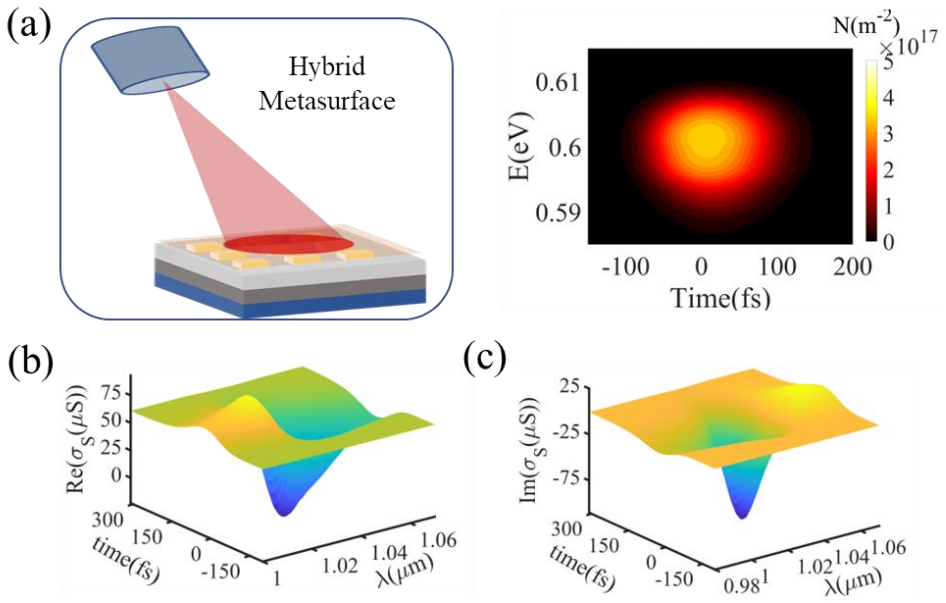


Figure 10: (a) Simulated 2D Contour Plot of the Photoexcited Non-Equilibrium Electron Distribution, N(m^{-2}), in Graphene Over the GPMSA Nanogaps at Energies Around $\frac{1}{2}\hbar\omega$, i.e., From 0.585eV to 0.615eV Above the Conduction Band Edge (E_C=0eV). The Pump Fluence is 112nJ/cm^2. The x-Axis Represents the Time Delay With Respect to the Peak of

the Incident Light Pulse. (b, c) Simulated Modulation of Optical Surface Conductivity, σ_S (μS/m), (b-Real and c-Imaginary part) of Graphene Inside the Nanogap of GPMSA With Respect to Time and Wavelength After Pump Excitation With Fluence 112nJ/cm^2. GPMSA Device Parameters Used in All Simulations Presented Here: Au Nanobar Length, L=120nm, Width, W=100nm, Gap, g=30nm, Thickness, t_{Au}=40nm, X-period, P_x=150nm, Y-period, P_y=450nm and Al_2O_3 Thickness, t_{AlO}=40nm. For This and the Following Simulations in this chapter, I Have Used the Same Laser Parameters as Our Measurement Setup: Pulse Width 100fs; Repetition Rate 100MHz; Central Wavelength 1.035μm, and Spectral Broadening ~17nm.

So, absorption in graphene is highly increased, and a large amount of photoexcited hot carriers are generated from valence band (VB) to conduction band (CB). Now, as we have combined plasmonic metasurface with graphene, graphene's optical conductivity (σ_S(μS)) goes through even larger instantaneous changes than as we discussed in the last section, illustrated in Figure 10.

Moreover, due to this large instantaneous change, the reflection spectra of the hybrid metasurface also change instantaneously, as illustrated in Figure 11a. We can see from Figure 11b, that the absolute maximum change in reflection (ΔR) is around 25% which is around 30 times larger than the only graphene under similar conditions.

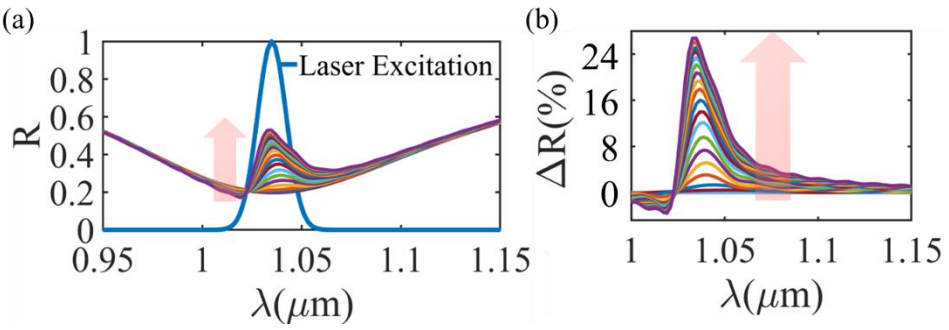


Figure 11:(a) Reflection Spectra of GPMSA Changing Over Time Upon Laser Excitation (b) Gradual Reflection modulation of GPMSA After Laser Excitation.

In Figure 12, a time and wavelength-dependent reflection modulation is shown. And depending on how fast the accumulated non-equilibrium carriers in graphene can relax (τ_1), we can expect to see different recovery times for our hybrid metasurface, as illustrated in Figure 12b. Also, Figure 12b shows that the maximum reflection modulation obtained from the device is different for different carrier relaxation time, τ_1. As the device recovery time is mainly dominated by the carrier relaxation time, we expect to see lower reflection modulation for faster devices. Figure 12c, shows the reflection modulation for different incident pump fluence and saturation behavior of the device. The simulations presented here are done in a self-consistent way presented in the previous section. The reflection modulation profile is fitted with a two-level atom model to extract the saturation fluence, and the obtained SA fluence is around $0.1\mu J/cm^2$. Figure 12c also shows the change in the reference device (only graphene on the same substrate as GPMSA) and suspended graphene. These devices show very little to no reflection modulation with very high saturation fluence.

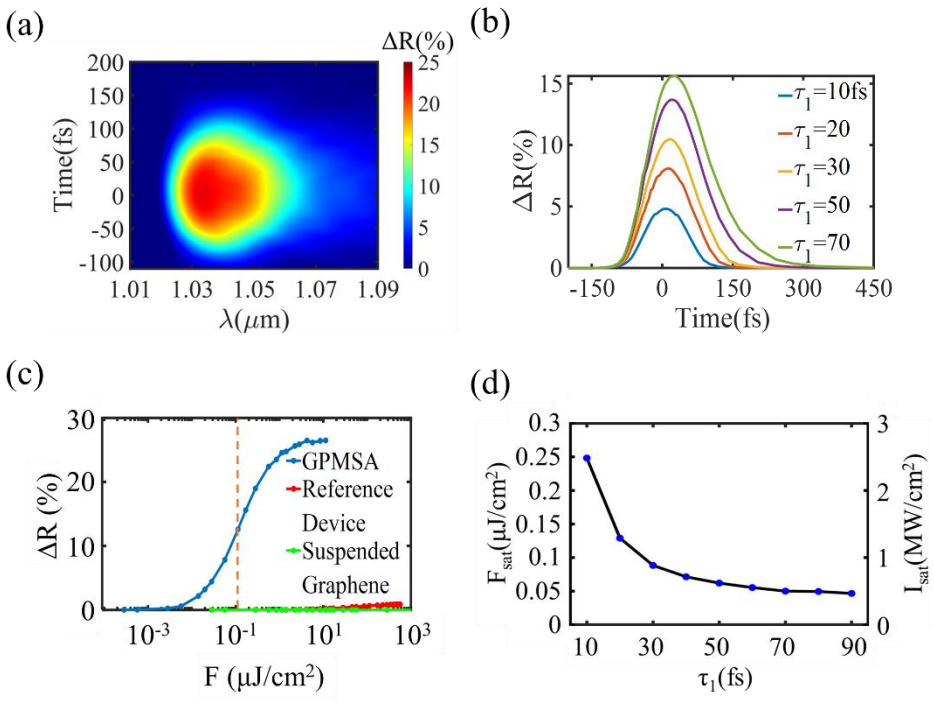


Figure 12: Simulated reflection modulation, ΔR (%), of GPMSA with respect to wavelength and time at 566nJ/cm^2 incident pump fluence. ΔR is ~25% at the laser pulse peak intensity around the laser center wavelength (1.035μm). (b) Simulated reflection modulation, ΔR (%), for different carrier relaxation times, τ_1, in graphene. (c) Simulated reflection modulation ΔR (%) of the same device at the peak intensity of different incident pump fluences for GMPSA, the reference device, and suspended graphene. (d) Simulated saturation fluence, F_{sat}, and peak saturation intensity, I_{sat} (MW/cm^2), of GPMSA at different graphene carrier-relaxation times, τ_1. Simulations are from my manuscript in preparation[128].

Figure 12d illustrates that the saturation fluence of the device is also dependent on the relaxation time constant of graphene, consequently device recovery time. From this figure, we can derive that, for faster devices, the saturation fluence tends to be larger. However,

as we are using GPMSA device, the high near-field enhancement makes it possible to absorb a very large number of photons in graphene; consequently, the resulting saturation fluence of the device becomes smaller.

CHAPTER 3

FABRICATION AND CHARACTERIZATIONS OF GPMSA (NIR) DEVICE

3.1 Device Fabrication and FTIR Characterization

To fabricate the device, we used a p-Si substrate. On it, we have evaporated (E-beam) 150nm Al back reflector and then deposited (ALD) 20nm Al2O3 dielectric spacer layer on top. The Au metasurface on the dielectric layer was fabricated by using Electron Beam Lithography (EBL) to pattern the double layer resist (PMMA/MMA), cold development (4^{o}) in 3:1 MIBK:IPA solution, Cr/Au (5nm/35nm) deposition, and lift-off using hot acetone for 12 hours. PMMA/MMA bilayer resist and cold development at 4^{o} is used for good lift-off of the plasmonic metasurface. The fabrication flow is shown in Figure 13(a). After fabrication, an SEM image of the device before graphene transfer is shown in Figure 13(b). Monolayer graphene is transferred on the device using a wet transfer method [see next section]. We confirm the monolayer graphene by doing Raman spectroscopy of the device, shown in Figure 14. Figure 14 shows that the 2D/G peak is more than two. In-house FTIR setup to measure the reflection spectra of the hybrid metasurface is shown in Figure 13(c). We are using liquid nitrogen-cooled MCT detector to detect the reflected spectra. Depending on the parameters (Length, width, gap, etc. of the plasmonic metasurface) we achieve in fabrication, we can tap into different wavelength ranges in the NIR wavelength range with the hybrid metasurface (Figure 13(d)). Reflection spectra before and after the graphene transfer for one of the hybrid metasurface is shown in Figure 13(e). We can see that the resonance wavelength of the device is blue-shifted by ~100nm, the reflection dip

decreased by 15%, indicating increased absorption, and the relative change in reflection (ΔR/R) at the working wavelength (~1.035um) is around 50% after graphene transfer.

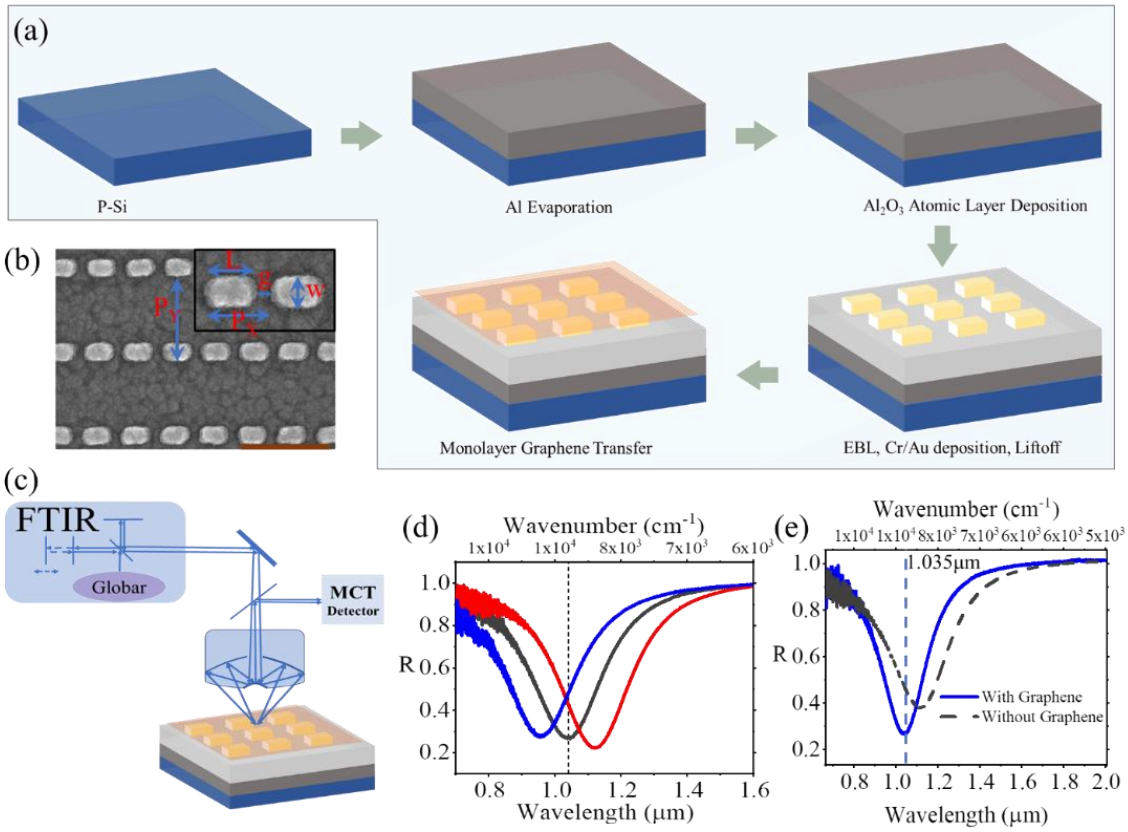


Figure 13. Fabrication of Graphene-plasmonic Hybrid Metasurface and Characterization. (a) Device Fabrication Process (b) An SEM Image of the Device after Fabrication (Before Graphene Transfer). The Scale Bar Is Indicated in Brown Solid Line on the Lower Right Corner (500nm). (c) FTIR Setup to Measure the Reflection Spectra of Our Device. (d) Measured Reflection Spectra of Different Devices. We Can Tap into Different Working Wavelengths in the near Ir Wavelength Range with Different Fabricated Parameters of the Hybrid Metasurface. (e) Reflection Spectra Before and after Graphene Transfer Indicate a 100nm Blue Shift of the Resonance and a 15% Decrease in the Resonance Dip.

3.2 Graphene Transfer and Raman Spectroscopy

We used CVD graphene sample on a Cu foil (covered both sides). We cut a small piece, enough to cover the whole area of the fabricated device, and put it on a glass slide. We used one drop of water on the glass slide to hold the small piece in its place. Then we spin-coated 950K A4 PMMA layer (at 2500 RPM for 1 minute) on top of the graphene piece to protect the graphene layer on that side from the following processing steps. Then we flip the small piece and expose the bare graphene side of the Cu foil. We take another glass slide and use Kapton tape on all four sides of the piece to hold it on the glass slide. Then put it inside an oxygen(O_2) plasma etcher for 15 minutes to etch the graphene off that side of the Cu foil. After the process is done, we remove the graphene-covered Cu foil (on one side now) and cut off the four edges of the foil as they were protected by the Kapton tape during the plasma etching and still have graphene on both sides. Then we put the graphene-covered Cu foil (Cu side down) in a copper etchant (a mixture of $CuCl_2$ and HCl). After the etching is completed, we can see a translucent piece of graphene (covered with PMMA on top) floating on the etchant. We scooped the floating graphene piece from the etchant with a small piece of SiO_2/Si wafer and put it inside a Deionized (DI) water bath. We provide three separate DI water baths (each one for 5 minutes) to the graphene piece to remove all the Cu etchant from it. After the last bath, we move the graphene sample to another DI water bath. Then we take our device (where we want to transfer graphene) and scoop the graphene sample out of the DI water with it (transferring graphene on our device). Then we gently blow the sample with an N_2 gun (optional as it may tear graphene) to remove any water puddle underneath it and let it dry on its own for 24 hours. After that,

we put the sample in an acetone bath for 5 minutes and then rinse it with acetone and IPA to remove all the PMMA from the top. Then blow-dry with an N_2 gun.

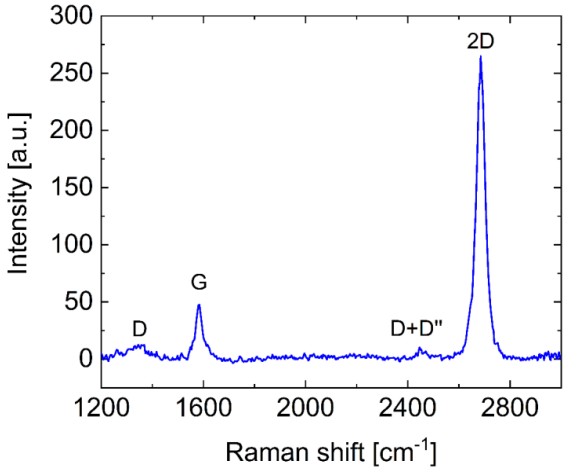


Figure 14: Raman Spectrum of the Transferred Graphene on the Metasurface. The Ratio Between 2d and G Peak Is More than 2, Which Indicates Monolayer Graphene and a Very Low D Peak Indicates High Quality and Defect-free Graphene.

After graphene transfer, we verify the monolayer graphene by doing Raman spectroscopy. The result of Raman spectroscopy is provided in Figure 14. The 2D peak around 2700cm^{-1} is more than two times compared to the G peak around 1580cm^{-1}. So, it confirms the monolayer graphene. Also, the D peak is very small, indicating a very minimal defect or disorder in graphene, ensuring high quality.

3.3 Saturable Absorption/ Nonlinear Response Measurement

The optical setup for the saturable absorption measurement is shown in Figure 15(a). The setup comprises a femtosecond laser (Orang-Menlo System) with a 100fs pulse width and a 100MHz repetition rate at wavelength 1.035μm ±17nm. The laser's output power can be changed from 0 to 1270mW by varying its pump diode currents. We have two power meters

in the setup, one at the laser's incident path and another at the device's reflecting path. We have used a 45º off-axis parabolic mirror to focus the laser on the sample. The diameter of the focus point on the sample is around 150µm (focused beam area is 150µm x 150µm) with a ±10% accuracy. As the device area is 200µmx200µm, we focused the laser beam at the device's midpoint for maximum effect. The incident and reflected power of the laser at a different output power of the laser are measured.

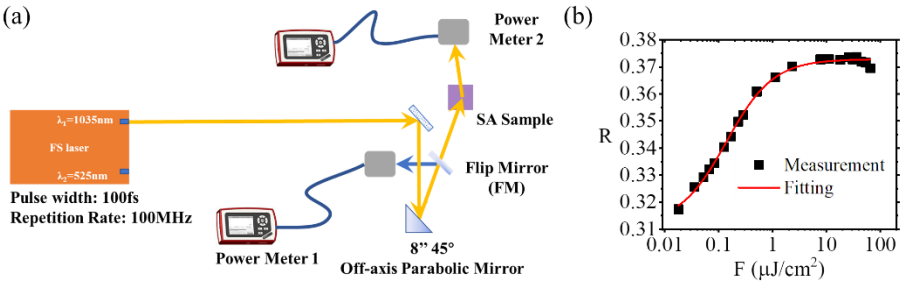


Figure 15. Saturable Absorption of Graphene-plasmonic Hybrid Metasurface. (A) Measurement Setup to Find the Saturable Absorption Property of the Device. (B) Change of Reflection of the Device at Different Incident Pump Fluence. The Reflection of the Device Is Fitted to Find the Saturation Fluence.

Figure 15(b) shows the change in reflection of the hybrid metasurface at different incident pump fluences. We can see that the reflection increases with increasing pump fluence and reaches a saturation level after it goes beyond the saturation fluence of the device. This behavior is a characteristic indicator of the devices' saturable absorption property. The figure shows that the net reflection modulation (ΔR) is around 5%, and the relative modulation depth (ΔR/R) is around 20% for a maximum 5µJ/cm^2 incident pump fluence. Moreover, the insertion loss of this hybrid metasurface is ~5dB.

To extract the device's saturation fluence, we follow the two-level atom model[129] and obtain a similar equation to describe the reflection profile of the hybrid metasurface,

$$R = 1 - (A_{NS} + \frac{A_s}{1 + \frac{F}{F_{s1}}} + \beta \frac{F^2}{F_{s2}^2}) \tag{23}$$

Here, A_{NS} describes the non-saturable absorption, A_s is the saturable absorption and β is the two-photon absorption of the device. We observe two-photon absorption in the measured reflection curve at different pump fluence; for example, in Figure 15(b), the reflection curve goes down after 34μJ/cm^2 pump fluence, which we tried to explain using the β parameter related to two photon absorption. F_{s1} is the saturation fluence within single photon absorption limit and F_{s2} is the saturation fluence for two photon absorption. Details of the fitting equation can be found in the next section. After fitting the measured data in Figure 15(b) with equation (23), we can find the unknown parameters such as F_{s1}, A_s and A_b. The fitted curve is shown in Figure 15(b) in red solid curve. Estimated saturation fluence from this fitting is 0.15μJ/cm^2. We have also done saturable absorption measurements for other hybrid metasurface devices, which can be found in Figure 17, Figure 27, and Figure 16.

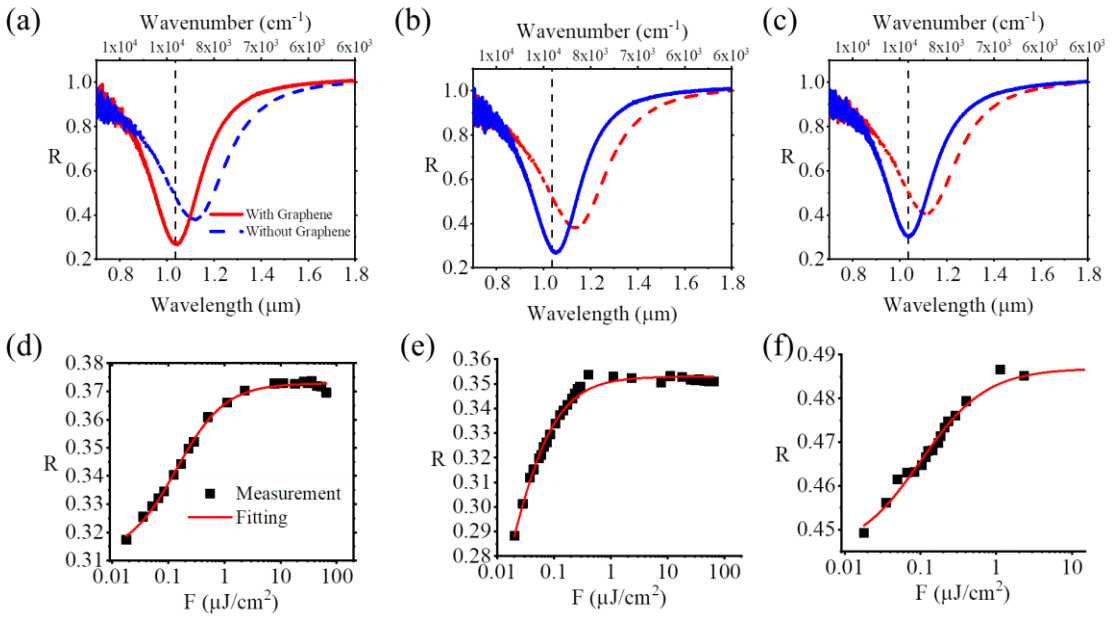


Figure 16: Saturable Absorption Measurement for 3 Different Hybrid Metasurface Devices Having Resonance at 1.035μm.

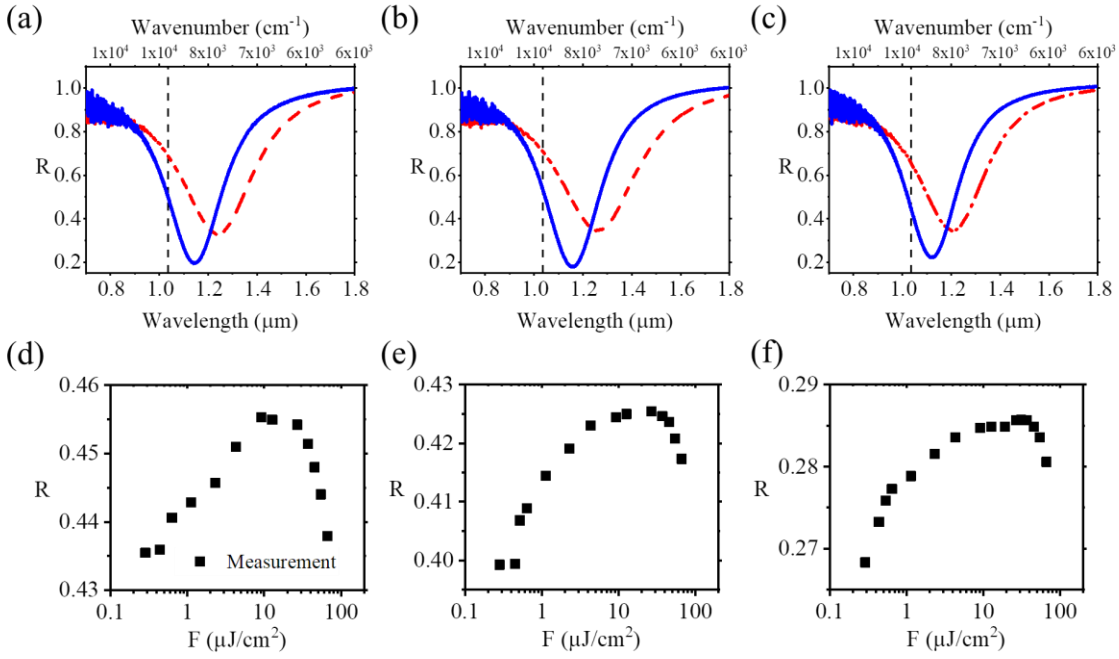


Figure 17: Saturable Absorption Measurement for 3 Different Hybrid Metasurface Devices with Resonance Larger than the Laser Wavelength (1.035μm)

To find the damage threshold of the hybrid metasurface, we have applied 72μJ/cm2 pump-fluence (maximum achievable with our laser), and we have not observed any damage (physical and optical property) to our sample. So, the hybrid metasurface has a damage threshold larger than 72μJ/cm2.

3.5 Transient Analysis Based on Pump-Probe Measurement

To investigate the transient behavior and find the recovery time of the hybrid metasurface, we have performed degenerate pump-probe measurement at 1035nm wavelength with a femtosecond laser (Orange-Menlo System) having a 100fs pulse width and 100MHz repetition rate. The optical setup is illustrated in Figure 18(a). A Pump-probe measurement, at a low pump-fluence of 0.67μJ/cm2, on the substrate (Al2O3/Al/Si), graphene, and graphene-plasmonic hybrid metasurface reveals the enhanced absorption and large reflection modulation within femtosecond time scale only by the hybrid metasurface (Figure 18(b)). The pump-probe measurement result for a hybrid metasurface at different pump fluences is presented in Figure 18(c). From this figure, we can see that the pump-probe signals are quite symmetric, which implies that the laser pulse width (100fs) used in the pump-probe measurement is not small enough to reveal the true ultrafast behavior of our device. Furthermore, a plot of the maximum value of the pump-probe signal at different pump fluences, Figure 18(d), reveals saturation of the pump-probe signal at higher pump fluences. This behavior corroborates our findings presented in Figure 15(d) and confirms the ultrafast saturable absorption property of the hybrid metasurface.

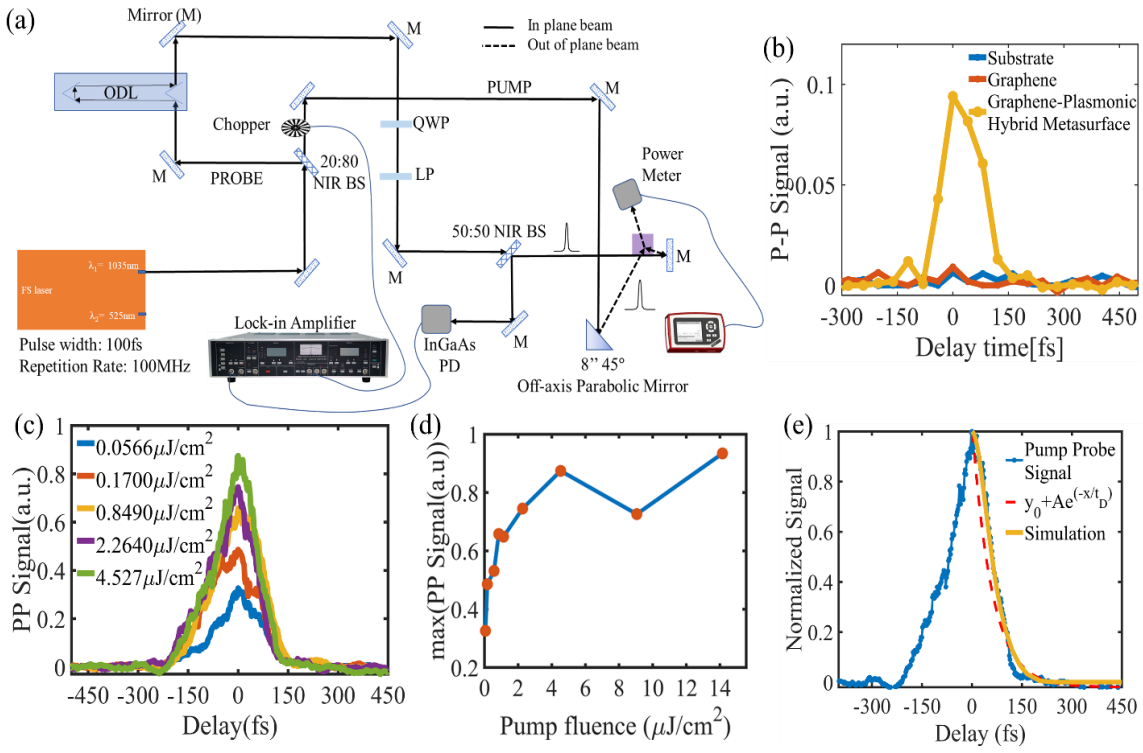


Figure 18. Degenerate Pump-Probe Measurement of Graphene-Plasmonic Hybrid Metasurface. (a) Pump-Probe Measurement Setup. A Chopper and Lock-in Amplifier Are Used to Increase the Signal-to-Noise Ratio (SNR) of the Measurement. (b) A Pump-Probe Measurement on the Substrate (Al_2O_3/Al/Si), Graphene, and Graphene-Plasmonic Hybrid Metasurface at a very low, 0.67μJ/cm^2, Pump-Fluence Show the Enhancement of the Pump-Probe Signal Only in the Hybrid Metasurface Due to Higher Absorption and Large Reflection Modulation. (c) Measured Pump-Probe Signal at Different Pump Fluences on the Same Hybrid Metasurface. (d) The Pump-Probe Signal Peaks (At Different Pump-Fluences) Follow Similar Saturable Absorption Characteristics As Presented in Figure 15(b). (e) Comparing the Measurement With the Simulated Pump-Probe Signal and Overlay of Exponential Fitting on the Decaying Part of the Measurement.

To find the decay time constant or the recovery time of the device, first, we fit the decaying part of the pump-probe signal with an exponential decay function to obtain the recovery time. Figure 18(e) shows the overlay of the measurement with the fitted exponential decay function (red dashed line). As we are using a larger pulse width (~100fs), in the measurement, than the recovery time of the hybrid metasurface, the exponential decay function does not provide a good fit with the measured pump-probe signal. So, we performed time-dependent FDTD simulations of our hybrid metasurface with the time-dependent optical conductivity of graphene. The process is explained in Appendix A. Then, we overlayed the simulation result on top of the measured pump-probe signal, and we found that the time-dependent simulations with 50fs to 60fs recovery times show good agreement with the measured pump-probe signal (Figure 18(e)). So, we can conclude that the device has a recovery time of less than <60fs. Due to its highly enhanced near-fields inside the nanogaps, the plasmonic metasurface provides new relaxation channels for the excited carrier in graphene CB to relax faster. Due to this effect, we can achieve a faster recovery time than other reported works in the literature (Section 4.6). We have also done more pump-probe measurements at different pump fluences for other hybrid metasurface devices that we fabricated. Please see Figure 19, Figure 20, and Figure 21.

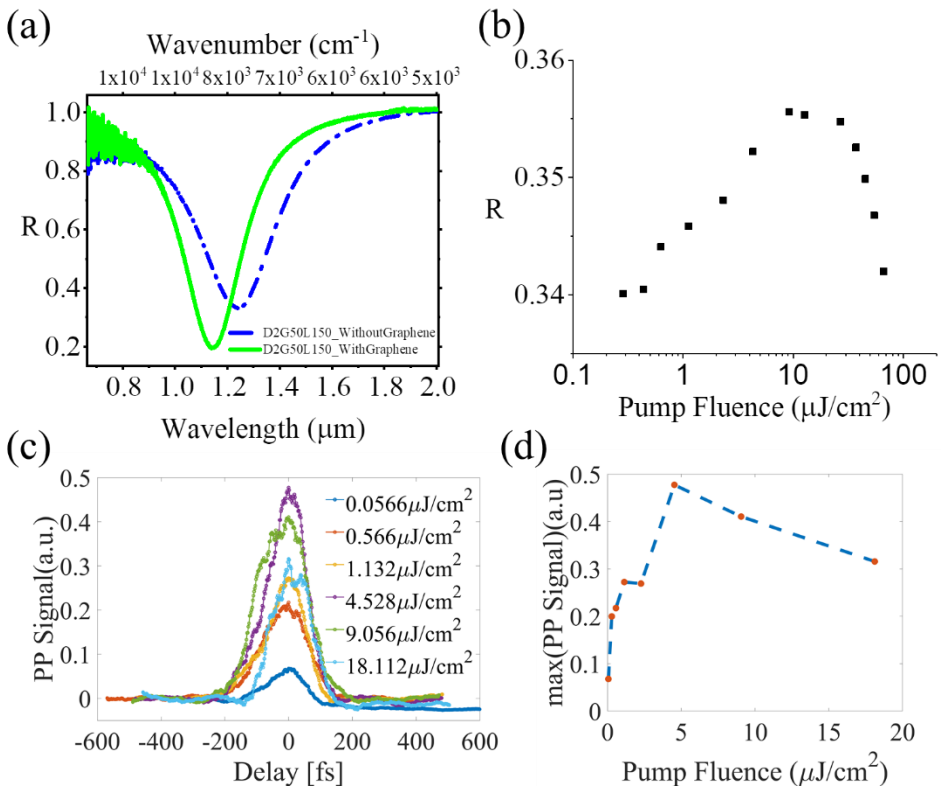


Figure 19: (a) Hybrid Metasurface Reflection Spectra Before and after Graphene Transfer. Device Has Resonance Larger than the Laser Wavelength (1.035μm) (b) Saturable Absorption Property of the Hybrid Metasurface Revealed in Reflection Measurement at Different Pump Fluence. For Large Pump Fluence Reflection Reduces. (c) Pump-probe Measurement Results for the Device at Different Pump Fluence. (d) Maximum of the Pump Probe Signal in (c) at Different Pump Fluences. Here Also, Maximum Signal Reduces for Larger Pump Fluences.

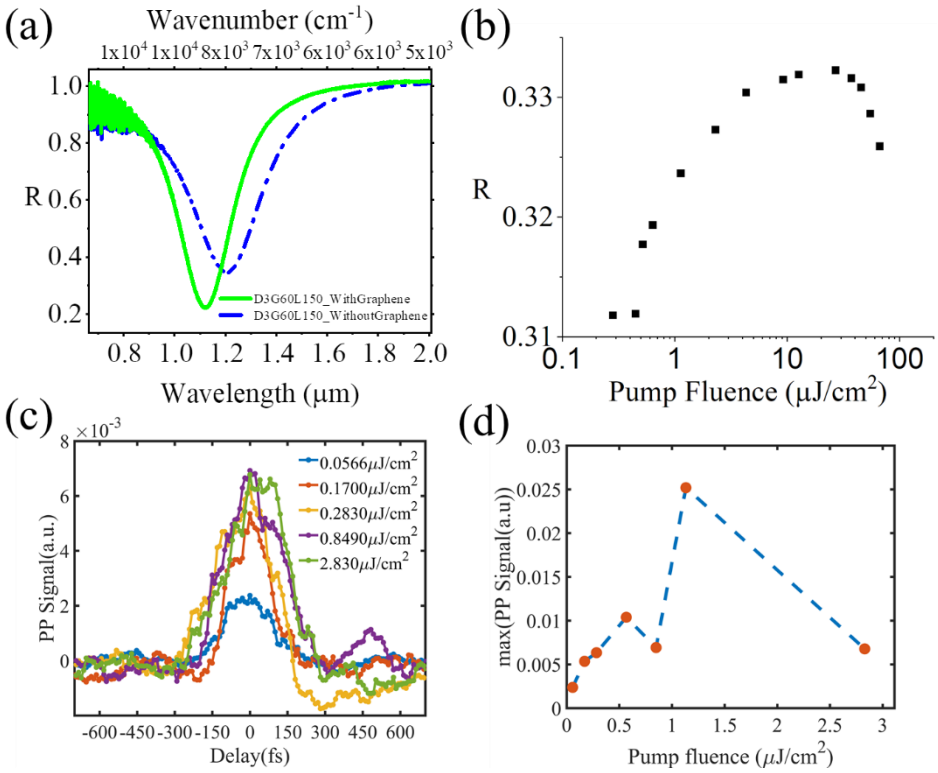


Figure 20 (a) Hybrid Metasurface Reflection Spectra Before and after Graphene Transfer. Device Has Resonance Larger than the Laser Wavelength (1.035μm) (a) Saturable Absorption Property of the Hybrid Metasurface Revealed in Reflection Measurement at Different Pump Fluence. For Large Pump Fluence Reflection Reduces. (c) Pump-probe Measurement Results for the Device at Different Pump Fluence. (d) Maximum of the Pump Probe Signal in (c) at Different Pump Fluences. Here Also, Maximum Signal Reduces for Larger Pump Fluences.

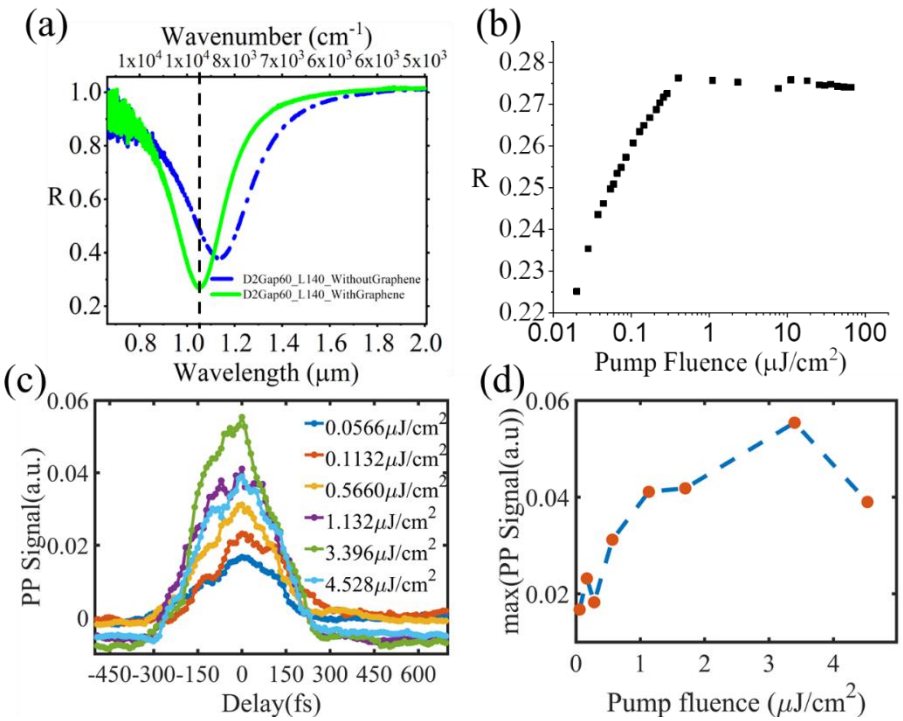


Figure 21: (a) Hybrid Metasurface Reflection Spectra Before and after Graphene Transfer. Device Has Resonance Equal to the Laser Wavelength (1.035μm) (b) Saturable Absorption Property of the Hybrid Metasurface Revealed in Reflection Measurement at Different Pump Fluence. (c) Pump-probe Measurement Results for the Device at Different Pump Fluence. (d) Plot of Maximum of the Pump Probe Signal in (c) at Different Pump Fluences. Here Also, Maximum Signal Reduces for Larger Pump Fluences.

3.6 Pulse Width Narrowing Effect Measurement

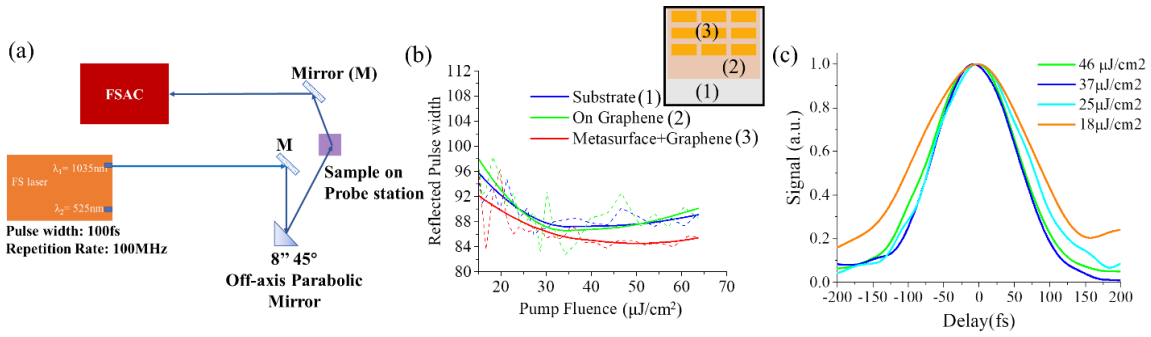


Figure 22. (a) Reflection Spectra of a Hybrid Metasurface Having Resonance Smaller than the Laser Wavelength (1.035μm). (b) Measured Pump-probe Signal For .283μj/Cm^2 Incident Pump Fluence Showing Ultrafast Relaxation of the Hybrid Metasurface after the Light Excitation Passes.

We know from theory that incident laser pulse width becomes narrower after being reflected from saturable absorber[39]. So, we tried to do some measurements to find if this is the case for our device. For this measurement, we shine laser light on our sample then redirect the reflected light beam into the autocorrelator (FSAC Interferometric Autocorrelator, Thorlabs) to measure the pulse width of the reflected light pulses, shown in Figure 22(a).

Figure 22(b) shows the measured reflected pulse widths from different parts of the sample (Substrate, graphene, graphene + plasmonic metasurface; as shown in the inset of Figure 22(b)). First, we can look at the blue and green curves representing the reflected pulse width from the device's substrate (Al_2O_3/Al) and only graphene on the substrate. They both follow a similar trend, and the pulse width decreases as we increase the power. We also notice that the pulse width for both, only graphene and the substrate are pretty similar. It

means there is no apparent effect on the laser's pulse width after interacting with these two parts of our device, mainly because the incident pump fluence is insufficient to induce any observable nonlinear effect in graphene or the substrate.

On the other hand, in Figure 22(b), we have the red curve representing the laser's pulse width after reflecting from the hybrid metasurface; this region has both the plasmonic metasurface and graphene. This pulse width is noticeably smaller than the previous two regions, and with increasing power, the pulse width becomes even narrower. It happens because the nonlinear effect in the hybrid metasurface increases with increased pump fluence. Figure 22(c) illustrates the reflected pulse shapes obtained from the hybrid metasurface at different pump fluences. From this figure, we can see that the pulse shape becomes narrower with increased pump fluence, and the leading edge of the pulse shows a clear dependence on the increasing pump fluence. Thus, we observe the pulse narrowing effect from these measurements and confirm the graphene-plasmonic hybrid metasurface's highly nonlinear saturable absorption effect.

3.7 Comparison of Our Work with Reported Works

Here we compare the performance of our hybrid metasurface SA with different reported works in the literature. In Table 2, we have listed some of the saturable absorbers in the literature and their performance parameters, such as saturation fluence, recovery time, modulation depth, and insertion loss.

Table 2: Comparison of Our Work with Reported Works

	Wavelength (nm)	Saturation fluence	Saturation intensity	Recovery time	Modulation Depth	Insertion Loss
SESAM[121]	1314	1.1 uJ/cm^2	-	-	4%	-10.2119dB
Monolayer Graphene[122]	750	5.3 mJ/cm^2	5300 MW/m^2	100fs	65.9%	4.9592 dB

Multilayer Graphene[41]	630	7.13 mJ/cm^2 (3 layer)	7100 MW/m^2 (3 layers)	0.21ps	~65% (3 layer)	4.34 dB
MoS_2 Nanoplatelets[105]	400, 800 and 1064	0.124 mJ/cm^2	136.1+/- 20GW/cm2, 279.6+/- 60 GW/cm2 and 8.7 MW/m^2	-	10% (N), 34% (N) and 4.6%	-, - and 0.34 dB
WS_2 film[107]	1560	1.1 J/cm^2	250 GW/m^2	-	1.20%	0.18 dB
Plasmonic-Metasurface[116]	1555	0.64 J/cm^2	-	-	60%	8.42 dB
Waveguide integrated with Graphene[24]	1550	0.17uJ/cm^2 (PS Laser) 0.04uJ/cm^2 (FS Laser)	-	~100fs	1.5dB	19 dB
Graphene-Bi2Te3 heterostructure[123]	1550	0.82-1.23 J/cm^2	72.9 to 109.5 GW/m^2	191-296fs	11.5-41.88% (depending on coverage of Bi_2Te_3)	2.5 to 5 dB
Few Layer $PtSe_2$ [106]	1064	-	0.346 GW/cm^2	-	26% (ΔT/T)	-
CMOS-compatible Gr-Si Hybrid WG[21]	1565	1.353 mJ/cm^2	-	1.65 ps	22.7% (ΔT/T)	-
Large-Diameter SWCNT[109]	2300	18 uJ/cm^2	-	220fs	30% (ΔT/T)	-
Carbon-nanotubes[110]	1500-1630	0.56 J/cm^2	-	~1ps	9.182%(ΔT)	-
Ionic Liquid Gated CNT[111]	1540	-	-	80 to 190fs (At different Applied voltages)	0.7 to 3.2 % (ΔT/T)	-
Our work	**1035**	**0.15 uJ/cm^2**	**0.15 MW/m^2**	**<60fs**	**20% (ΔR/R)**	**0.3dB**

Modulation Depth= ΔR/R*100% or ΔR or 10 log (Ron/Roff) dB.

Insertion loss = 10 log R dB.

In Figure 23(a, b), we illustrate the comparison between different approaches and our hybrid metasurface.

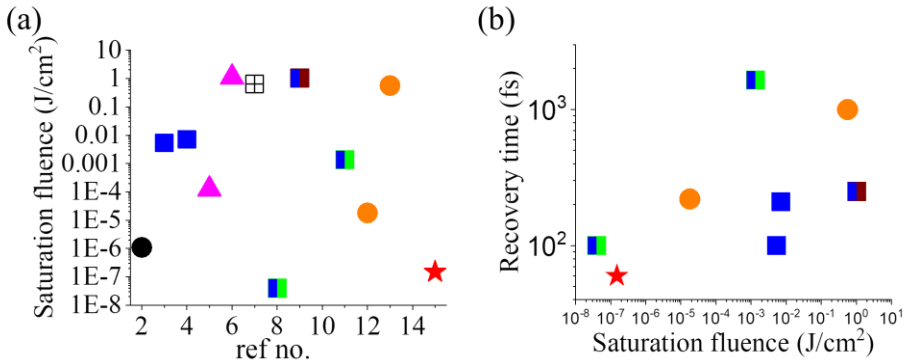


Figure 23: Our Work in Comparison to Other Reported Works in Literature. (a) Performance Parameter Saturation Fluence (b) Recovery Time Combined with Saturation Fluence. The Red Star Represents Our Work. The Reference Number is Provided as Provide in the Reference List.

3.8 Hot Electron Assisted Ultrafast Graphene Plasmonic Hybrid Metasurface Reverse Saturable Absorber in NIR Wavelength

In this part of the dissertation, I have shown that combining graphene monolayer with plasmonic metasurface makes it possible to achieve reverse saturable absorption with the help of hot carriers generated by the plasmonic metasurface while maintaining ultrafast response time. Using the reverse saturable absorption property of our subwavelength hybrid planar metasurface design, it is possible to develop ultrafast low-threshold optical limiters, switch or pulse shapers for ultrafast-photonic circuits where fast response time and the low-optical limiting threshold are essential.

3.8.1 Design, Simulation, and Measurement Results

The hybrid metasurface presented here is a combination of a 2D graphene monolayer and planar metasurface of plasmonic nanobar antenna array [69], illustrated in

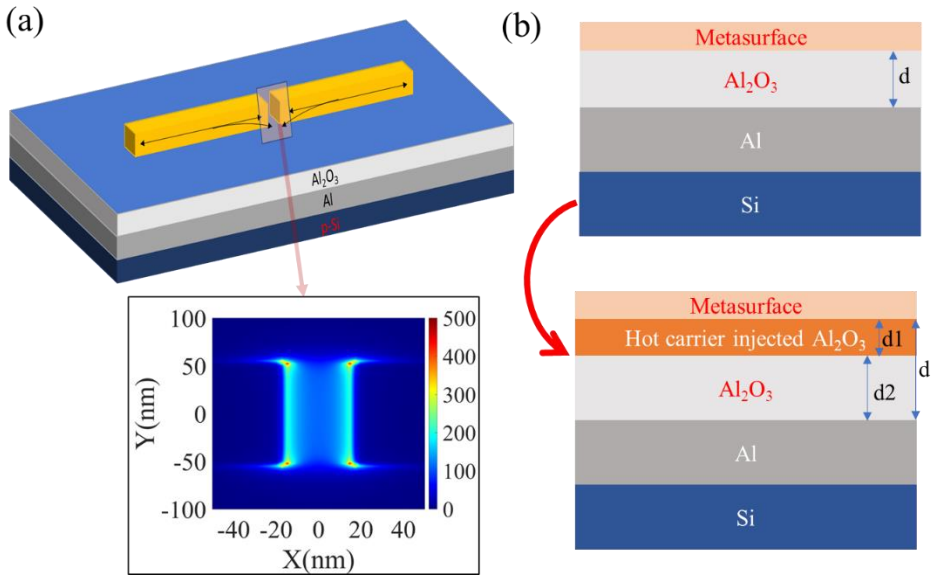
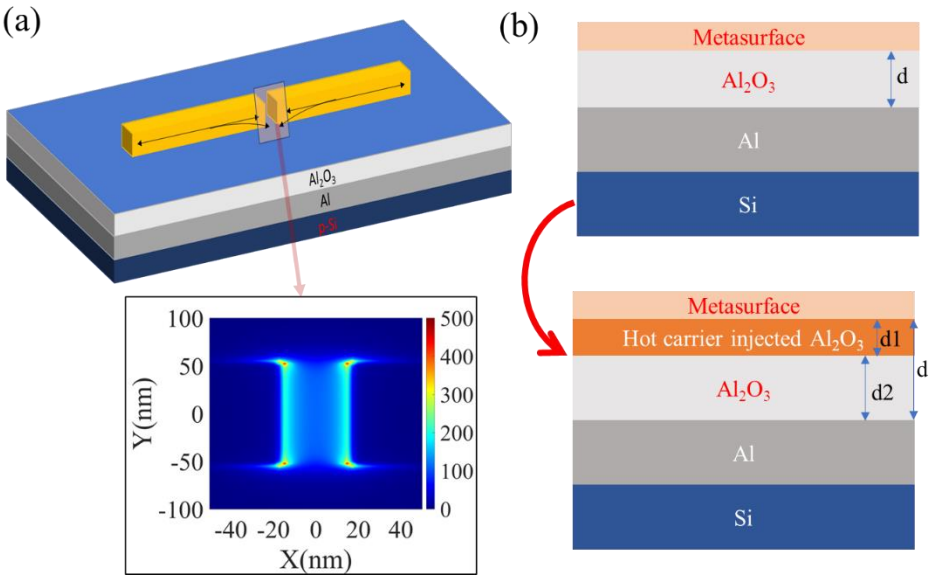


Figure 24(a). The device has a broadband response in the near-IR wavelength range,



illustrated in

Figure 24(c)-simulation and Figure 26(a)-measured. Also, it has a very large near-field enhancement even at off-resonance (inset,

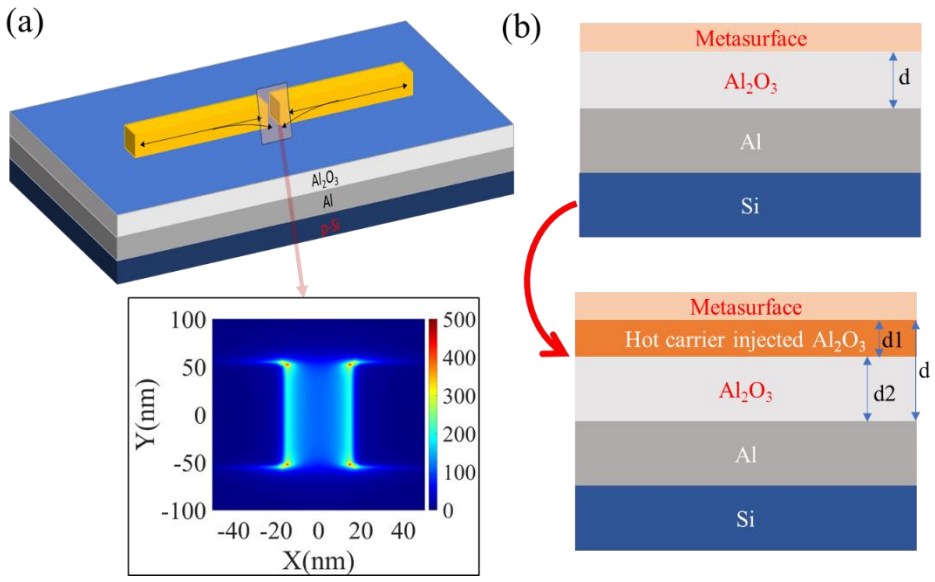


Figure 24(a)).

Two things happen when the device is pumped at a wavelength larger than the device's resonance. First, due to the large near-field enhancement inside the nanogap, a huge number of hot carriers are generated and injected into the Al_2O_3 spacer layer. Due to this,

a small, highly doped Al2O3 region is created beneath the Metasurface layer (

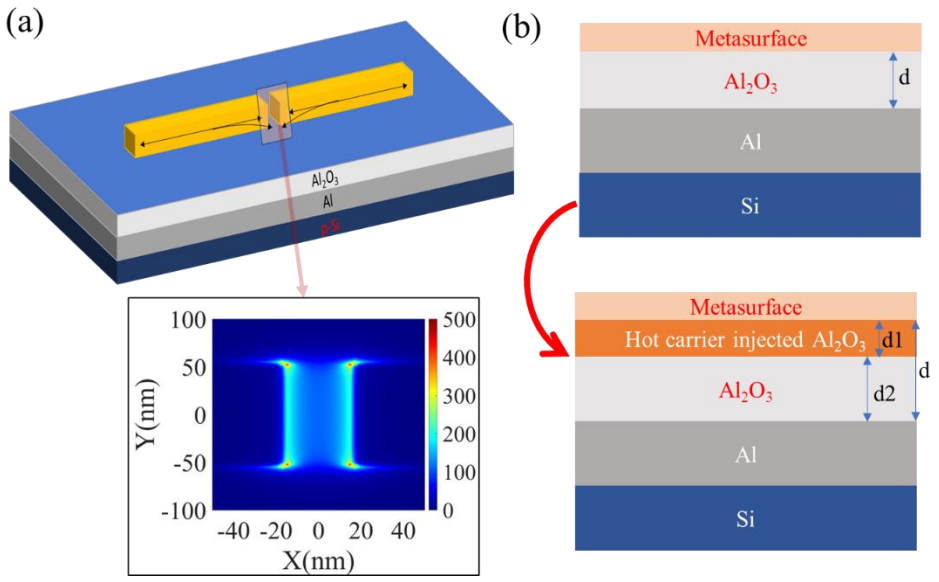


Figure 24(b) bottom), which acts as a separate layer because its permittivity is changed due to the influx of hot carriers (according to the Drude model, $\varepsilon(\omega) = \varepsilon_\infty - \frac{{\omega_P}^2}{\omega(\omega+1i*\gamma)}$). This causes a redshift of the device reflection spectra, as shown in the simulated spectra in Figure 25(a).

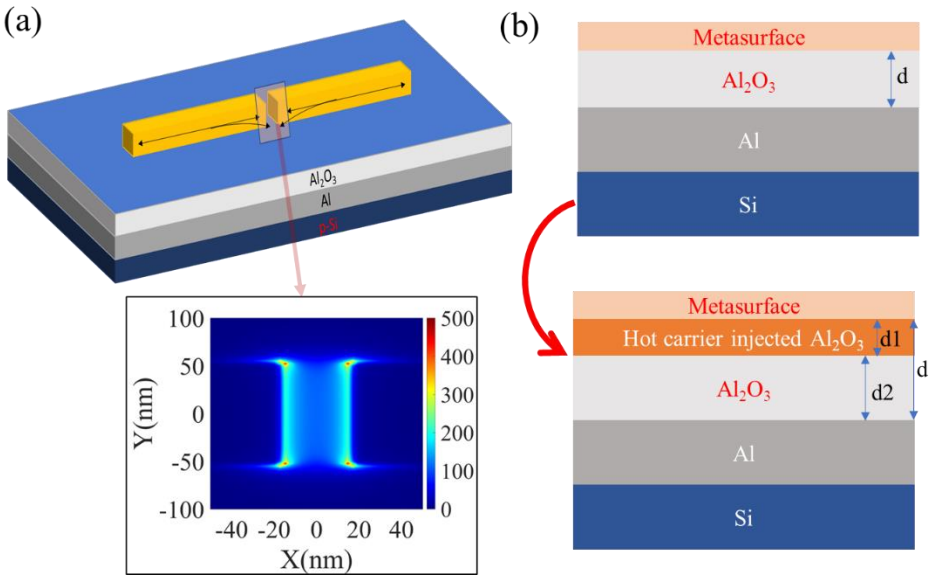


Figure 24: (a) Schematic of Hybrid Metasurface Design. Inset: Off-resonance near Field Enhancement (b) Vertical Cross-sectional View of the Device Showing Metasurface and on the Bottom Figure Shows a Hot Carrier Doped Layer of Al_2O_3.

The simulation is done by solving the multireflection model with metasurface conductivity presented in section 2.3 and considering the change of permittivity of the Al_2O_3 layer obtained by the Drude model. First, note that the spectral region on the left of the resonance, where we can see saturable asborbption[69], is not affected. Figure 25(a) shows the two different spectral regions in two different colors. Second, a lot of ultrafast photoexcited carriers are generated in the graphene monolayer, which instantaneously changes its surface conductivity around the laser bandwidth, so the device reflection spectra also change instantaneously. Here, the change of graphene surface conductivity is dependent upon incident pump fluence so that we can see different reflection modulation at different pump fluences, Figure 25(b). The combination of these two effects produces the reverse saturable absorption behavior of graphene plasmonic hybrid metasurface at longer than resonance wavelengths, shown as decreasing reflection modulation, ΔR = R(reflection at 1.035μm with pump) - R_0(reflection at 1.035μm without pump), in Figure 25(c).

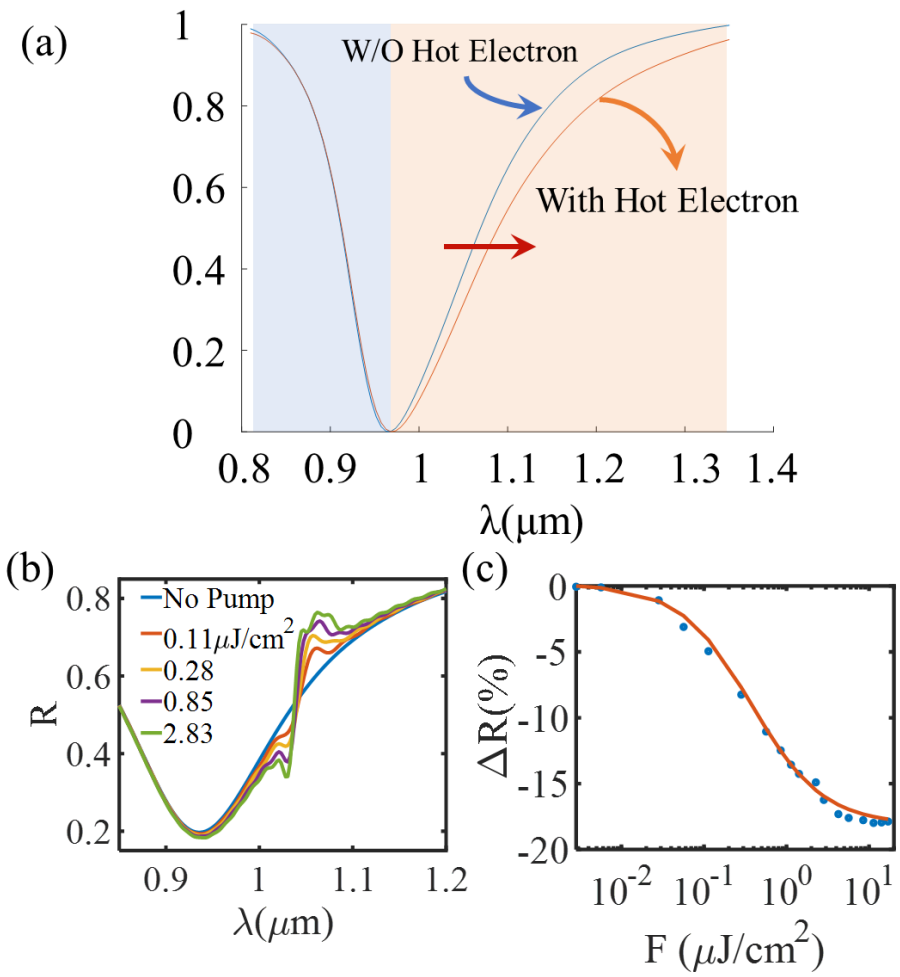


Figure 25: (a) Simulated Reflection Spectra While Considering the Hot Electron Injection and No Hot Electron Contribution. (b) Reflection Spectra Modulation at Different Incident Pump Fluences (c) Reflection Modulation at Increasing Pump Fluence at the Laser Wavelength.

The hybrid metasurface's reverse saturation behavior, i.e changes in absorption with increasing pump fluence, was measured with a femtosecond Ytterbium fiber laser (Menlo Systems) at 1.035μm, having a pulse width of 100fs and repetition rate 100MHz. The laser light is focused on the device (150μm diameter) at 45˚ with an 8" parabolic mirror. The measurement results are presented in Figure 26(b).

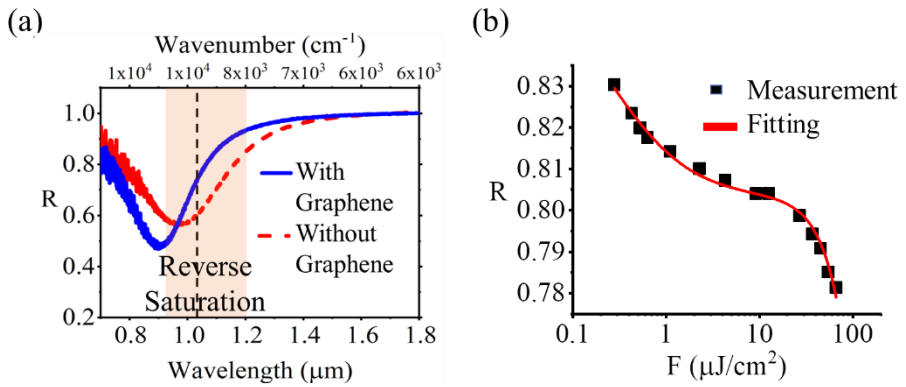


Figure 26: (a) Measured Reflection Spectra of the Device. The Region Indicated with Faded Brown Color Is the Reverse SA Region. (b) the Device Absorption Changes with Increasing Pump Fluence.

We followed equation (23), and from the fitting, we found the reverse saturation fluence of the device is ~0.25μJ/cm^2. We have done more measurements on this type of device, presented in Figure 27, all showing reverse SA behavior.

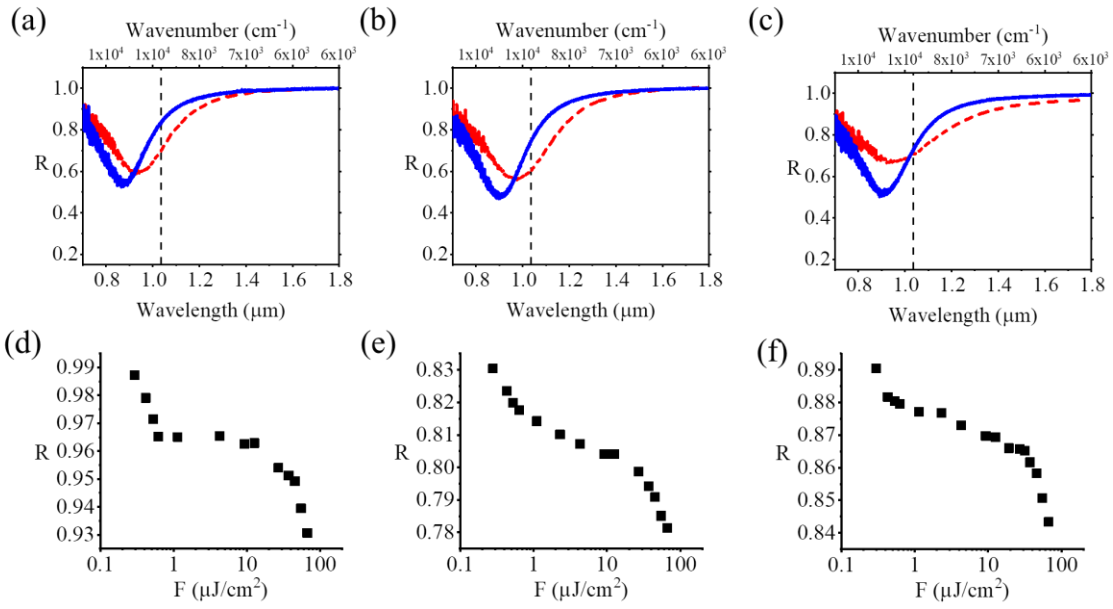


Figure 27: (Reverse) Saturable Absorption Measurement for 3 Different Hybrid Metasurface Devices with Resonance Smaller than the Laser Wavelength (1.035μm).

The degenerate pump-probe measurement (at 1.035μm) with the same laser on one of or devices having resonance smaller than the laser wavelength, Figure 28(a), shows ultrafast behavior of the RSA, Figure 28(b). A model describing the instantaneous modulation of graphene surface conductivity considering the ultrafast carrier accumulation upon laser pulse incidence and subsequent relaxation of these photocarriers due to electron-electron and electron-phonon scattering in graphene[69] provides a better fit with the resulting pump-probe measurement signal for a photoexcited carrier relaxation time of ~50fs. So, the recovery time of the RSA is ~50fs.

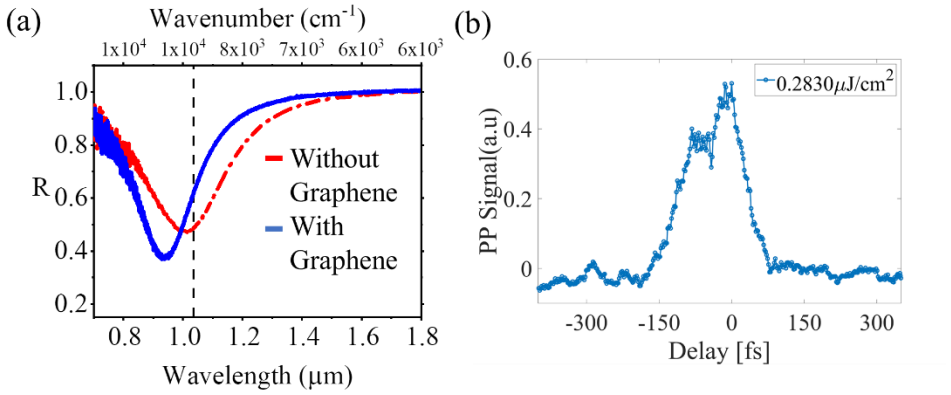


Figure 28: (a) Reflection Spectra of a Hybrid Metasurface Having Resonance Smaller than the Laser Wavelength (1.035μm). (b) Measured Pump-probe Signal For .283μj/Cm2 Incident Pump Fluence Showing Ultrafast Relaxation of the Hybrid Metasurface after the Light Excitation Passes.

3.9 Summary

In summary, we have successfully demonstrated a graphene-plasmonic hybrid metasurface Saturable Absorber device design, achieving a very low saturation fluence of 0.15μJ/cm^2. Furthermore, it demonstrates an ultrafast recovery time (<60fs), a modulation depth of

20%, and a large damage threshold (72μJ/cm^2). To the best of our knowledge, this is one of the best performing saturable absorbers found in the literature. We are confident that this type of device design can pave the way for ultracompact compact, high power, low threshold self-starting passive mode-locked lasers with very small operating power. Also, we have demonstrated reverse saturation behavior with ultrafast response time (~50fs) and low pump fluence ~0.25μJ/cm^2 with a subwavelength (~λ/5) thick graphene plasmonic hybrid metasurface device. This device design can pave the way for future developments in ultrafast low-threshold optical limiters, switch or pulse shaper for ultrafast photonic integrated circuits.

CHAPTER 4

GRAPHENE PLASMONIC HYBRID METASURFACE ALL-OPTICAL MODULATOR (GPHM) IN NIR AND MIR

All-optical modulation is the next-generation solution to ultrafast communication with no limitation set by the bandwidth of the electrical modulation circuitry. In this part of my dissertation, a near-infrared (NIR) pump is used to control the amplitude modulation of the mid-infrared (MIR) laser utilizing graphene-plasmonic hybrid metasurface (GPHM) and demonstrated high modulation speed ultrafast carrier dynamics of monolayer graphene with low pump fluence thanks to high near-field enhancement in GPHM. With our hybrid metasurface design, it is possible to achieve 40% reflection modulation at 6.5μm with ~55μJ/cm^2 pump fluence and ultrafast relaxation time of ~1ps at 1.56μm with less than 8μJ/cm^2 pump fluence. This is one of the first demonstrations of all-optical modulation in the MIR wavelength range beyond 6μm.

4.1 Introduction

Modulating signals, either electrical or optical, is essential for data processing. Nowadays, optical modulators[47] are getting a lot of attention because of their capability of achieving high speed and low power requirements for high-speed optical signal processing[48] and communication[49], optical interconnects[3], ultrafast pulse generation[50], quantum information, ultrafast spectroscopy[51] etc. There are many different methods to modulate optical signals, such as electro-optic modulators, acousto-optic modulators etc. For electro-optic modulators, optical signal is modulated utilizing traditional techniques such as

refractive index modulation of the material by applying an electric field (Pockels effect). However, this technique is dependent on an electrical circuit that usually determines the speed of the modulation of the optical signal. So, the optical signal modulation speed is limited by the capacitive effect of the electrical circuit (time constant). There is another type of novel optical modulation technique known as all-optical modulator. For all-optical modulators, the optical signal is modulated by another optical signal, not electrical. So, the latency imposed by the speed of the electrical circuit used to control the electro-optic modulator is not present here. Consequently, it is possible to operate all-optical modulators in ultrafast manner in THz or higher speed instead of the GHz or less speed achieved by electrically controlled optical modulators.

There are many all-optical modulators reported in the literature in near-infrared (NIR) wavelength ranges[22,24,49,50,52–67]. But due to insufficient nonlinearity in the available materials and lack of proper methods, there are a few demonstrations of all-optical modulation in mid-infrared wavelengths larger than 2μm, all of which are less than 5μm. So, there is no demonstration of all-optical modulation in wavelengths larger than 6μm wavelengths. Most of the all-optical modulators demonstrated are based on 2D materials such as graphene[23,65–67], two-photon absorption[54,59], silicon nanoantenna[52], Tamm-plasmon resonance[68], plasmonic semiconductor nanocrystals[50], 2D array of holes in Si membrane[55] etc. It is challenging to obtain ultra-compact, low pump fluence, and ultrafast all-optical modulators in MIR due to inherent optical absorption and small modulation of conventional materials.

Graphene is a novel 2D material with broadband nonlinear optical behavior, ultrafast carrier dynamics, and it is also compatible with many different types of substrates, including fiber optics. However, the demonstrated all-optical modulators utilizing only graphene in the literature require a large optical pump fluence (>1mJ/cm^2)[49,65–67] mostly due to low optical abruption of light (~2.3%) and ultrashort carrier lifetime. There are different methods to increase the optical absorption, such as increasing the optical interaction length by combining it with waveguide structure[66], increasing light-graphene interaction utilizing metasurface (plasmonic and semiconductor)[22,69], put graphene inside a cavity[70] etc.

In this part of my dissertation, it is demonstrated that with the help of a graphene plasmonic hybrid metasurface design, it is possible to achieve high all-optical modulation depth and ultrafast speed in wavelengths larger than 6μm[38] with very low pump fluence of about 75μJ/cm^2. The hybrid metasurface consists of a novel plasmonic metasurface of closely coupled pi (π) shaped antenna and monolayer graphene. The plasmonic metasurface creates nanoscale hot spots with high near-field enhancements when light is incident on the device. The hybrid metasurface is a double resonant device having one resonance in the MIR and the second resonance in NIR wavelengths. Due to this double resonance behavior of this device, it is possible to enhance the near field for both pump (NIR) and probe (MIR) wavelengths by a few orders of magnitude compared to the incident light electric field. Due to this effect, the pump photon absorption is highly boosted, and we can achieve larger modulation in the MIR wavelengths (probe beam) with very low pump fluence in the NIR wavelengths. NIR all-optical modulation is also demonstrated with this device design with very low pump fluence of 7.9μJ/cm^2. For optical signal modulation in MIR and NIR

wavelength, the required pump fluence for our device is more than two orders of magnitude lower than all-optical modulators reported in literature. Ultrafast pump-probe measurements in the NIR wavelengths (780 pump and 1560 probe) illustrated that the device conserves it ultrafast behavior of the graphene carrier dynamics and thus resulting in ultrafast response of our device in the order of 1-2ps range. In this project, I helped build the optical setup, participated in the fabrication of the device, did measurements and data analysis.

4.2 Device Design

The device design was conducted by Basiri et al. [22,38] for all-optical modulation. The device design is well documented in these published papers. Here, I have included some of the important points of the device design here as an introduction to this chapter.

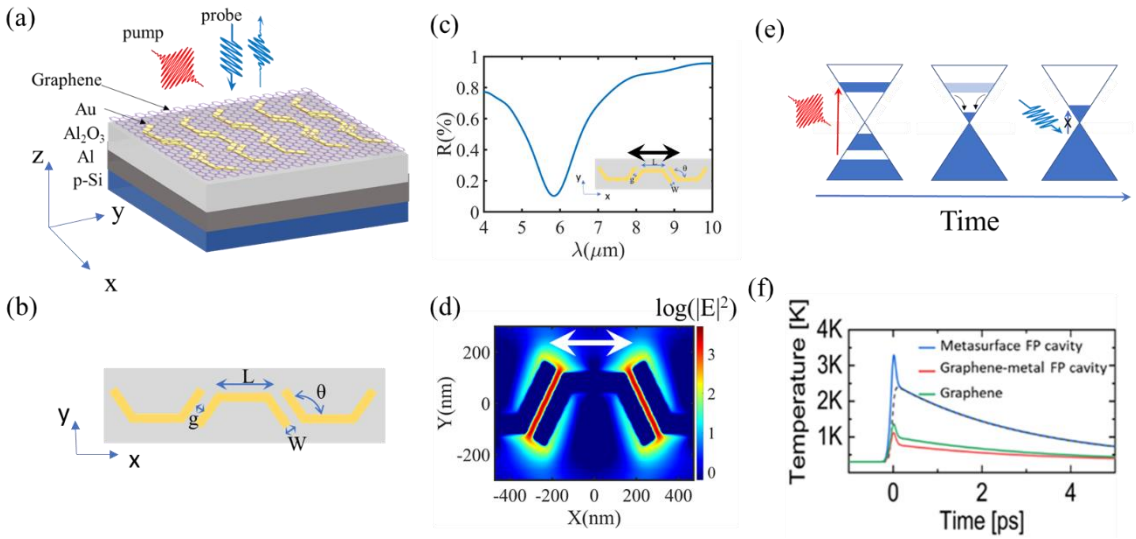


Figure 29: (a) All-Optical Modulator Design in Mid-IR Utilizing the Metuasrface with Pi Shape Nanoantenna Structure. (b) Top View of the Device Design Showing the Pi Shape Antenna Metasurface Close View. (c) Reflection Spectra of the Design Showing

Resonance Around 6μm. (d) Near-field Enhancement Inside the Nano-gap of Metasurface. (e) Graphene Carrier Dynamics as the Device Is Hit by the Pump (at Higher Energy) and Then Lower Probe in Mid-IR. (f) Increase in Electronic Temperature of Graphene (in Different Structures) after Laser Excitation of the Device[22,38].

Figure 29(a) shows the hybrid metasurface design incorporating the plasmonic metasurface and graphene. Figure 29(b) shows the top view of the hybrid structure showing the pi-shaped nano-antenna structure. In the figure, different parameters can be used to modify the surface conductivity of the metasurface. This hybrid metasurface is designed to have very low reflection, around 6μm, by satisfying perfect absorption conditions derived from the modified Fresnel's equation in[37]. A brief description can be found in section 2.2.1. A representative reflection spectrum is presented in Figure 29(c). Figure 29(d) shows the high near-field enhancement inside the metasurface nano-gap.

We are utilizing a near-IR pump in this all-optical modulation scheme to modulate the Mid-IR probe light. The near IR pump excites the carriers from the valence band (VB) to the conduction band (CB). The subsequent ultrafast carrier dynamics is shown in Figure 29(e). After the pump hits the device, a lot of carriers (within the energy band of the pump) are excited from VB to CB. This happens within less than 10fs[11]. These excited carriers form a non-equilibrium carrier distribution that gets thermalized within 10-100fs due to electron-electron (e-e) and electron-optical phonon (e-ph) scattering of the excited carriers. As the carriers thermalize from the non-equilibrium carrier distribution, a hot Fermi-Dirac carrier distribution forms near the band edge of VB and CB. At this time, incoming MIR-IR light will not be able to excite any carriers from VB to CB because the states are occupied in

CB. So, the Mid-IR laser will be modulated at this time. After that, different Intra and inter-band phonon-assisted scattering and other processes further cools the thermalized carriers to lattice within 1-2ps.

As the equilibrium hot Fermi-Dirac distribution is formed (Figure 29(e)), the electronic temperature of the carrier distribution is gradually increased (illustrated in Figure 29(f)). Due to this increase in electronic temperature, optical surface conductivity will change also. In this case, graphene sheet optical conductivity can be derived within the random phase approximation in the local limit[8,130],

$$\sigma(\omega) = \sigma^{\text{intra}}(\omega) + \sigma^{\text{inter}}(\omega)$$
$$= \frac{2ie^2Tk_B}{\pi\hbar(\omega + i\tau^{-1})} ln[2\,cosh(\mu/2T/k_B)] + \left(\frac{e^2}{4\hbar}\left[\frac{1}{\pi}\,tan^{-1}\left(\frac{\hbar\omega - 2E_F}{2k_BT}\right)\right.\right. \quad (24)$$
$$\left.\left. - \frac{i}{2\pi}\,ln\left(\frac{(\hbar\omega + 2E_F)^2}{(\hbar\omega - 2E_F)^2 + 4(k_BT)^2}\right)\right]\right)$$

Although surface conductivity has contribution from both intraband and interband carriers, we are mostly interested in intraband contribution in our calculations as the hot Fermi-Dirac distribution is formed due to intraband carrier interactions.

Figure 30 (a-b) illustrates the change in graphene surface conductivity as the electronic temperature is changed from 300k to 4600k. This is extracted from the Lumerical 2D graphene model at different electronic temperatures. We can calculate the in-plane (parallel) component of graphene permittivity from the surface conductivity of graphene using equation (24). Figure 30 (d-e) shows the change in graphene permittivity utilizing this equation. Due to this effect, the plasmonic nano-antenna resonance will blue shift, as

illustrated in Figure 31(a). In Figure 31(b), we can see the expected reflection modulation at different laser wavelengths.

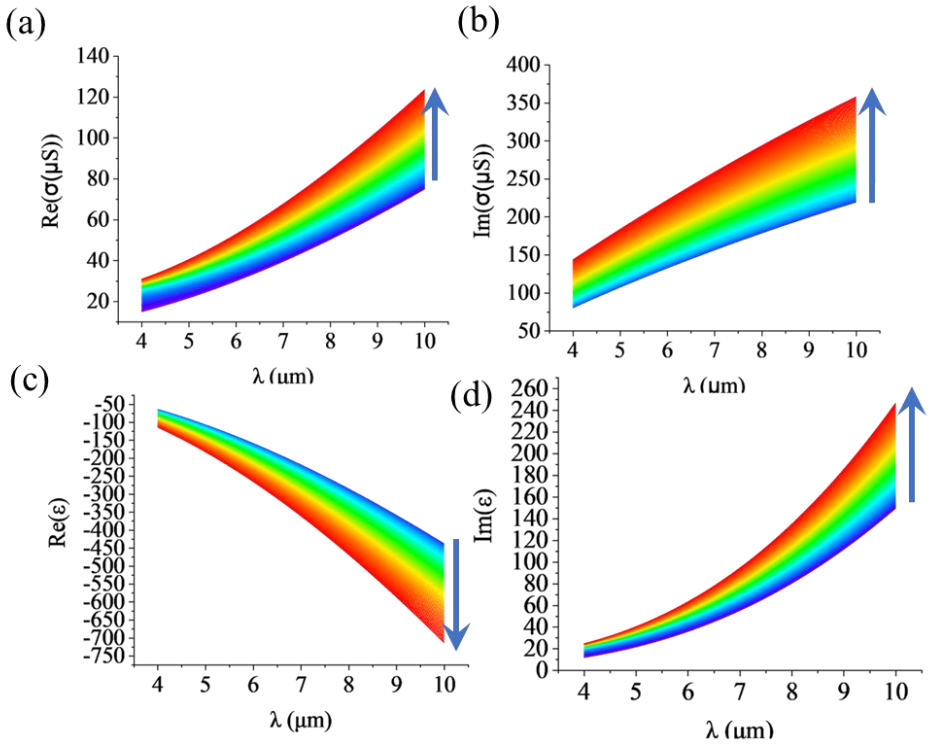


Figure 30: Graphene Surface Conductivity (a) Real Part (b) Imaginary Part (Extracted from Lumerical, 2D Graphene Model at Different Electronic Temperature). In-Plane Component of Graphene Permittivity (c) Real Part (d) Imaginary Part. The Direction of Change in Surface Conductivity and Permittivity with Change in Electronic Temperature is Indicated by the Blue Arrow.

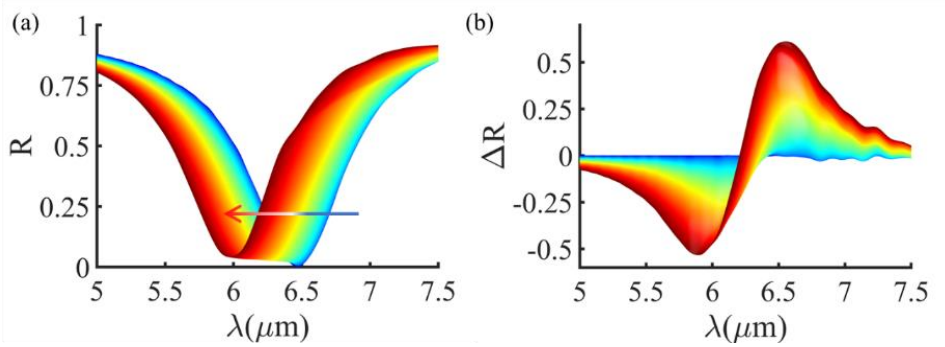


Figure 31: (a) Reflection Spectra of the Nano-antenna Will Blue Shift as the Excited Carriers Form Fermi-dirac Distribution and the Electronic Temperature Increased. (b) Corresponding Reflection Modulation as the Resonance Is Blue Shifted.

4.3 Device Fabrication and Optical Characterizations

Figure 32 (a-d) illustrates the fabrication process of the hybrid metasurface. The fabrication process is very similar to what we discussed in section 4.1 and 4.2. Figure 32(e) shows the SEM image of the plasmonic metasurface before the graphene transfer. Figure 32(f) shows the Raman spectrum of the graphene after transferring on top of plasmonic metasurface.

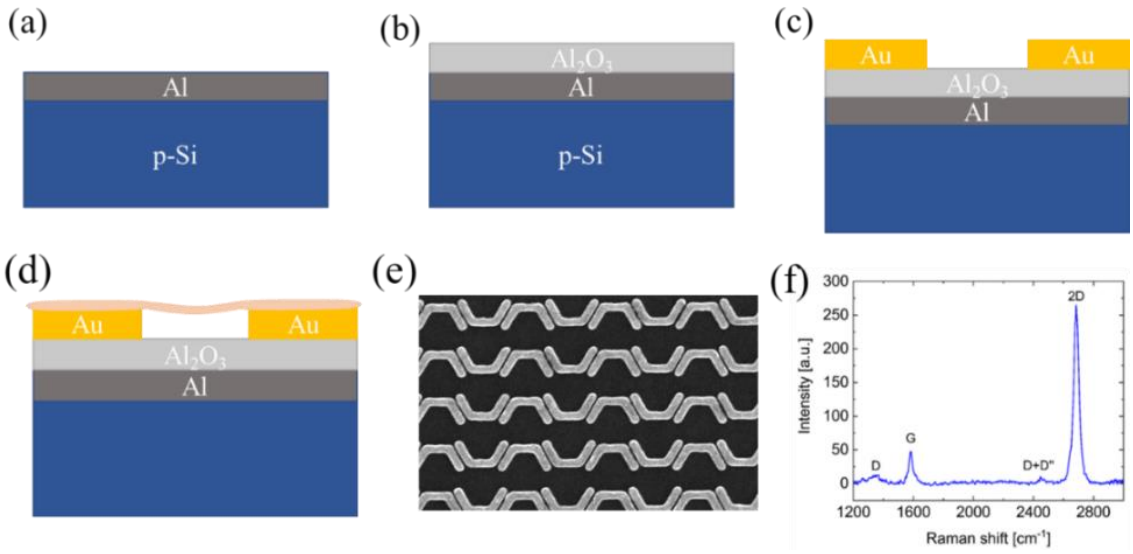


Figure 32: (a) Electron Beam Evaporation of Al to Deposit It on Si Substrate. (b) Al_2O_3 Is Deposited Using Atomic Layer Deposition. (c) Electron Beam Lithography with Double Layer Resist (Pmma/Mma), Development, Plasma Etching, Cr/Au Deposition, and Lift-ff to Make the Plasmonic Metasurface. (d) Graphene Transfer on Top of the Metasurface. (e) Scanning Electron Microscopy Image of the Pi Shaped Plasmonic Metasurface (f) Raman Spectra of the Graphene Layer after Transferring It on the Plasmonic Metasurface.[38]

Reflection spectra of the device was measured using FTIR (Courtesy: Ali), illustrated in Figure 33(a). Reflection spectra in MIR and NIR is presented in Figure 33(b) and (c).

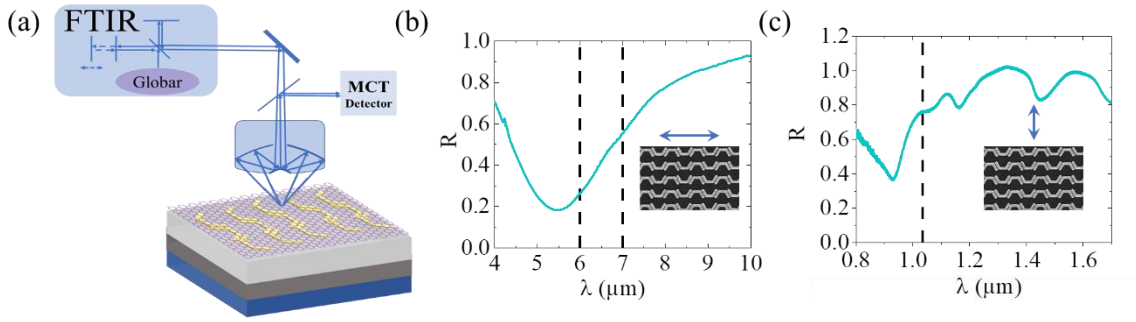


Figure 33: (a) FTIR Setup for Characterizing the Reflection Spectra of Our Device at Different Wavelengths. (b) Reflection Spectra of Our Hybrid Metasurface in Mid-IR (c) Reflection Spectra of the Hybrid Metasurface in NIR.[38]

4.4 All-Optical Modulation Measurement

For the first all-optical modulation measurement, we used the setup illustrated in Figure 34. The setup includes a NIR pump working at 1.035μm. Modulation is observed using a continuous wave MIR QC-laser at 6.31μm. After the MIR laser is reflected from the device through the objective lens, it is directed towards a power meter to measure the reflected power of MIR laser. By dividing this with the MIR reflected power from the substrate, we can calculate the reflection of the device. And as we change the incident NIR pump power, we should see changes in the MIR reflected power (i.e., the reflection of the device). This optical modulation is illustrated in Figure 35 for different devices having resonances at different wavelengths. For these devices, maximum reflection modulation is about 4%.

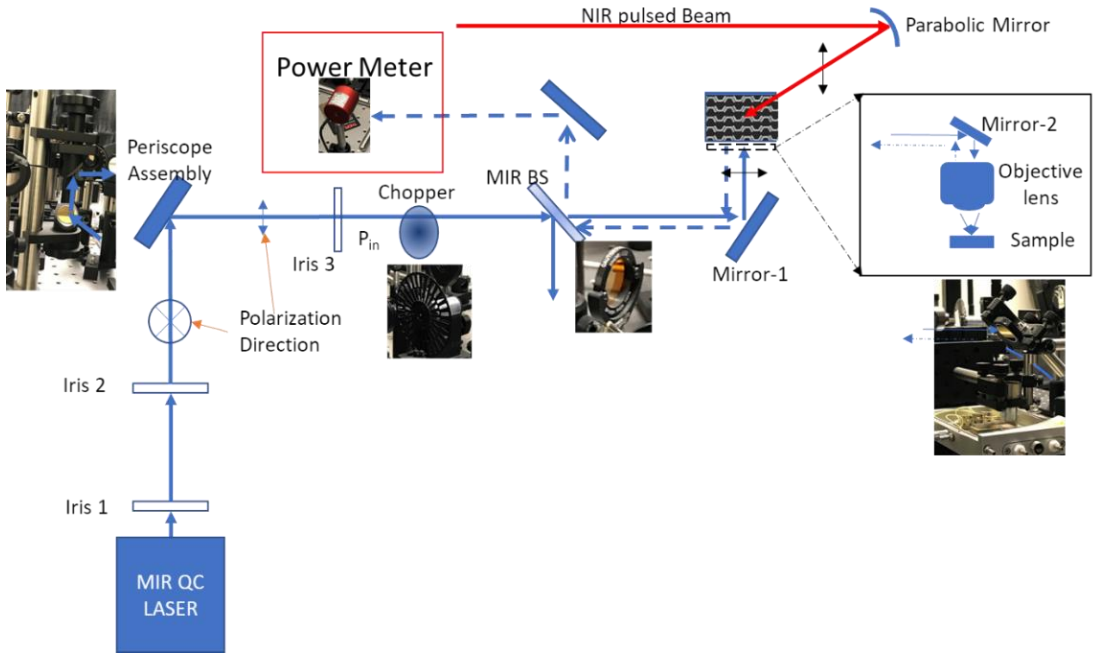


Figure 34: All-Optical Measurement Setup with Power Meter. NIR Pump Is at Wavelength 1.035μm, and MIR Probe Beam Is at 6.31μm

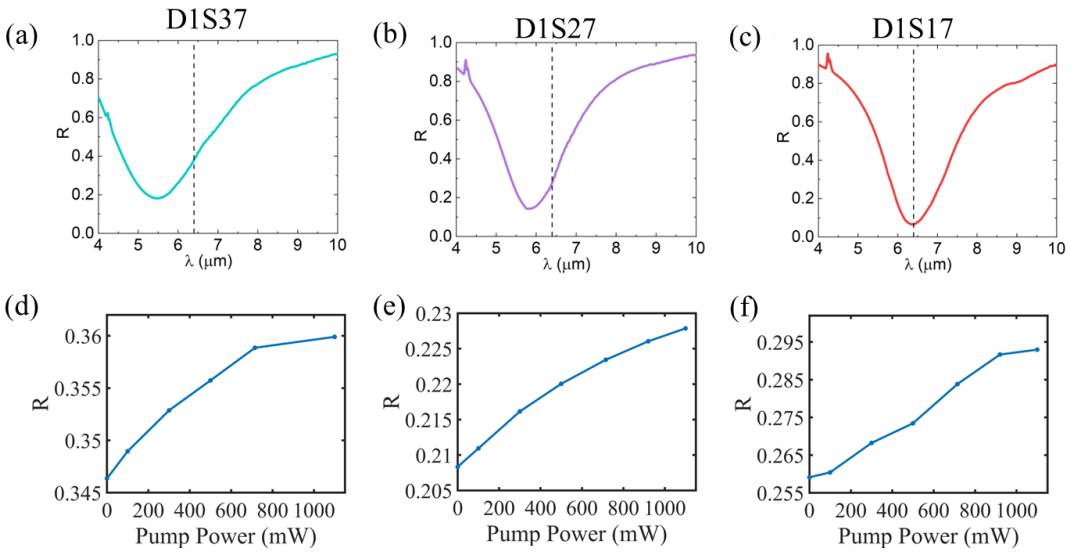


Figure 35: (a-c) MIR Reflection Spectra of the Devices under All-Optical Modulation Measurement (d-f) Change of the Device Reflection at 6.31um upon Pump Excitation at Different Incident Power.

As discussed in section 5.1, after the pump laser excited the carriers from VB to CB the excited carriers relax to lower energy band (it is also possible that the scattering process takes the electrons in higher energies from the pump) and forms a hot Fermi-Dirac distribution near the CB and VB band edge. This blocks the absorption of MIR laser. As a result, the reflected MIR laser becomes modulated or it forms a pulse shape according to the time resolution of the ultrafast carrier relaxation time in graphene. We can also observe the pulse shape and pulse formation using a photodetector. Figure 36 illustrates the measurement setup used to see this MIR pulse formation from the CW MIR laser wave. Here we use a fast MIR photodetector with a large bandwidth (~1GHz).

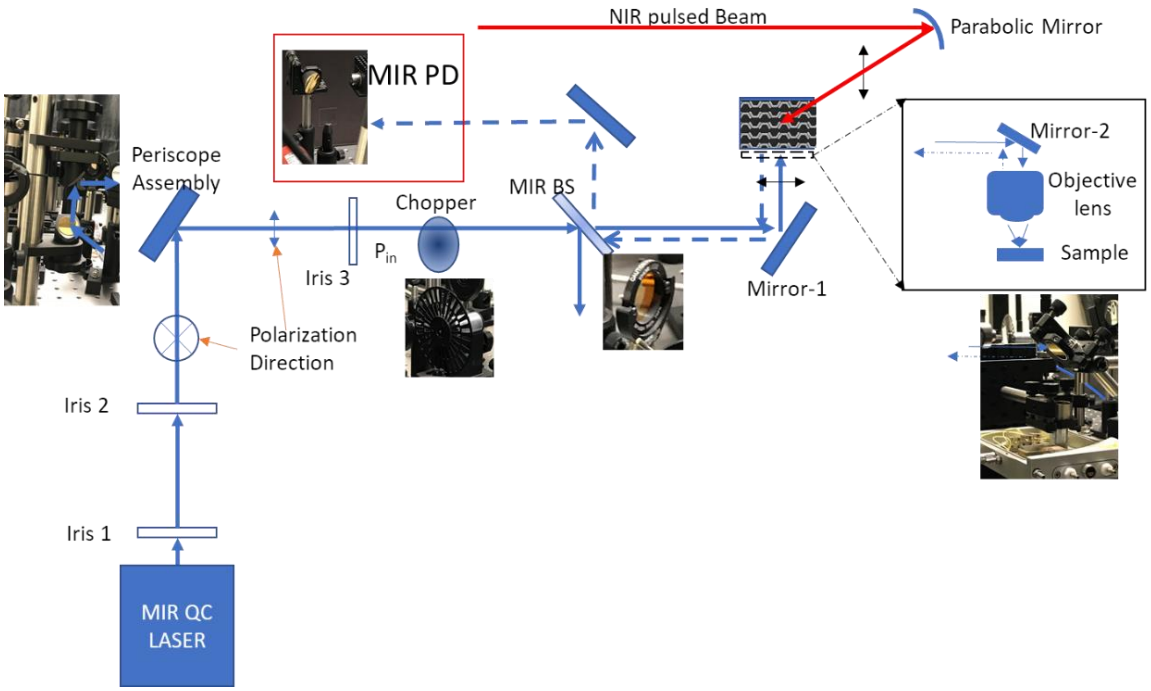


Figure 36: All Optical Measurement Setup with MIR Photodetector. NIR Pump Is at Wavelength 1.035μm and Mir Probe Beam Is at 6.31μm

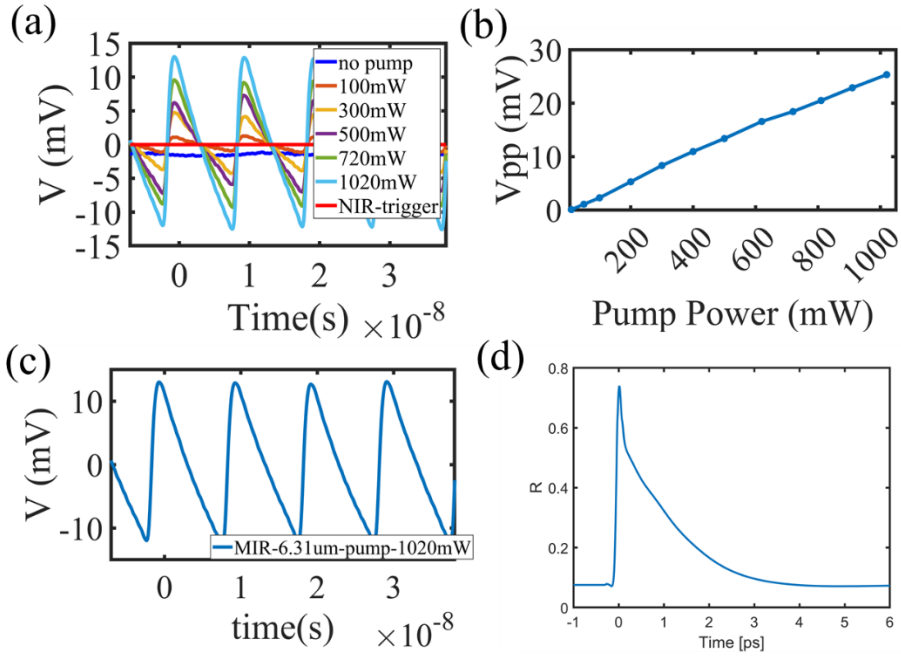


Figure 37: (a) MIR Light Modulation and Pulse Formation from CW Mode MIR Laser with NIR Pump. (b) Change in the Peak-to-Peak Value of the Pulses as We Increase the NIR Laser Power. (c) MIR Pulses Formed Due to NIR Pumping at 1.034um with 1020mw Incident Power. (d) Simulated MIR Pulse with Smaller Relaxation Time and Duty Cycle and Larger Amplitude.

Figure 37(a) illustrates the detected reflected MIR laser on the MIR photodetector (through oscilloscope) as we increase the NIR pump power from no pump to 1020mW. When there is no NIR pump, we can see that the photodetector detects the MIR laser as a constant value. When we turn on the pump at 100mW, we immediately see a pulse shape is formed. And as we increase the pump power, the pulse shape amplitude increases. We can plot the peak-to-peak value of the detected voltage (MIR pulse) on the oscilloscope. Figure 37(b), shows that as we increase the NIR pump power, the MIR pulse amplitude increases, and its bending at a large incident NIR pump. Figure 37(c) shows the MIR pulse that is formed

due to the 1020mW NIR pump. In simulation, we can also estimate the expected pulse shape, illustrated in Figure 37(d). We can immediately see that the pulse shape predicted by simulation has a smaller time constant than the detected pulse in the oscilloscope. This can be explained by the parasitic RC circuit, which modifies the signal time constant as it travels from the photodetector to the oscilloscope. This process is illustrated with representative images in Figure 38.

Figure 38(a), shows the parasitic RC circuits, from photodetector response time and oscilloscope, on the way of the pulse signal coming from the photodetector toward the oscilloscope. There are other contributions from the cable that we use, type of connector used to connect the cable with oscilloscope and photodetector etc. All of these adds up and broaden the pulse. If we assume that there is no energy loss in this process, then its amplitude also gets smaller as the pulse gets broadened. This is illustrated in Figure 38(b).

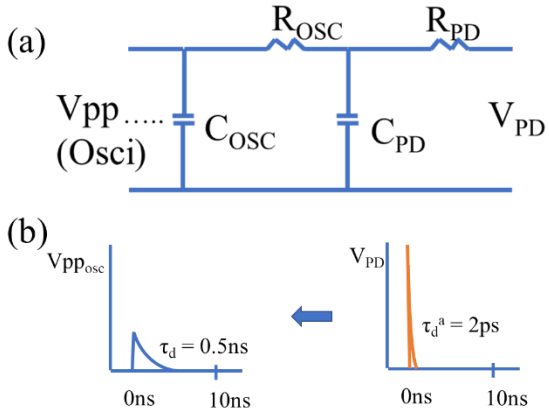


Figure 38: (a) RC Circuit Build up along the Path from Photodetector and the Oscilloscope. (b) Representative Picture of How the Pulse at Photodetector End Will Broaden and Shrink in Amplitude along the Way

Figure 37(a), shows the pulse modulation with NIR pump power, but we don't know the reflection modulation of the device. To obtain the reflection modulation, we utilize the chopper to chopper the MIR CW laser to obtain the R_0, or the reflection without any pump. Utilizing the following equation (please refer to Figure 36 to make sense of these equations)

Without pump (chopper on):

$Vpp0 = \Delta Tchopper\ (Pin \times T1)\ (R_{sub} \times T2 \times Responsivity)$

With pump (chopper stopped):

$Vpp = (Pin\ T1)\ \Delta R_T\ T2\ \times (Responsivity)$

Here, T = optical transmission considering all the optical components and environmental effect; $T1 = T_{MIR\_BS}\ R_{mirror\text{-}1}\ R_{mirror\text{-}2}\ T_{obj}$; $T2 = T_{obj}\ R_{mirror\text{-}2}\ R_{mirror\text{-}1}\ R_{MIR\_BS}$; $\Delta Tchopper$= Maximum transmission through the chopper =Tmax - Tmin ~=97%; Rsub = reflectivity of substrate = ~98% and ΔR_T = reflection modulation of the sample

We chop the MIR laser with the optical chopper at 1kHz. Also, for some cases, we reduce the MIR laser power to reduce the photodetector saturation. The detected MIR chopped pulses is presented in Figure 39. To calculate the R0 or the base reflection from the sample, we do the measurements presented in Figure 39 (a, b). From the peak-to-peak value obtained from Figure 39 (a, b), we can calculate R0.

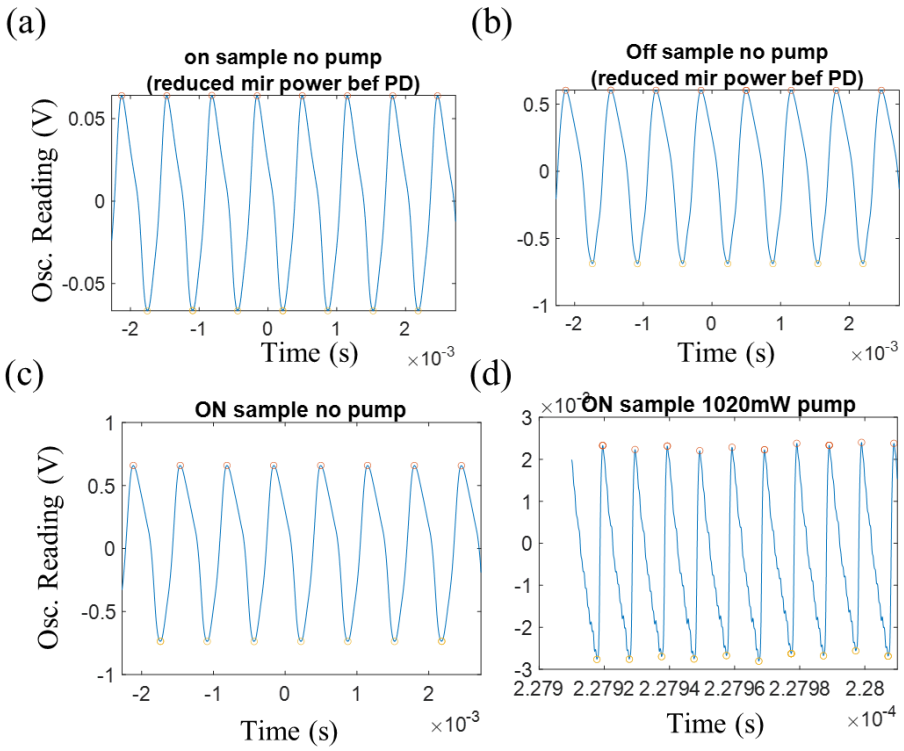


Figure 39: (a) Chopped MIR Reflected Pulse from the Sample. No Pump. MIR Power Reduced to Reduce Saturation of the Photodetector (b) Chopped MIR Pulse Reflected from Substrate. No Pump. MIR Power Reduced to Reduce Saturation of the Photodetector (c) Chopped MIR Pulse Reflected from the Sample. No Pump. MIR Power Is the Same as Used in (d) for with Pump Condition. (d) Reflected MIR Pulse When We Have NIR Pump. MIR Beam is Not Chopped.

Then we can calculate the reflection modulation using,

$$Vpp/Vpp0\_sub = m(\Delta R_T/Rsub)\ (1/\ \Delta Tchopper)$$

$$\Delta R_T = (Vpp/Vpp0\_sub)\ (\Delta Tchopper)\ (Rsub)\ m \qquad (25)$$

Here, m=Area under curve for simulation/ Area under curve for measurement.

The factor m is calculated from Figure 40 by dividing the area under the curve for the simulation and the area under the curve for measurement. This factor is to compensate for the reduced amplitude of the detected signal due to pulse broadening by the RC circuit.

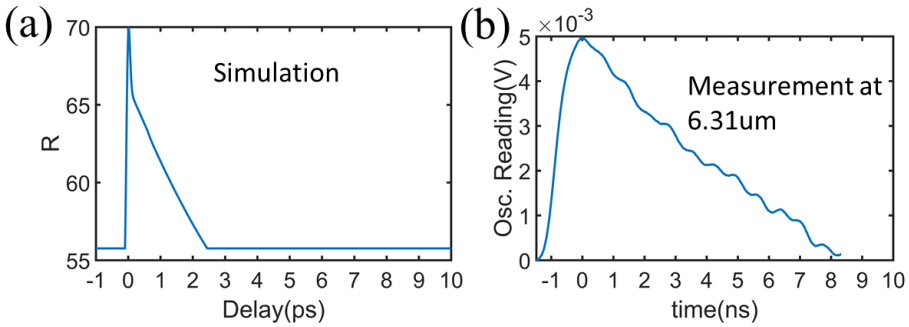


Figure 40: (a) Simulated Expected Reflection with Pump at 6.31um (b) Measured Reflection from the Oscilloscope

To calculate the ΔR_T, we need to measure the Vpp0_sub, which is the reflected peak-to-peak voltage from the substrate at the same MIR laser power used to measure the modulation with NIR pump. This reflected MIR power is too large for the photodetector and gets saturated. So, we go around this. First, we measure the chopped MIR laser reflected from the sample as the reflected MIR power is less in this case, and the photodetector doesn't get saturated (Figure 39 (c)). And by multiplying this peak-to-peak value by R0, we can get the peak-to-peak voltage for the reflected MIR laser from the substrate (Vpp0_sub). Then we do the measurement with the pump on the sample, stopping the chopper and using a continuous wave MIR laser (Figure 39(d)). From here, we can calculate the Vpp and, using equation (28), calculate the ΔR for the device at the NIR pump power. This was done at different MIR laser wavelengths, presented in Figure 41.

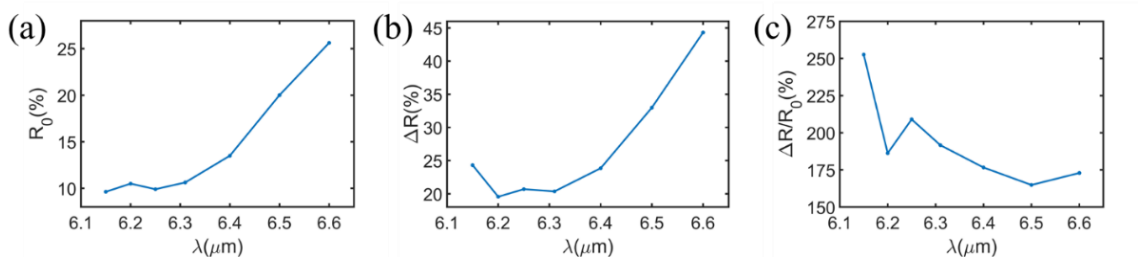


Figure 41: (a) R0 Measured on Sample S37 (Dose1) (b) Reflection Modulation Calculated from the Measurement (c) Normalized Reflection for the Same Sample.

Figure 41(a) shows the R0 measured at different MIR laser wavelengths. Figure 41(b) is the reflection modulation at different wavelength, calculated using the methods described earlier. And normalized reflection modulation is presented in Figure 41(c). The results closely match the simulated reflection modulation in Figure 31(b).

In Figure 31(b), we have seen that we expect different modulation directions for different probe wavelengths. So, if we could probe with MIR laser with wavelengths falling on the left side of the device resonance, we would see a negative modulation of the signal, or we would see that the reflection is going down with NIR pump. Unfortunately, we don't have any laser having a wavelength less than 6μm. To go around this problem, we have selected some devices with resonances on the right side of our laser wavelength (6.31μm). And if we do the same measurement, we should see the reflection going down, or the formed pulse will go down only. These measurement results are presented in Figure 42(a, b). Here, we can see that for the devices with larger resonances than the probe wavelengths, the modulated pulses go down instead of up, indicating negative modulation of the pulse. Figure 42(c) shows results for a device similar to the device we explored earlier, having

resonance on the left of the probe wavelength. Here we see the positive modulation of the signal, as expected.

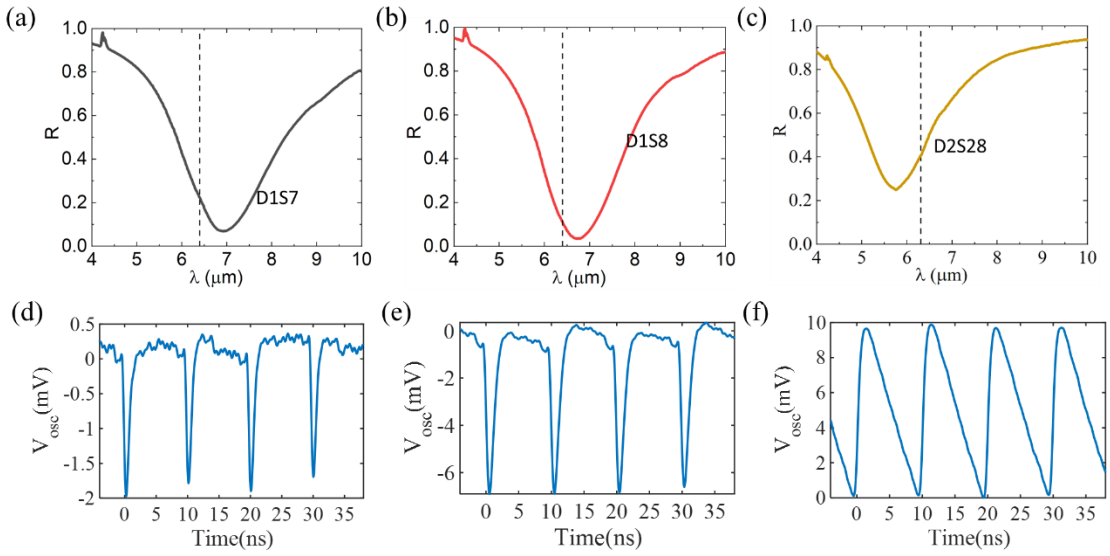


Figure 42: (a-c) Device Reflection Spectra in MIR Wavelength Range. (d-f) Corresponding Signal Modulation for the Devices in (a-c). All of These are Obtained With 57μJ/cm^2 NIR Pump Fluence.

So, it is possible to change the all-optical modulation direction by changing the probe wavelength position compared to the resonance of the device in MIR.

4.5 Ultrafast Pump-Probe Measurement in NIR

To measure the true time resolution of our device, we need to perform ultrafast pump-probe measurements with ultrafast MIR lasers. Unfortunately, we don't have access to ultrafast MIR lasers in our lab. To get some idea about our device's carrier dynamics and relaxation time, we performed pump-probe measurements with NIR pump at 780nm and a probe at 1560nm (100MHz repetition rate, 100fs pulse width). 1560nm does not really fall into mid-IR lasers, but it can give us very close results. For the pump to work at 780nm, we need to

have some absorption of light or near-field enhancement at this wavelength. Fortunately, the device design is a miracle, and it shows a reflection of about 60% around 780nm (FTIR measurement in Figure 43(a)) with an incident light polarization along the y-axis of the nano-antenna.

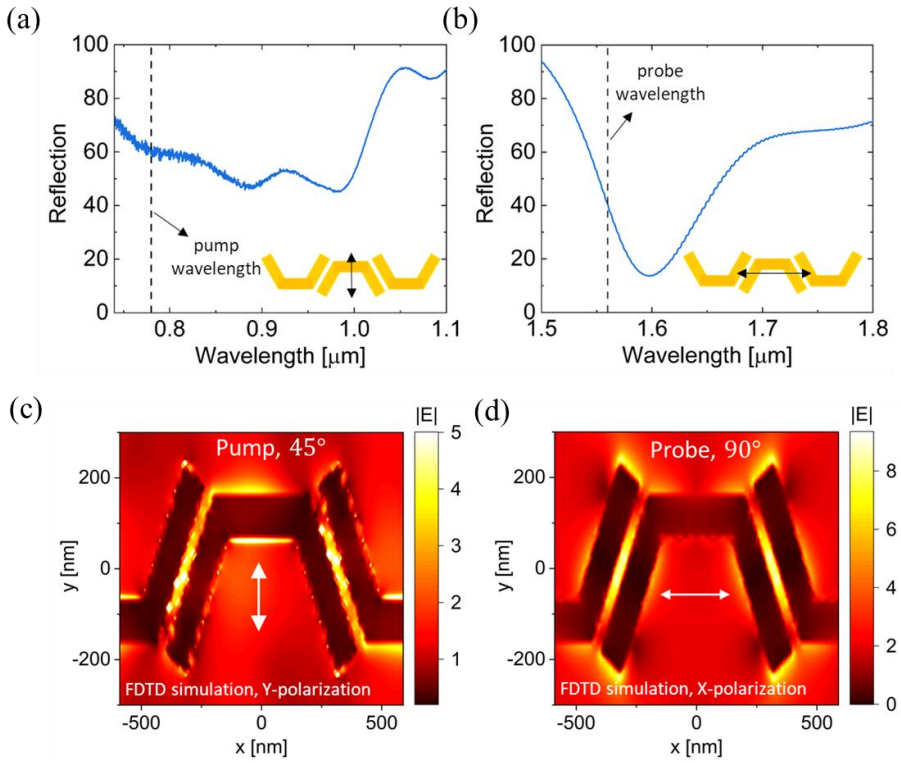


Figure 43: Reflection Spectra of the Device Measured with FTIR (a) from 700 to 1100nm (b) from 1500 to 1800nm with Light Polarization Indicated in Inset with Black Arrow. (c) near Field Enhancement for Light Polarization along the Y-Axis (d) near Field Enhancement for Light Polarization along X-axis[22,38].

Also, at the probe wavelength, 1560nm, we see a smaller reflection of about 40% with the incident light polarization along the x-axis of the nano-antenna (Figure 43(b)). The simulated near-field enhancement for the same conditions is presented for pump wavelength in Figure 43 (c) and probe wavelength in Figure 43(d).

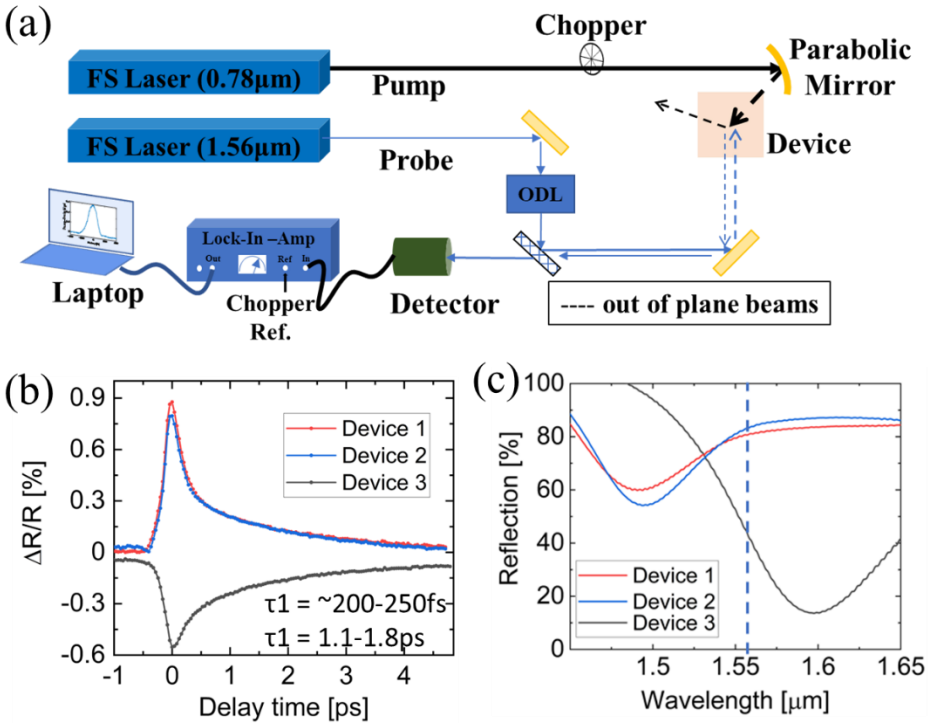


Figure 44: (a) Pump Robe Measurement Setup. Pump at 780nm and Probe at 1560nm. (b) Pump-Probe Results for Different Devices. Negative Modulation Obtained for Device with Resonance on the Left Side of the Probe Wavelength (c) Corresponding Device Reflection Spectra Around the Probe Wavelength (1560nm).

Utilizing this doubly enhanced device, we can do pump-probe measurements and find out the relaxation time of our device. As we have enhancement at the probe wavelengths also, we need to use very small power for the probe wavelength so that it does not have any effect of its own (saturable absorption or self-modulation) on the device.

The pump-probe measurement setup is shown in Figure 44(a). We are using the lock-in-amplifier to increase the signal-to-noise ratio. Figure 44(b) presents the pump-probe measurement results for different devices. The corresponding reflection spectra of the devices are shown in Figure 44(c).

As expected from our previous analysis, we see positive modulation for the devices with resonance on the left of the probe wavelength and negative modulation for the device with resonance at a larger wavelength than the probe (shown in Figure 44(b)). These pump-probe data follow a double exponential decay function that corroborates the carriers' ultrafast decay from the excited state into the hot Fermi-Dirac distribution. By fitting a double exponential function, we can find our device's time constant or relaxation time. The time constants are $\tau 1$=200-250fs and $\tau 2$=1.1-1.8ps. The range in the time constant is coming from measurements with different incident NIR pump power. These time constants match with the reported graphene carrier dynamics time constant in the literature. Also, it indicates that our device works very close to the relaxation time limit set by the ultrafast carrier dynamics of graphene.

4.6 Summary

In summary, all-optical modulation in near-infrared (1560nm) and mid-infrared (6-7μm) is demonstrated in this part of my dissertation with a graphene plasmonic hybrid metasurface modulator. The high optical modulation depth with low pump fluence is achieved due to the simultaneous enhancement of pump and probe beams via this double resonance hybrid metasurface design. The device design can be tuned to work in any wavelength from near-IR to THz range. This is the first demonstration of MIR optical modulation beyond 6μm with such low pump fluence (three orders of magnitude lower than reported works in NIR and 1-2 orders of magnitude lower in MIR range), to the best of our knowledge. This demonstration opens the possibility of developing an ultrafast optical signal processor and all-optical modulator with ultra-low power requirements and an ultra-compact footprint.

CHAPTER 5

GRAPHENE PLASMONIC HYBRID METASURFACE SATURABLE ABSORBER IN MID-IR

Saturable absorption beyond 6μm is desirable because it can help us build ultrafast mode-locked lasers in MIR wavelength. Currently, a device based on GPHM to achieve saturable absorption in MIR wavelengths is under development. Some of the initial measurements and plan for future measurements to find the working principle and the characteristics of the MIR saturable absorption in GPHM is demonstrated.

5.1 Introduction

Saturable absorbers are used as a passive optical modulator to produce ultrafast and ultrashort laser pulses[39,131]. There are a lot of ultrafast lasers with ultrashort pulses in the near-IR wavelengths but there is no commercially available ultrafast ultrashort pulse laser in mid-IR wavelength due to a lack of available materials having large nonlinearity in the mid-IR wavelength range. In this chapter, we talk about mid-IR saturable absorbers and the device characterization for an initial understanding of the working principle.

Graphene is a good candidate for mid-IR saturable absorber due to its broadband optical absorption, which can be saturated with high-intensity pumping. However, the absorption of light is very small (~2.3%) to produce a saturable absorption effect with low saturation fluence. To increase the photon absorption in mid-infrared wavelength, a plasmonic metasurface can be used to increase the light-matter interaction. Furthermore, ultrafast

carrier dynamics of graphene is preserved while in contact with a plasmonic metasurface, as seen experimentally in [38,69].

In this chapter, it is demonstrated that a graphene plasmonic hybrid metasurface can work as a saturable absorber in mid-IR wavelength. The hybrid metasurface design consists of a plasmonic metasurface of pi-shaped coupled nano-antenna structure and monolayer graphene on top of it. The design of the is extensively covered in reports[22,38]. The fabrication of the device is discussed in Chapter 4.

5.2 Device Design

To demonstrate the saturable absorption in MIR, we use the same device design presented in chapter 4 and in[22,37,38]. A summary of the device design and the FTIR characterization is presented in Figure 45. The ultrafast carrier dynamics of the device is similar to the discussion presented in chapter 2, section 2.3.1, only here we are using a MIR laser. However, the carrier dynamics can be different in different wavelength ranges, especially when it's close to the Dirac point. Using MIR laser and changing the incident power of the laser, we can achieve reflection modulation, and at higher power, we expect to see the saturation of the reflection. Unfortunately, we do not have access to an ultrafast MIR laser. So, we have done some measurements that can confirm the saturable absorption in MIR wavelengths, but we do not know the relaxation time of the device.

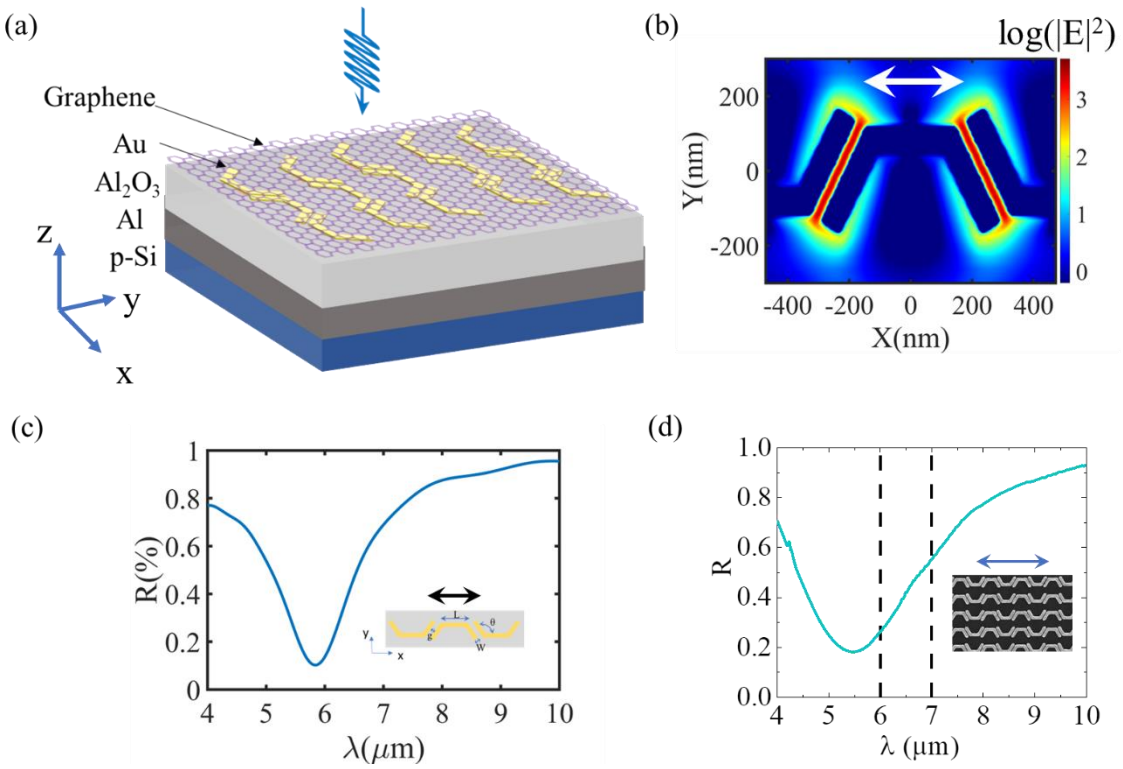


Figure 45: (a) Hybrid Metasurface Working in MIR. (b) near Field Enhancement of the Device at the Resonance (6μm). (c) Simulated Reflection of the Device. (d) FTIR Measurement of the Reflection Spectra of the Device.

5.3 Mid-IR Saturable Absorption Measurement

One of the first measurements we did was in CW mode of MIR laser at 6.31μm. The measurement setup is illustrated in Figure 46(a). Here, we use the power meter to measure the reflected laser power from the sample and from the substrate. By dividing these reflected powers, we get the reflection. At different incident MIR laser power, we get different reflections, as presented in Figure 46(b). The reflection changes from around 2.5% to 7.5%, with a reflection modulation of ~5%.

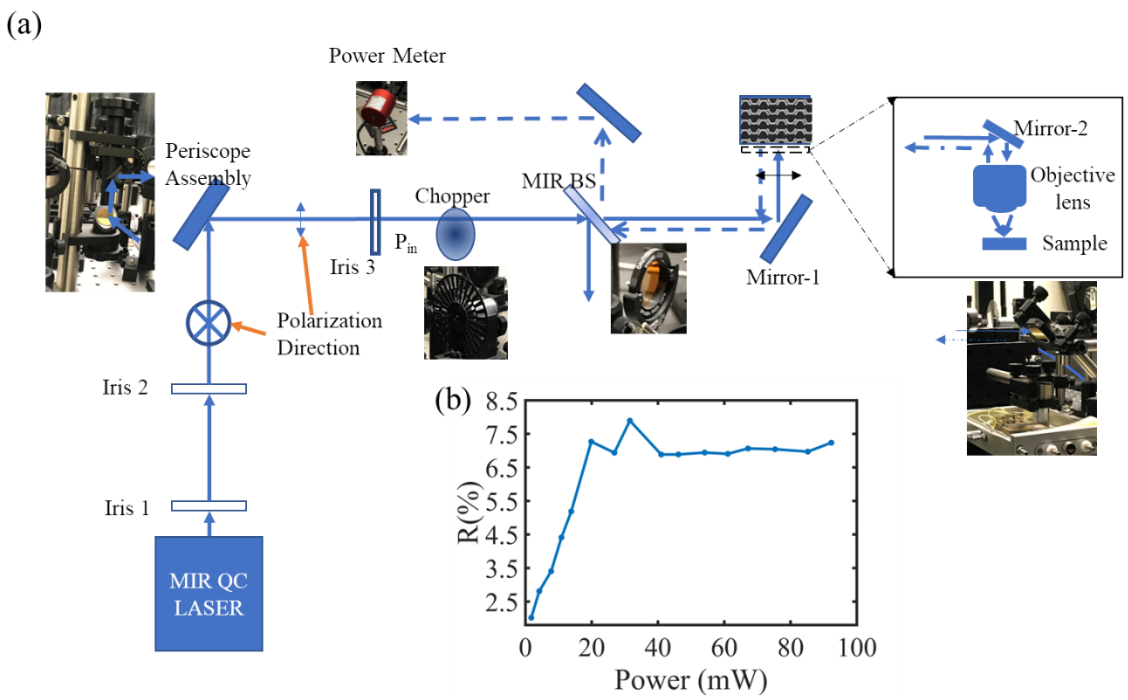


Figure 46: (a) Saturable Absorber Measurement Setup with Power Meter. (b) Saturable Absorption Measurement Result

We see that the reflection obtained from the power meter measurement is not matching with the reflection obtained from the FTIR characterization. A possible reason is the loss of MIR laser power in the optical path, most probably due to the tilting of the objective lens presented in Figure 46(a). Also, the low power measurements are not very accurate as they involve reflected power in the μW range, which is very close to the room's noise level (thermal background).

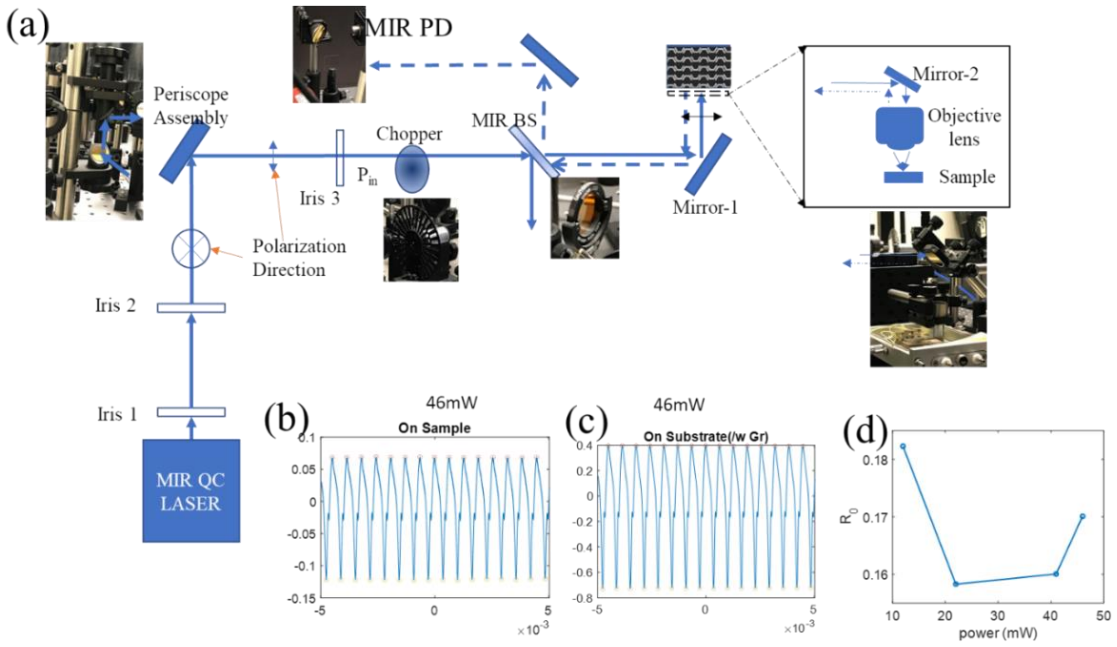


Figure 47: (a) Saturable Absorption Measurement Setup with Photodetector (b) Reflection of Chopped Mir Laser Light Reflection from the Sample (c) Chopped Mir Laser Light Reflection from the Substrate (d) Reflection Change with Changing Incident Power on the Sample

To remove the effect of thermal background, we can use the optical chopper, chop the MIR laser beam at 1kHz and detect the modulated MIR laser pulse in the MIR photodetector. In Figure 47(a) we show the measurement setup with the photodetector and chopper. We measured the chopped MIR laser reflected from the sample (Figure 47(b)) and the substrate (Figure 47(c)) at different incident MIR power. The result from the measurement is resented in Figure 47(d). We can see that R0 is very large at small MIR power, then it decreases with MIR lase power and then it increases again. This result is not conclusive, and we need to do measurement with finer incident power steps and at higher incident

powers (which we don’t have access to due to the low power of our MIR laser and environmental absorption along the optical path)

5.4 Summary

In summary, we observe some saturation behavior of the hybrid metasurface in MIR wavelength, but we need to do more measurements with high power ultrafast laser in MIR to extract the saturation fluence and the relaxation time of our device. We plan to go to CINT to do these measurements as they have some high-power ultrafast MIR lasers available in their facility.

CHAPTER 6

GEOLOCATION DETERMINATION USING UNDERWATER POLARIZATION MAPPING

Underwater navigation is essential for various underwater vehicles such as submarines and Unmanned Underwater Vehicles (UUV). However, there is no reliable method for underwater navigation yet because current technology such as Global Positioning System (GPS) or Wi-Fi does not work underwater. Light polarization mapping under the water can be a viable solution to this problem as the map of the polarization state of light, coming from different directions, is dependent on the specific coordinate system connecting the position of the observer and the position of the source, the sun. So, if one can derive the sun's position from polarization mapping, it is possible to find its location on earth. In this part of my dissertation, numerical simulations and algorithms are presented to find underwater light polarization mapping and fit the measured underwater polarization mapping data. With the methods developed, for clear water in a swimming pool, it is possible to achieve a sun position error of 0.35˚ azimuth and 0.03˚ zenith angle, and the corresponding location prediction error is ~23Km. For turbid lake water, a location determination error of ~100Km is achieved. A numerical model to describe light polarization in turbid water utilizing Mie scattering from nanoparticles is also developed in this part.

6.1 Introduction

Polarization of light, if manipulated properly, can be used as a tool that allows us access to different applications such as polarimetric imaging[72], data communication[73,74], optical image processing[75], quantum computing[76], and our topic of interest 'navigation'[77]. We live under an abundance of light that comes from the sun. Sunlight is unpolarized outside our atmosphere. Once it enters our atmosphere, it gets scattered by different scatters and becomes polarized[78]. And the polarization of this skylight introduced by scattering inside the atmosphere has a unique way of showing us the way. This unique application is to navigate using the polarization of skylight[77]. Navigation with the help of skylight polarization has been used from ancient times by Vikings to find their way in their vast sea explorations[79–81]; insects use it to find their way back to their home[82,83] and so on.

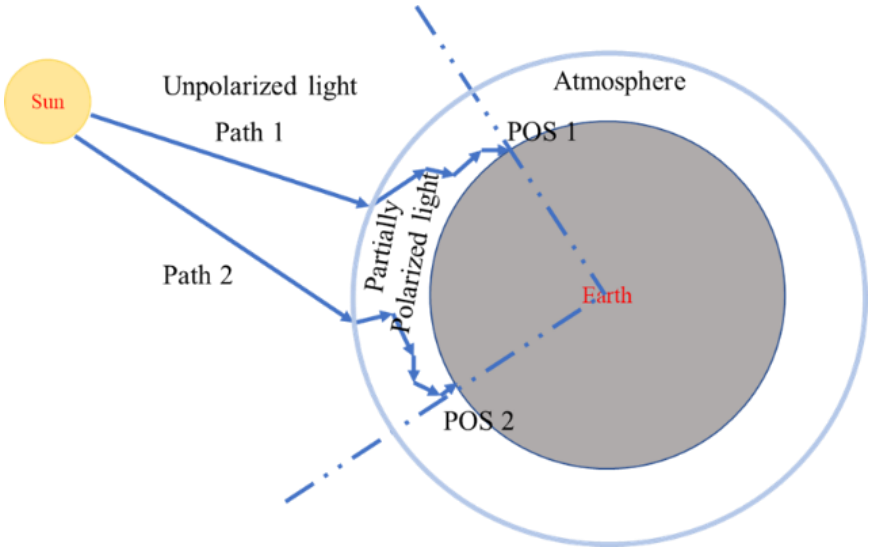


Figure 48: Different Paths of Sun Light Beams and Different Positions on Earth

As the light enters the earth's atmosphere, it goes through multiple scattering events and eventually reaches the earth's surface. In Figure 48, the idea is represented in a very simple

illustration. The two light paths are going through different scattering events; thus, the ultimate polarization state of the two light beams reaching the earth's surface is not the same. Similarly, there are countless photons going through countless scattering events; ultimately, they all have some uniqueness compared to others. That means, if we do a mapping of the polarization at one point on earth, it should be a unique representation of the polarization states of the skylight at that position on earth. In other words, at each position on earth, we should have unique polarization mapping depending on the sun's position. This is the main idea of our work which utilizes the skylight polarization mapping to determine the position on earth.

Inside the atmosphere, there are a lot of particles of different types[84], which introduces different polarization components to the scattered light. There are particles that are small compared to the wavelength of light, and they are known as Rayleigh scatter, which introduces Rayleigh scattering[85]; particles that are larger than the wavelength of light and introduce Mie scattering[86], also there are non-spherical cloud particles, aerosol particles, and all of them contribute to the polarization state of the scattered light. From the literature, we know that clear skylight is mostly linearly polarized, and the circular polarized and unpolarized light is very small compared to the linear polarized part[87,88]. Also, we know that the particles smaller than the wavelength of light or Rayleigh scatters introduce most of the linear polarization that we see in skylight[85]. So, Rayleigh scattering of light is responsible for skylight polarization formation.

Now, to see the polarization distribution from underwater, we need to consider different contributions from different sources. Sunlight and skylight both go inside water through

refraction. Due to different scatterers in the water, sunlight and skylight both can get scattered and change the polarization state of the incoming light. As water has refractive index (n_{water}=1.333) larger than air (n_{air}=1), overall probability of mie scattering increases. Also, there are many more scatterers suspended in water than in air. So, scattering inside water is mainly due to mie scattering.

Consequently, polarization navigation in marine environments is much more challenging than skylight polarization navigation[77,78,132–134] in a terrestrial environment. As a result, the progress of polarization-based navigation in marine environments has fallen behind skylight navigation in terrestrial environments and other marine navigation techniques. In one of the recent demonstrations[135], using a manual rotation camera mounted on a tripod in water depth within 2 to 20m, sun azimuth and elevation angle determination with RMS error of 8.57º and 5.82º respectively and global position error of 817Km (when the sun elevation was more than 40º) is reported. Further processing of the data with the kNN regression model reduced the error to 6.02º, 2.92º, and 422Km, respectively. When sun elevation was below 40º, they obtained an RMS error of 5.46º, 6.23º, and 1970Km, respectively, for sun azimuth, elevation, and global positioning (with kNN regression). Continuous measurement showed their capability to distinguish between two consecutive global positions is around 61Km on average (or 6m error for every 1Km travel). They used a theoretical model considering sunlight refraction and single Rayleigh scattering inside the water to determine the sun and global positions from the measurement. In another more recent demonstration[136], continuous measurements were done inside a pool (depth 1.5m, instrument height unknown) using an underwater fisheye lens camera system. RMS error of the solar zenith and azimuth angle tracking was respectively 0.3º and 1.3º, obtained using

a theoretical model considering single-Rayleigh scattering of sunlight in air and refraction of the skylight into water. However, the performance of state-of-art polarization-based navigation systems[135,136], esp. position accuracy and operation speed, is insufficient for practical applications.

This part of my dissertation focuses on developing the numerical models and perform field tests and data analysis to find the best suitable model and methods for underwater geolocation determination. My lab mates developed the polarization mapping setup here in my lab.

6.2 Problem Statement/ Conceptual Understanding

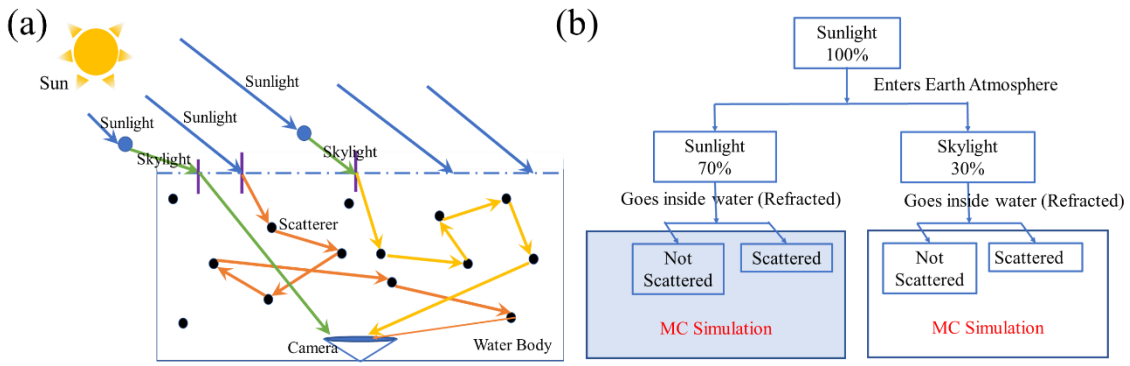


Figure 49: (a) Different Contributions in Underwater Polarization from Different Sources (b) Flow Chart View of the Contributions. Considering a Sunny Day Sunlight and Skylight Ratio is 70% and 30%.

Figure 49 shows the contributing sources for the underwater polarization distribution. The new simulation model tries to capture and describe the events that contribute to the change of the polarization state of light along its path, illustrated in Figure 49(a). Sunlight enters our atmosphere, and some part of that light becomes partially polarized skylight after getting scattered by small scatters (Rayleigh scatters) in the atmosphere[88,134,133,77,137,138].

We call it skylight for simplicity, and the percentage of sunlight that becomes skylight depends on different factors such as weather conditions, presence of aerosol in the air, pollution, etc. When incident upon a water surface, Sunlight and Skylight both refract into the water, and their polarization states change due to refraction from air to water. Depending on the turbidity and depth inside the water, Sunlight and Skylight can get scattered many times and change their polarization states with each scattering event.

In this chapter, we try to decouple the problem into different parts. First, we try to find the skylight polarization pattern and geolocation based on the Rayleigh scattering theory. Second, we try to figure out the appropriate model to describe the underwater polarization pattern obtained from waters from different sources. Third, we present our model to describe the polarization pattern in waters with high levels of turbidity utilizing mie scattering of sunlight and skylight and keeping track of all the possible scattering events using the Monte Carlo Algorithm.

6.3 Skylight Polarization: Single Rayleigh Scattering

The polarization state of light can be fully described by the Stokes parameters[139–142],

$$\vec{S} = \begin{pmatrix} I \\ Q \\ U \\ V \end{pmatrix} \tag{26}$$

Where I is the total intensity of light, Q is the polarization component of light in parallel or perpendicular to the reference plane, U is the polarization component of light at +45° or -45°, and V represents the circular polarization component of light.

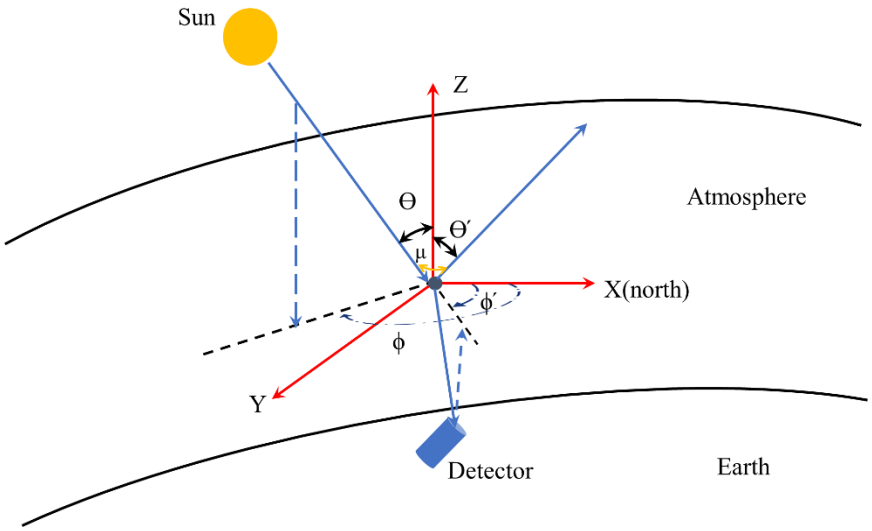


Figure 50: Sun Light Scattering and the Detector System. θ' Is Detector Zenith Angle, φ' Is Detector Azimuth Angle and θ Is Sun Zenith Angle, φ Is Sun Azimuth Angle and θ Is the Scattering Angle.

The total intensity of incoming unpolarized light,

$$I^2 = Q^2 + U^2 + V^2 \tag{27}$$

If the light is partially polarized, then

$$I^2 > Q^2 + U^2 + V^2 \tag{28}$$

Partially polarized light contains different polarization components such as linear, circular etc. The degree of polarization of light can be calculated as

$$DoP = \frac{\sqrt{Q^2 + U^2 + V^2}}{I} \tag{29}$$

Degree of linear polarization of light,

$$DoLP = \frac{\sqrt{Q^2 + U^2}}{I} \tag{30}$$

Degree of circular polarization,

$$DoCP = \frac{V}{I} \tag{31}$$

Angle of polarization,

$$AoP = \frac{1}{2} \arctan \frac{U}{V} \tag{32}$$

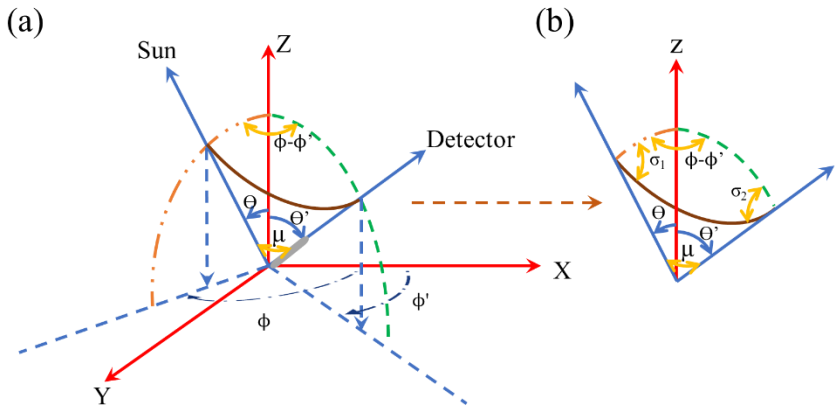


Figure 51: Coordinate System of the Incoming and the Scattered Light. Here, We Only Consider the Scattered Light That Reaches the Detector.

Figure 50 illustrates how particles scatter the sunlight and finally reach the detector on the ground. If light gets scattered by a particle smaller than the wavelength of the light, then the scattering is classified as Rayleigh scattering. When the light gets scattered, the stokes parameter of the scattered light can be calculated as,

$$\vec{S}' = F_r(\vartheta) \,.\, \vec{S} \tag{33}$$

$$F_r(\vartheta) = \left(\frac{2\pi r}{\lambda}\right)^6 \left|\frac{n^2-1}{n^2+2}\right|^2 \begin{bmatrix} \frac{1}{2}(1+cos^2\vartheta) & -\frac{1}{2}(1-cos^2\vartheta) & 0 & 0 \\ -\frac{1}{2}(1-cos^2\vartheta) & \frac{1}{2}(1+cos^2\vartheta) & 0 & 0 \\ 0 & 0 & cos\,(\vartheta) & 0 \\ 0 & 0 & 0 & cos\,(\vartheta) \end{bmatrix} \quad (34)$$

$F_r(\vartheta)$ is the scattering matrix (similar to Muller matrix of optical components) for the Rayleigh scatter[143]. r is the radius of the scatter, n is the complex refractive index of the scatter, λ is the wavelength of the incident light, ϑ is the scattering angle. Scattering angle ϑ depends on the detector and sun position. Both detector and sun position can be specified by their corresponding zenith and azimuth angles. We can derive the scattering angle from spherical trigonometry from Figure 51(b).

$$cos(\vartheta) = cos(\theta).cos(\theta') + sin(\theta).sin(\theta').cos(\phi - \phi') \quad (35)$$

Here, θ' is the detector zenith angle, φ' is the detector azimuth angle and θ is the sun zenith angle, φ is the sun azimuth angle.

Now, we need to remember that the incoming light and the scattered light direction are different, and the corresponding polarization of light also changes. So, the reference coordinate system describing the polarization before and after the scattering is also different. With each scattering, this coordinate system rotates or changes. To account for this, we need to compensate for the rotation introduced by the scattering (σ_1 and σ_2) with the following rotation matrices[144],

$$R(\sigma_1) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & cos\,(2\sigma_1) & sin(2\sigma_1) & 0 \\ 0 & -sin\,(2\sigma_1) & cos\,(2\sigma_1) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{36}$$

$$R(\sigma_2) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & cos\,(2\sigma_2) & sin(2\sigma_2) & 0 \\ 0 & -sin\,(2\sigma_2) & cos\,(2\sigma_2) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{37}$$

σ_1 is the rotation angle introduced to rotate the coordinate system from the incident plane to the scattering plane. $\sigma_2$is the rotation angle introduced during the rotation of the coordinate system from the scattering plane to the detector plane (Figure 51). σ_1 and σ_2 can be derived using spherical trigonometry from Figure 51(b),

$$cos\ \sigma_1 = \frac{cos\theta' - cos\mu\ cos\theta}{sin\mu\ sin\theta} \tag{38}$$

$$cos\ \sigma_2 = \pi + \frac{cos\theta - cos\mu\ cos\theta'}{sin\mu\ sin\theta'} \tag{39}$$

So, to find the actual stokes parameters of the scattered light we need to multiply the scattering matrix by these rotation matrices in the following manner,

$$\vec{S'} = R(\sigma_2)\ F_r(\vartheta)\ R(\sigma_1)\ \vec{S} \tag{40}$$

After we calculate the scatted light stokes parameters, we have the following equation,

$$\vec{S'} = \begin{pmatrix} I' \\ Q' \\ U' \\ V' \end{pmatrix} = = \begin{bmatrix} \frac{1}{2}(1 + cos^2\,\mu) \\ -\frac{cos(2\sigma_2)\,(1 - cos^2\,\mu)}{2} \\ \frac{sin(2\sigma_2)\,(1 - cos^2\,\mu)}{2} \\ 0 \end{bmatrix} \tag{41}$$

And the normalized scattered stokes parameter is,

$$\overrightarrow{S_n}' = \begin{pmatrix} 1 \\ Q'_n \\ U'_n \\ V'_n \end{pmatrix} = \begin{bmatrix} 1 \\ -\dfrac{cos(2\sigma_2)\,(1-cos^2\mu)}{(1+cos^2\mu)} \\ \dfrac{cos(2\sigma_2)\,(1-cos^2\mu)}{(1+cos^2\mu)} \\ 0 \end{bmatrix} \tag{42}$$

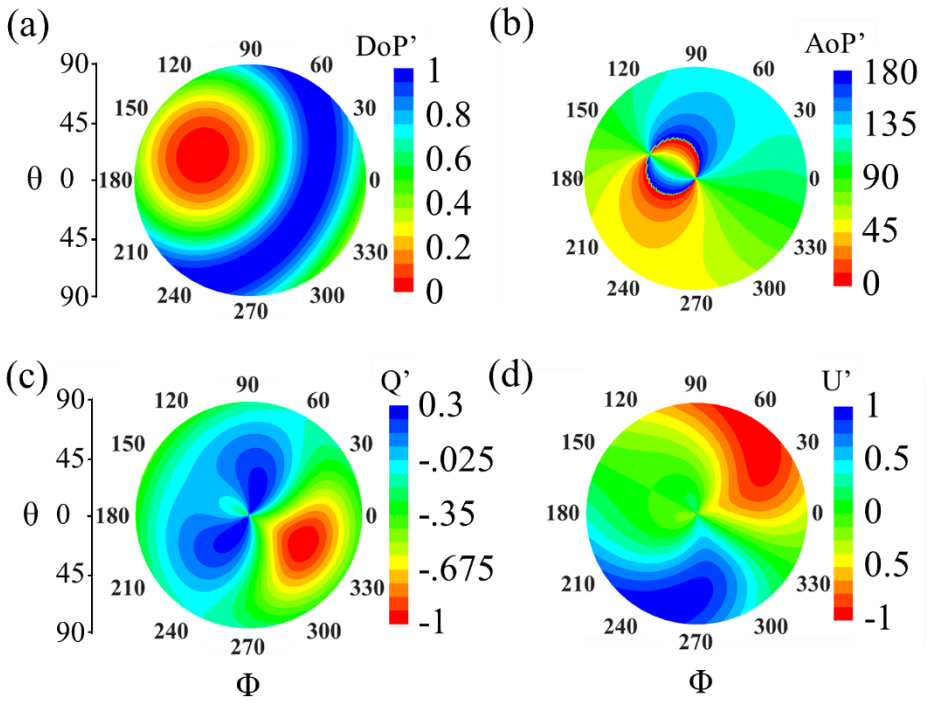


Figure 52: Simulated Mapping of the Skylight Polarization (a) DoLP' (B) AoP' (C) Q'_n (D) U'_n. Simulation Parameters: Latitude 33.4415 and Longitude -111; Date: 10/02/2019; Time: 11am; GMT=-7. Sun Position 41.74° Zenith and 152.25° Azimuth.

From here, we can derive and calculate different polarization parameters for scattered light, such as,

$$DoLP' = \frac{1-cos^2\mu}{1+cos^2\mu} \tag{43}$$

$$Q'_n = -\frac{cos(2\sigma_2)\,(1 - cos^2\,\mu)}{(1 + cos^2\,\mu)} \tag{44}$$

$$U'_n = \frac{sin(2\sigma_2)\,(1 - cos^2\,\mu)}{(1 + cos^2\,\mu)} \tag{45}$$

$$AoP' = \frac{1}{2}tan^{-1}(-tan\,(2\sigma_2)) \tag{46}$$

These parameters are illustrated for certain sun positions in Figure 52.

6.4 Skylight Polarization Measurement

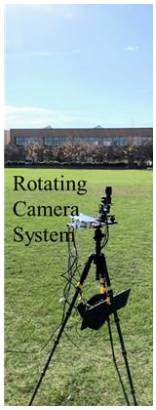


Figure 53: Skylight Polarization Measurement Setup Includes the Polarization Sensitive Camera and Two Robotic Arms to Rotate the Camera in Any Azimuth and Zenith Angle in the Hemisphere. Developed by My Lab Mates: Jing, Jiawei, Jaiu, Neel.

In our lab, we have an automatic measurement setup to take images of the whole sky or hemisphere, as shown in Figure 53. The images are processed to obtain their polarization information and extract the stokes parameters. The images are then stitched together to get the full map of the sky. The camera position can be automatically changed with two motors controlled by a microcontroller.

We have done many skylight polarization measurements with the setup illustrated in Figure 53. Figure 54 and Figure 55 present some of the measured DoLP and AoP, and corresponding simulated DoLP and AoP. The simulation was done using the Rayleigh scattering theory presented in the previous section. The measurement presented here is for a portion of the hemisphere covering different zenith angles on different days.

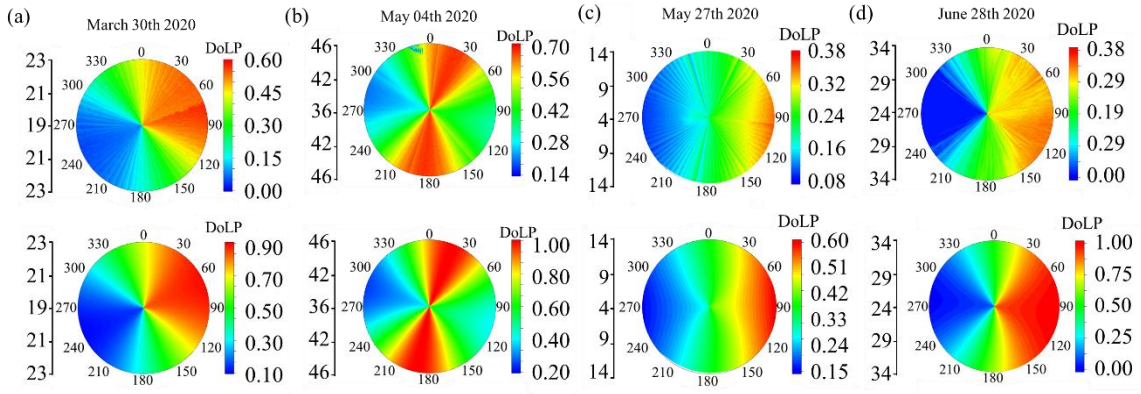


Figure 54: (a-d) First Row: Measured Skylight DoLP and Second Row: Simulated Skylight DoLP Calculated Using the Rayleigh Scattering Theory. The Measurement Presented Here is for a Portion of the Hemisphere Covering Different Zenith Angles on Different Days.

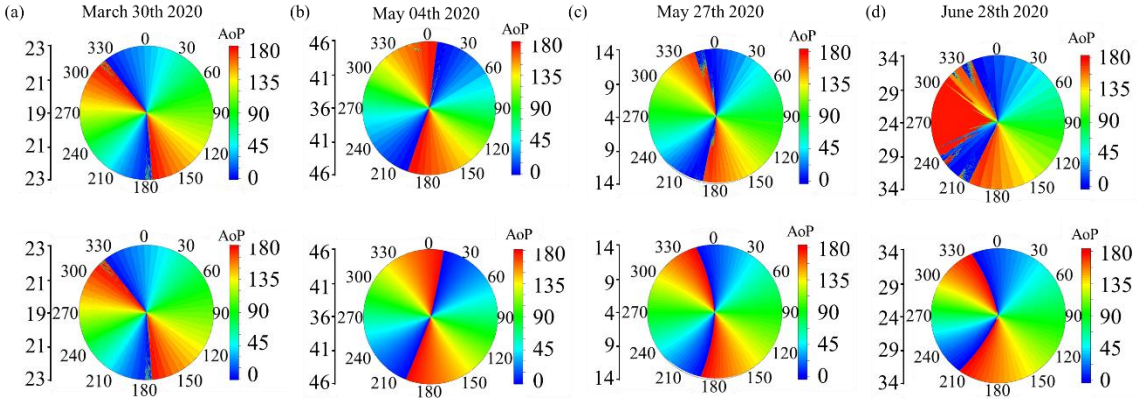


Figure 55: (a-d) First Row: Measured Skylight AoP and Second Row: Simulated Skylight AoP Calculated Using the Rayleigh Scattering Theory. The Measurement Presented Here is for a Portion of the Hemisphere Covering Different Zenith Angles on Different Days.

6.5 Geolocation Determination with Skylight Polarization Mapping

We can find the location of the earth from the skylight polarization measurement. To do this, we need to go through a two-step process. First, we need to extract the sun position from the polarization measurement, and second, we need to find the location of the measurement from the extracted sun position.

6.5.1 Sun Position Determination from Skylight Polarization Mapping

From the measurement, we can have different information at our hand, such as, full stokes parameter mapping of the skylight of that location, different parameters such as DoP, DoLP, and AoP mapping, measurement time and date, time zone of the measurement (GMT), etc. We can take a different route to determine the sun's position from the measurement results.

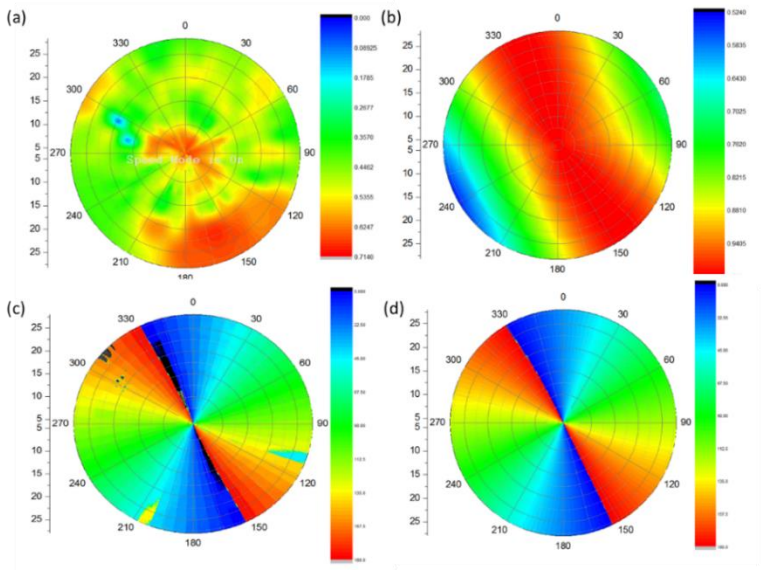


Figure 56: Comparison of DoLP and AoP with Simulation. (a) Measured DoLP (b) Simulated DoLP (c) Measured AoP (d) Simulated AoP. The Sky Was Cloudy at the Time

of Measurement. The Missing Data in (c) Is Due to Bad Frame in the Raw Image. Measurement Date: 02/17/2020; Time: 16:15:00 (Mst); Location: 33.41669(Latitude), -111.932134 (Longitude).

We know that, DoLP is minimum at the sun position. So, we can use DoLP to determine the sun's position. In this case, we need to find the minimum DoLP and find the corresponding azimuth and zenith angle. So, that should be the sun's position. But as illustrated in Figure 56, if the sky is very cloudy, the DoLP becomes very noisy. However, we see that the AoP mapping is very robust even if there is cloud in the sky. So, we look into AoP and analyze the AoP data to find the sun's position.

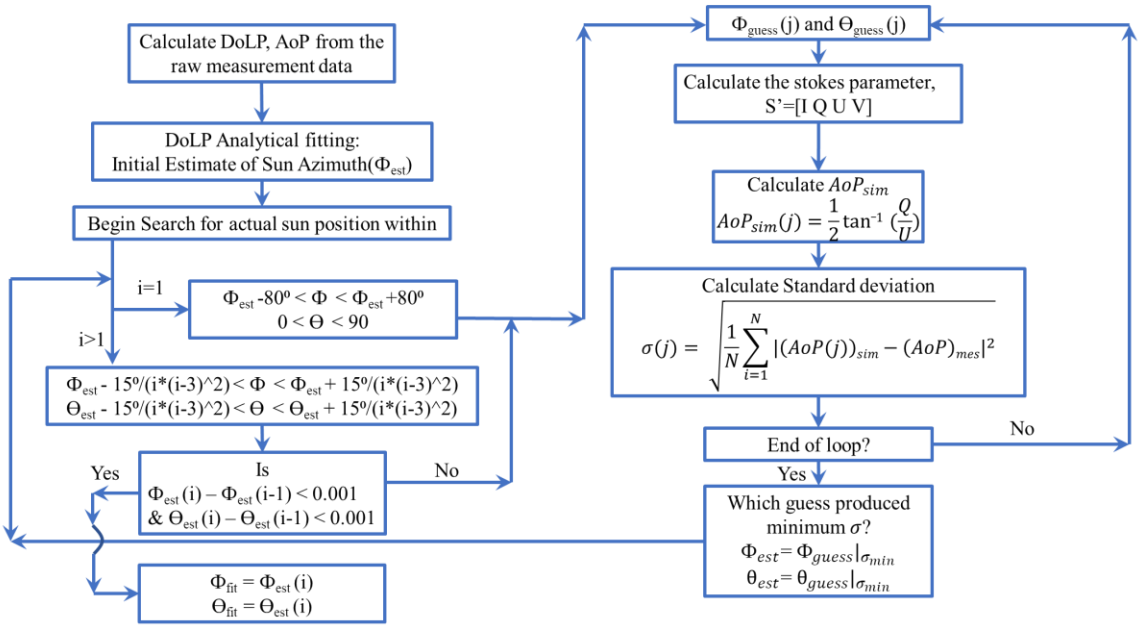


Figure 57: Flow Chart Illustrating the Numerical Fitting Algorithm Used to Find the Sun Position from Measured AoP of Skylight

The sun position can be obtained by comparing the measured AoP with calculated AoP data. First, we can select a certain zenith angle from the measurement data and fit the corresponding AoP data to find a rough estimate, within ±5° accuracy, of the actual sun position. After that, we can fit the whole data set containing all the zenith angle and azimuth angles of the detector. The numerical fitting procedure is illustrated in the flow chart in Figure 57.

In Figure 58, measured AoP data was fitted to find the sun position so that the standard deviation between the calculated AoP at different assumed sun positions and the measured AoP is minimum. As described in the flow chart in Figure 57.

Sun position,

$$(\theta, \varphi) = (\theta_{est}, \varphi_{est}) \; when \; min\left(std\left(AoP_{measurement}(\theta, \varphi), AoP_{Simulation}(\theta_A, \varphi_A)\right)\right) \quad (47)$$

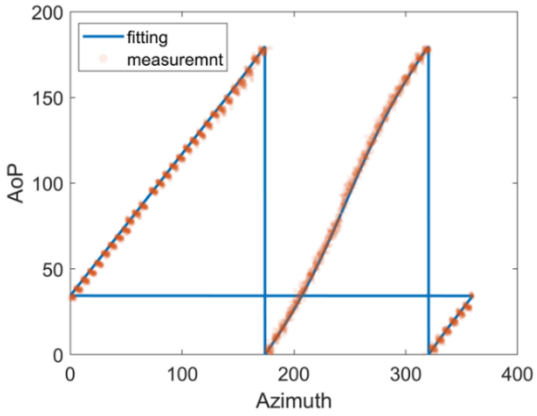


Figure 58: Measured and Fitted AoP at Different Azimuth Angle at a Fixed Zenith Angle of 21.1°. Measurement Date: 03/30/2020; Time: 15:42 (MST); Location: 33.41669(Latitude), -111.932134 (Longitude).

From Figure 58, the Fitted sun position obtained from the numerical fitting procedure is: 247.2755° azimuth and 52.8469° zenith; the Actual sun position: 246.5029° azimuth and 53.2274° zenith. So, the error is around 0.77˚ azimuth and 0.38˚ zenith angle.

6.5.2 Exact Location Determination

After we have determined the sun's position, we can move forward to find the location of the earth that this sun's position corresponds to. To do this, we need to use the measurement date, time, time zone (GMT), and the extracted sun position from the previous section.

The sun's position is related to the location on earth through the following equations (derived from spherical geometry in)[145],

$$cos\ (\theta) = cos(lat)\, cos(\omega)\, cos(\delta) + sin(lat)\, sin\ (\delta) \tag{48}$$

$$cos\ (\phi) = \frac{cos(\delta)\, sin(lat)\, cos(\omega) - sin(\delta)\, cos\ (lat)}{cos\ (90 - \theta)} \tag{49}$$

Here, lat is the latitude and lon is the longitude of the location on earth, ω is the hour angle, δ is the declination angle, Θ is the zenith angle of the sun, and ϕ is the azimuth angle of the sun. The hour angle ω is derived from local solar time or LST, which represents the actual solar time at that position. The equations for ω and LST are provided below[145–147],

$$\omega = 12(LST - 12) \tag{50}$$

$$LST = LT - \frac{1}{15}(LSM\ - Lon) + EOT\ - DLS \tag{51}$$

LST or Local solar time is derived from the local time (LT), the difference between the actual longitude (Lon) and the standard meridian for the local time zone (LSM), Equation of Time (EOT), and correction for daylight saving or DLS.

The equation for LSM is provided below[145–147],

$$LSM = 15 * GMT \tag{52}$$

LSM is obtained from Greenwich Mean Time or GMT. GMT is the difference between your time and the time in the prime meridian. Whether you are east or west of the prime meridian, GMT will be either positive or negative. Equation of time is provided below[145–147],

$$EOT = 0.165\, sin(2B) - 0.126\, cos(B) - 0.025 sin\,(B) \tag{53}$$

$$B = \frac{360}{365}(n - 81) \tag{54}$$

The equation of time depends on the number of day (derived from the date of measurement) of the year or n.

The equation of declination angle is provided below[145–147],

$$\delta = 23.45 si\, n\left(\frac{360}{365}(284 + n)\right) \tag{55}$$

The declination angle, δ, also depends on the number of the day or n and it's not dependent on the observer's position on earth.

Now, if we know the sun's position, date, time, and time zone of the measurement, then we know everything except the location or the latitude and longitude of the earth. So, we can use the extracted sun position and these equations to calculate the latitude and longitude of the measurement location numerically. A brief description of the location determination process is presented in the flow chart in Figure 59.

For the measured AoP presented in Figure 55 and the fitting of AoP presented in Figure *58*: GMT=+7; Date: 03/30/2020; Time: 15:42 pm and Fitted Sun position: 247.2755°

azimuth and 52.8469° zenith (from Figure *58*). Now, using the equations from (48) to (55), we can calculate the sun's position for different assumed latitudes and longitude, as described in Figure 59. And the assumed latitude and longitude that produces the closest sun position compared to the fitted sun position is our location. Utilizing this process, the location of our measurement on 03/30/2020 is

Latitude predict = 32.354

Longitude predict = -111.886

And the actual location is 33.41669 (Latitude), -111.932134 (Longitude).

So, the location prediction error is around 118.241 Km.

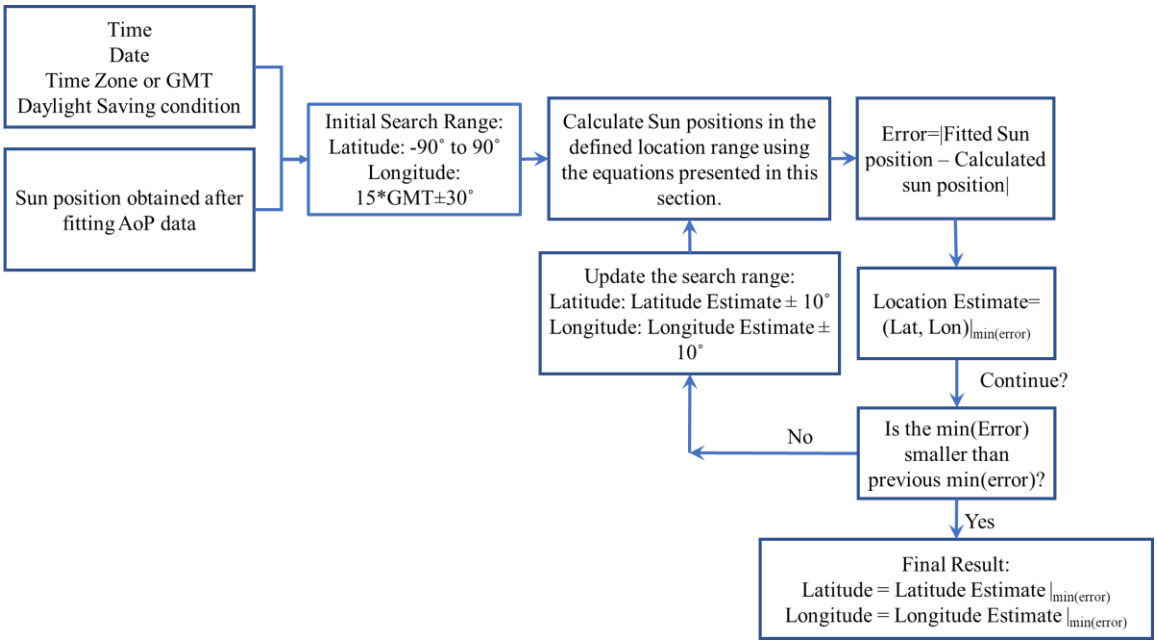


Figure 59: Flow Chart Describing the Process of Location Determination from Fitted Sun Position

6.6 Underwater Light Polarization

On the ground, Using the polarization mapping of the sky, it is possible to find the sun's position (zenith and azimuth angle) using a traditional fitting procedure and comparing it with the Rayleigh scattering theory calculation. After finding the sun's position, we can find the location on earth utilizing the equations that relate the sun's position with the location of the observer on earth.

When we go into the water, the problem becomes intricate. If we look up from inside water, the whole hemisphere shrinks inside the Snell's window (±48.6º from the zenith direction), which covers only 97.2º of the hemisphere inside water. So, here, we need to consider what can contribute to the polarization states of light outside Snell's window. The sunlight and skylight both can transmit into the water, and different types of scattering of these lights contribute to the polarization distribution over the whole hemisphere seen from under the water. So, we need to account for these contributions and track the polarization states of each photon as it travels from outside water towards the detector inside water. Monte Carlo simulation can help us do this very efficiently by tracking each photon and its polarization states.

If we put our detector inside water, it will be able to measure the polarization distribution from under the water. Now, depending on the depth of water and turbidity, we can determine the contributing sources of the underwater polarization distribution. If the detector is in shallow water and the water is very clean, then the skylight getting refracted inside water is not going to get scattered by particles inside water. The polarization state of light will only change due to the refraction of light from air into water. Figure 60 illustrates

this situation. Sunlight gets scattered in the air (forms skylight), and this scattered light gets inside water, and the detector detects the polarization state of the light. A simple demonstration of this process is presented in Figure 60.

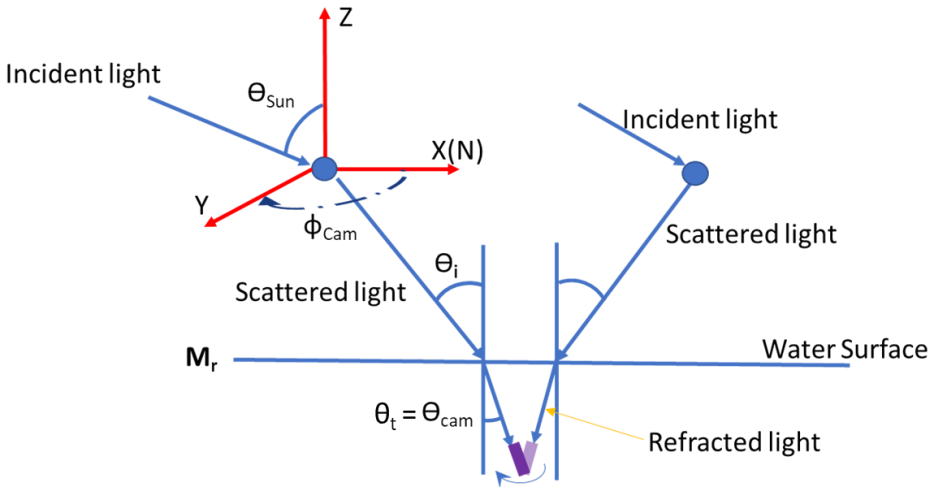


Figure 60: A Representative Sketch of the Skylight Refracting Inside Water.

So, equation (40) is slightly modified to include the polarization state change due to the refraction of the skylight into the water through the air-water interface,

$$\vec{S}' = M_r \, R(\sigma_2) \, F_r(\vartheta) \, R(\sigma_1) \, \vec{S} \tag{56}$$

Here, M_r is the muller matrix (transmission) for the air-water interface[135,136,148–151]

$$M_r = \frac{1}{2}\begin{bmatrix} {t_s}^2 + {t_p}^2 & {t_s}^2 - {t_p}^2 & 0 & 0 \\ {t_s}^2 - {t_p}^2 & {t_s}^2 + {t_p}^2 & 0 & 0 \\ 0 & 0 & 2tstp & 0 \\ 0 & 0 & 0 & 2tstp \end{bmatrix}$$

And the perpendicular (t_s) and parallel (t_p) transmission coefficient of light[135,136,148,149] considering the refractive index of air and water; and the zenith angles of the incident light and transmitted light (θ_i and θ_t).

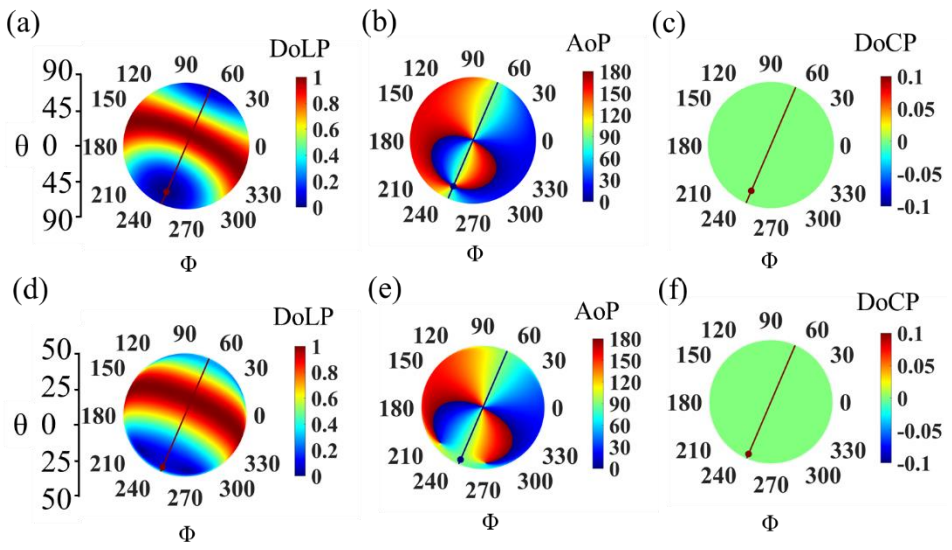


Figure 61: Skylight Polarization Map (a) DoLP (B) Aop and (C) DoCP. Light Polarization after Skylight Goes into Water (D) DoLP (E) AoP and (F) DoCP. These Are Simulated for Sun Position, Azimuth= 246.76º, and Zenith=71.83º. The Black dot indicates Sun Position.

Figure 61 illustrates the simulated skylight polarization pattern and the polarization pattern seen from under the water. Here, we can see the differences in polarization distribution in the skylight and the light refracted inside water.

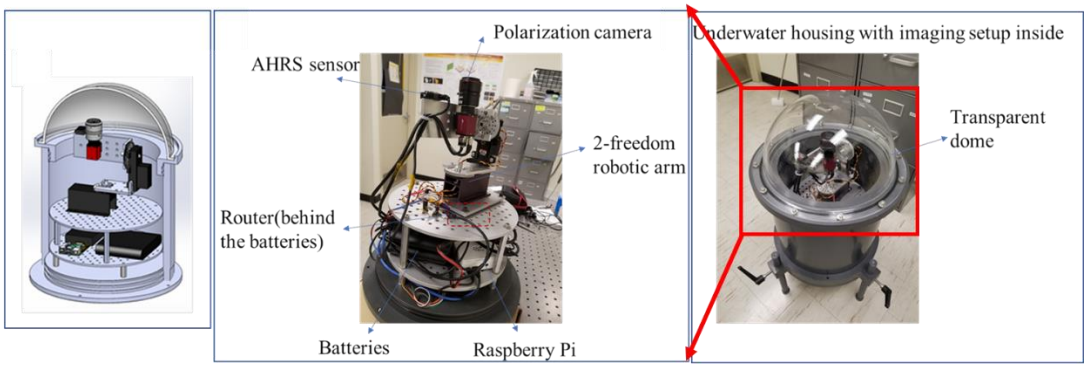


Figure 62: Underwater Polarization Measurement Setup Developed in Our Lab. In the Figures, the Important Components in the Setup Are Indicated. Setup Developed By: Jing, Jaiwei, Jaiu, Neel, Ahmed.

To observe the polarization distribution from under the water, we would not be able to do it with our previous measurement setup. So, we developed a waterproof polarization measurement setup for underwater polarization measurements. The setup is illustrated in Figure 62.

The underwater polarization measurement setup includes two robotic arms (2 degrees of freedom), two polarization sensitive camera[152] the same lens (f =35mm, F/# =2.8, FOV = 14°x 10°) mounted on the robotic arm used to capture the full stokes parameter and obtain DOLP/AOP and DOCP (using a CP filter on one of the cameras), a router to connect to the microprocessor and the arms, AHRS sensor to determine the azimuth angles, Raspberry pie microcontroller, and batteries. The images captured at different directions are later stitched together based on the information captured by the AHRS sensor mounted on the same robotic arm as the camera.

All the components inside the housing are covered with a waterproof transparent dome. For this setup, we needed to make sure that the plastic used to make the transparent dome does not change the polarization state of light. So, we did some skylight polarization measurements with and without the dome, at the same location and date, and the results indicate that the dome does not change the polarization state of light.

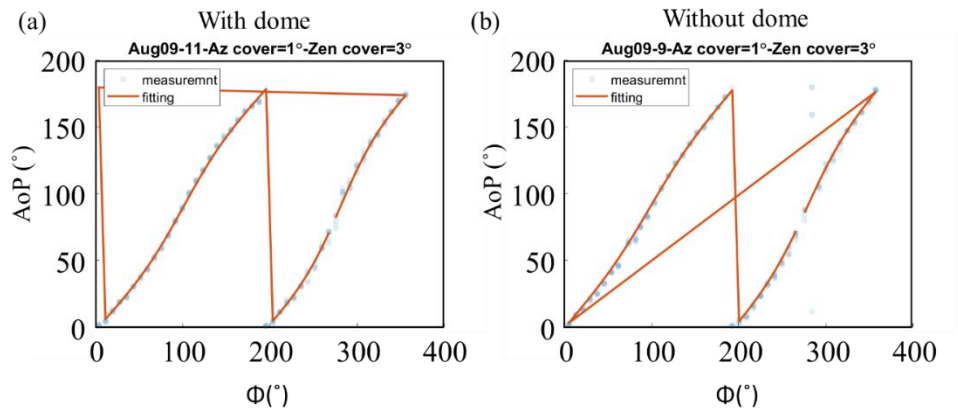


Figure 63: Skylight Polarization Mapping (a) Measured AoP and Fitted AoP with the Dome. (b) Measured AoP and Fitted AoP Without the Dome. Fitting Was Done Using the Method Described in the Flow Chart in Figure 61 Using Rayleigh Scattering Theory to Calculate the Stokes Parameters after Light Scattering.

In Figure 63, we show two different skylight polarization mapping, AoP, one with the dome and one without the dome. The fitted AoP is also shown in the figure. The fitting results:

With Dome:

	AZ	ZEN
Actual Sun position:	283.60˚	82.67˚
Fitted Sun position:	283.98˚	82.22˚
Error:	-0.38˚	-0.5˚

Without Dome:

	AZ	ZEN
Actual Sun position:	282.27˚	80.60˚
Fitted Sun position:	282.78˚	80.80˚
Error:	0.51˚	0.2˚

The skylight polarization mapping and the fitting results suggest that the measurement error is similar with and without the dome. So, the effect of the dome is very minimal or no effect. In Figure 64, the measurement accuracy across multiple measurements is shown, and the results seem to be unaffected by the inclusion of the dome.

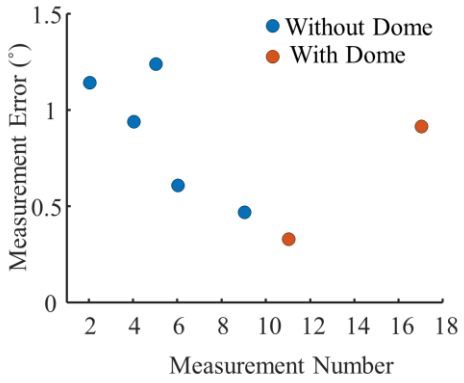


Figure 64: Measurement Accuracy (Sun Azimuth Fitting Error) Across Multiple Measurements to Check the Effect of Adding the Transparent Dome.

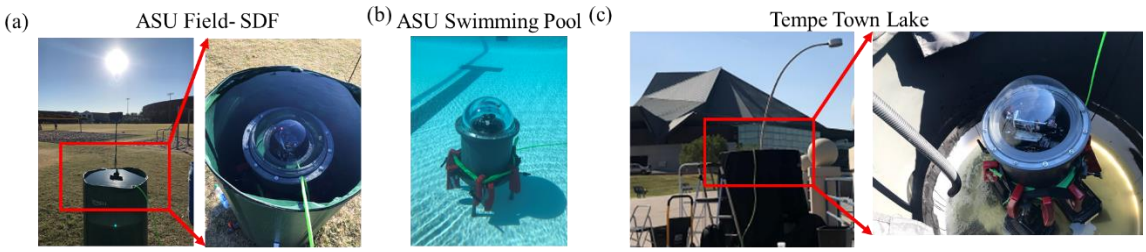


Figure 65: Underwater Polarization Measurements with Our Setup in (a) ASU Field, Sun Devil Field, Inside a Flexible Water Tank (b) ASU Swimming Pool (c) Tempe Town Lake Water Inside a Large Water Tank.

After confirming that the dome does not change the polarization state of light by itself, we moved to do some underwater polarization measurements. We did the underwater polarization measurements in three different settings, as shown in Figure 65. First, we needed to know if our setup would work underwater. So, we utilized a flexible water container large enough to hold more than 120 gallons of water, put our setup inside, poured clean tap water to fill it up, and conducted the measurements. Second, we put our setup inside the ASU swimming pool. Third, we moved to Tempe town lake and did the underwater polarization mapping using a lake water-filled container.

6.6.1 Underwater Polarization Measurement Inside Water Tank in ASU Field (SDF)

We have done multiple measurements with this setup, putting our polarimetric imaging instrument inside a portable water tank, as shown in Figure 65a. Most of the measurement with this flexible water tank was done in ASU Sun Devil field, Lat: 33.417205, Lon: -111.931065. During the measurements, we used a pole outside of the water tank to block the sun to avoid overexposure. Two of these measurements are shown respectively in Figure 66 and Figure 67.

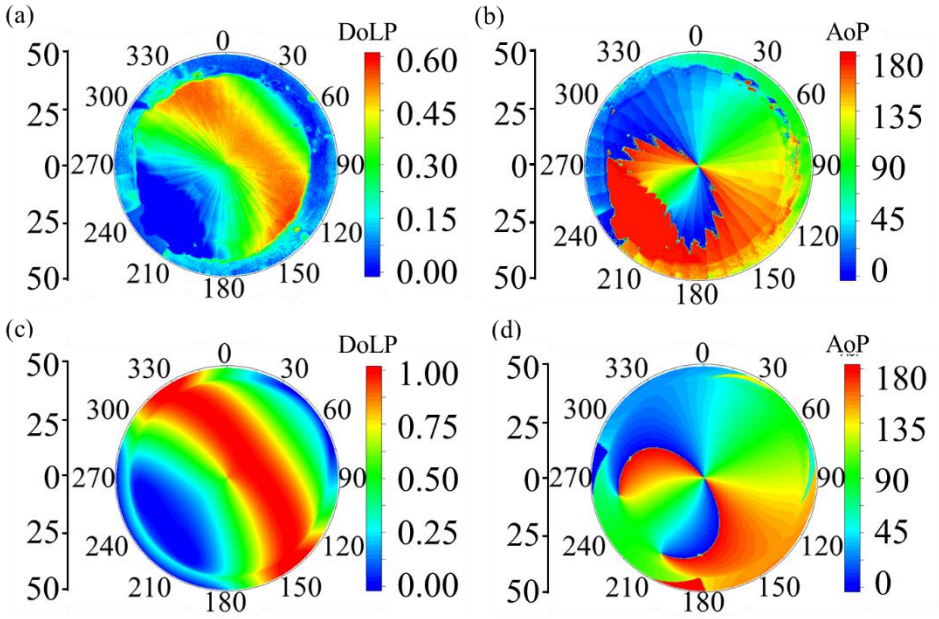


Figure 66: Underwater Polarization Mapping Done on October 26, 2020. Location: ASU Football Field, Lat: 33.417205, Lon: -111.931065. (a) Measured DoLP (b) Measured AoP (c) Simulated DoLP (d) Simulated AoP

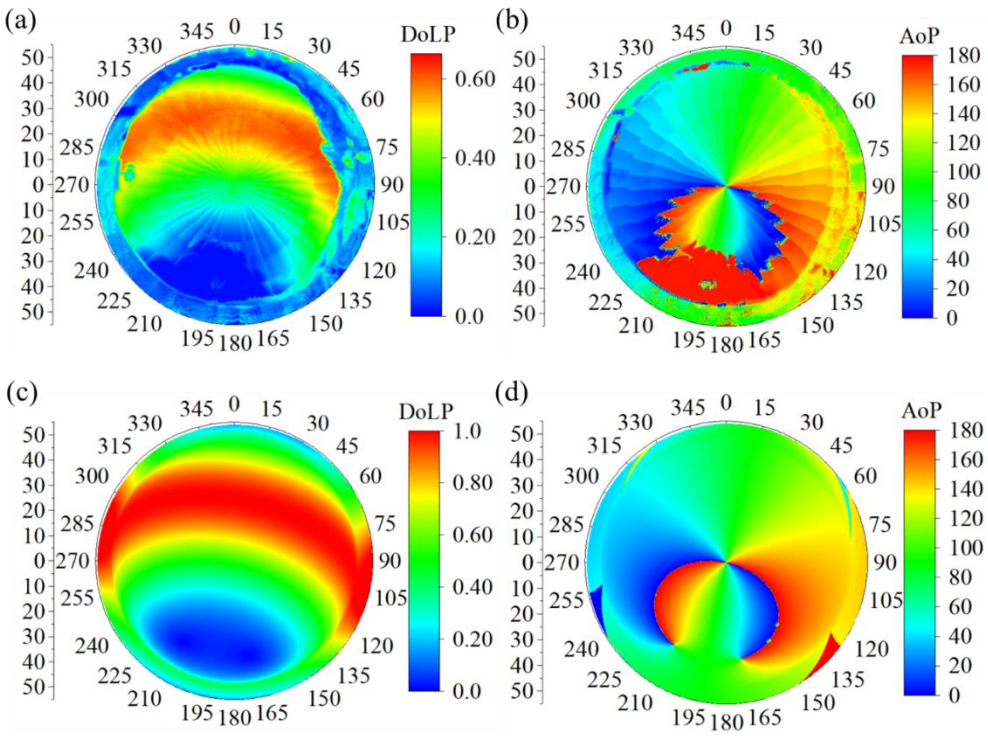


Figure 67: Underwater Polarization Mapping Done on December 1st, 2020. Location: ASU Football Field, Lat: 33.417205, Lon: -111.931065. (A) Measured DoLP (B) Measured AoP (C) Simulated DoLP (D) Simulated AoP

As the measurements were done in very shallow water (~5" depth) and the water was very clean, the simulated DoLP and AoP for both of these measurements match with the measured DoLP and AoP quite well. The simulations are obtained with the simple model described in the previous section, where we consider that the skylight goes into water and changes its polarization only due to the air-water interface, equation (56). No underwater scattering is considered for this simulation.

We have also fitted these measured data to find the sun position from these measured AoP. One of the fitting results for December 1st, 2020, is shown below and in Figure 68.

AZ ZEN

Fitted Sun position: 188.31˚ 56.35˚

Actual Sun position: 187.62˚ 56.16˚

Fitting error: 0.69˚ 0.19˚

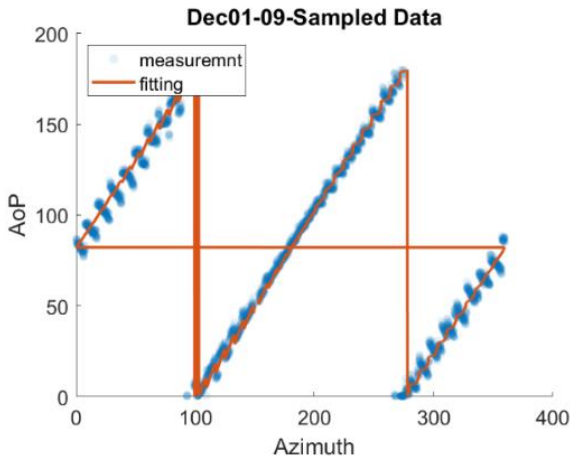


Figure 68: Measured AoP and Fitted AoP. Measurement Was Done in ASU Soccer Field on December 1st, 2020. Fitting Was Done Using the Method Described in the Flow Chart in Figure 61 While Using the Simple Underwater Polarization Model, Change of Skylight Polarization Only Due to Light Transmission Inside Water, Equation (61) to Calculate the Stokes Parameters.

From the fitting, we can say that the model used to describe the measured data is okay for these measurements and the fitting accuracy is also good enough.

6.6.2 Off-center Camera Correction

One thing to mention is that we have also considered a correction for off-center or off-axis camera position errors for these measurements. To understand the correction, we need to look at Figure 69. Notice that the camera is hooked on the robotic arm vertically, but they are not in the center. This does not cause any problem when the underwater housing is in

the air but when it is underwater, due to two interfaces: one air-water interface and again a water-air interface, the light position recorded in the image sensor is a bit different than the actual light direction coming towards the dome.

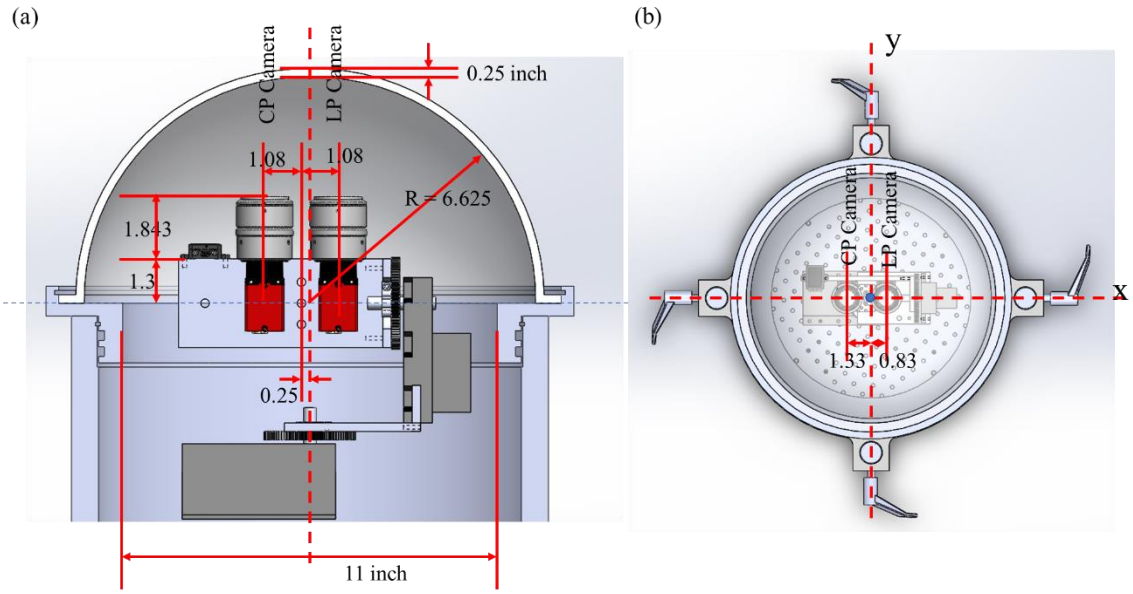


Figure 69: (a) a Cross Section of the Underwater Polarization Mapping System Showing the Side View of Inside. Some of the Dimensions Are Also Indicated. (b) Top View of the Same System.

So, after doing the measurement, we need to make some corrections to the images' recorded azimuth and zenith angles to offset this error. The calculation process for the off-center camera position correction is presented in Appendix B.

Zenith Angle Correction for off-center camera position:

$$\theta^{new} = \frac{a\cos(\cos(\theta))}{\sqrt{((\tan(\theta_t^D - \theta_i^D))^2 + (\sin(\theta))^2 + (\cos(\theta))^2)}}$$

So, the zenith error $\Delta\theta = \theta^{new} - \theta$; Here, θ^{new} is the corrected zenith angle and θ is the recorded zenith angle during measurement.

Azimuth Angle Correction for off-center camera position:

$$\Delta\varphi = \frac{a\,sind((tan(\theta_t^D - \theta_i^D))}{\sqrt{((tan(\theta_t^D - \theta_i^D))^2 + (sin(\theta))^2)}}$$

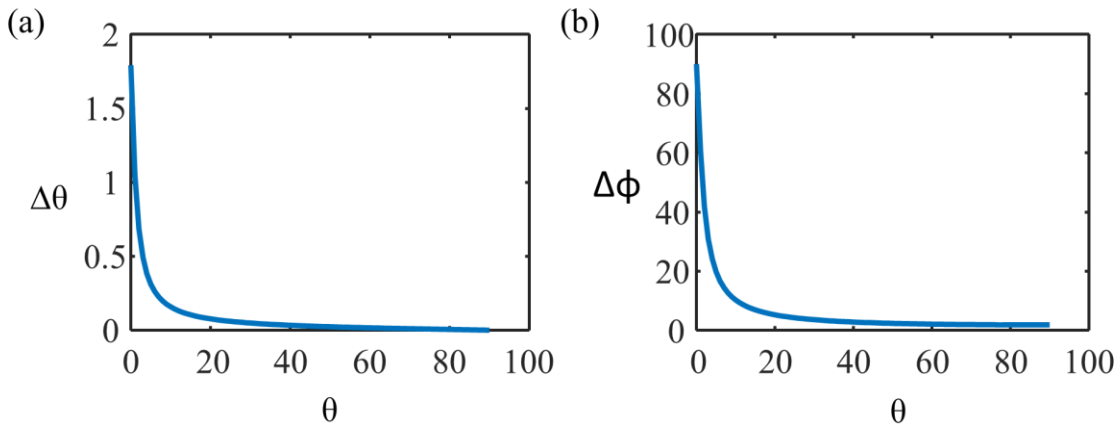


Figure 70: Off-center Camera Correction (LP Camera, .83 Inches Away from Center) for (a) Zenith Angle Error (b) Azimuth Angle Error. The Errors can be Different for Different Distances from the Center.

Obtained zenith and azimuth angle errors for different recorded zenith angles are presented in Figure 70. For the LP camera, We have to add these errors (Δφ and Δθ) with the recorded azimuth and zenith angles to obtain the correct azimuth and zenith angles. For the CP camera, subtract the azimuth angle error and add the zenith angle error.

6.6.3 Underwater Polarization Measurement in ASU Swimming Pool

We placed the 29-inch-high underwater housing (120lb weight) in our 48-inch-depth swimming pool in ASU (33.415196, -111.931602), as shown in Figure 71a. Then, 80lbs weights were mounted on the housing to completely sink it on the floor. Next, a remote-control Raspberry Pi is used to control the automatic polarization imaging system inside the housing from the poolside through an ethernet cable, as shown in Figure 71b.

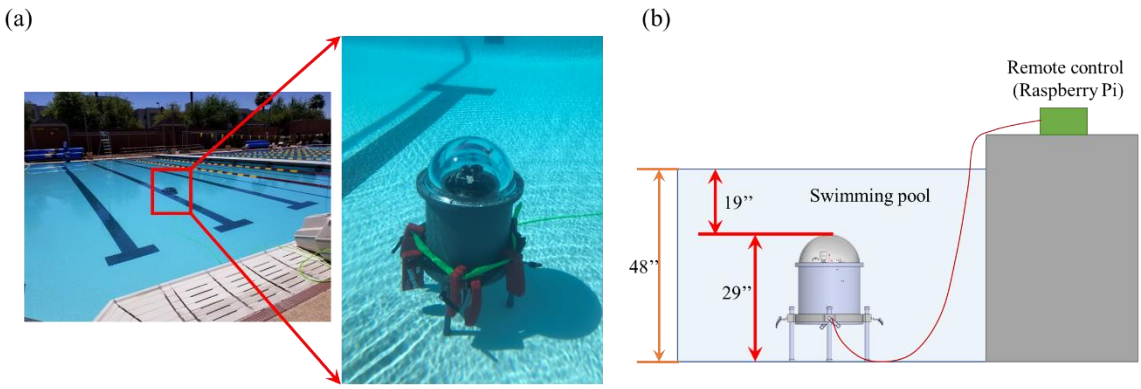


Figure 71: (a) Our Underwater Polarization Measurement Setup Inside the Swimming Pool (b) Depth of Measurement Was about 19". Raspberry Pi is Used to Control the Imaging Setup From the Side of the Swimming Pool Connected with an Ethernet Cable.

We first leveled the underwater housing by manually adjusting the length of the legs of the housing and checking a bubble leveler mounted on the housing. Then we performed automatic polarization image mapping to cover 0˚-360˚ azimuth angle and 0˚-50˚ zenith angle (Snell’s window) range above the housing.

The swimming pool had very clear water, and we could see the housing very clearly from above the water. The measurement depth was 19 inches. We did a couple of measurements in 2021. Some of the measured AoP and simulated AoP are presented in Figure 72. All the measured AoP has a certain portion of the mapping overexposed which is due to direct sunlight.

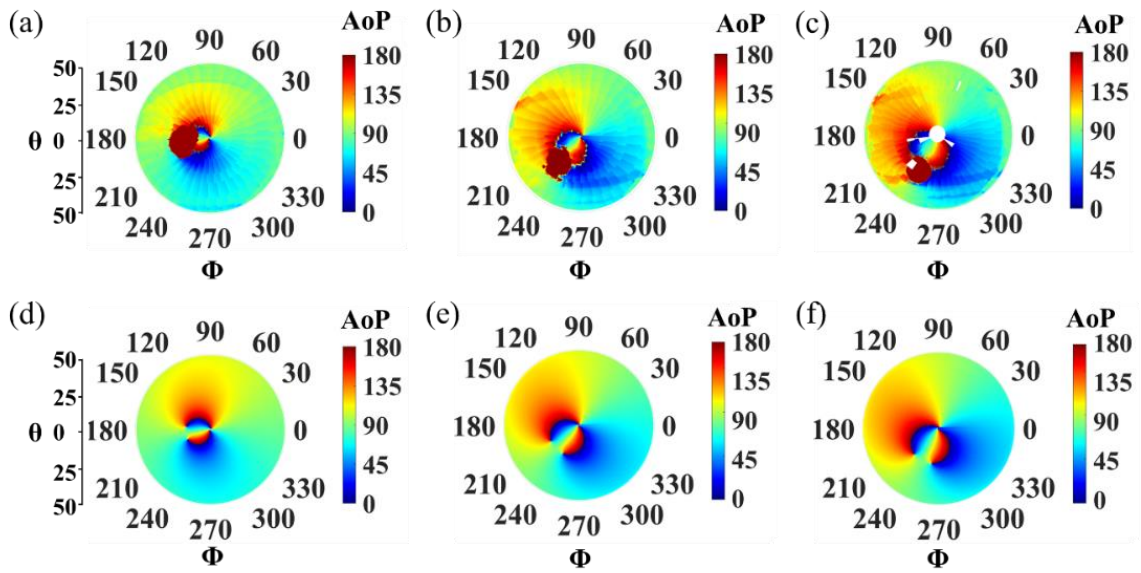


Figure 72: Underwater Polarization Measurements in Asu Swimming Pool on 04/21/2021 (a) Measured AoP at 12.50pm (b) Measured AoP at 1.54pm (c) Measured AoP at 2.28pm (d-f) Corresponding Simulated AoP Obtained Considering the Same Location, Date and Time of Measurements.

If we compare the measured and simulated AoP presented in Figure 72, we can see that they are quite similar. So, even at this depth (~19”), for clear water, we can reproduce the measured AoP with the simple skylight refraction model presented in the previous section.

6.6.5 Magnetometer Offset and DoCP Calculation from Underwater Measurement in Swimming Pool

Magnetometer Offset:

The magnetometer attached to the robotic arm of the polarization-sensitive camera deviates over time and introduces errors in the recorded azimuth angles. To find the magnetometer offset error, we take the photo of the sun, keeping it in the middle of the image, in between measurements as a reference to look back (we also record the time to calculate the actual sun position). We can also find the magnetometer offset by comparing the sun position

recorded in the intensity profile of both the LP camera and the CP camera. In this case, we also need to remove the off-center camera error. As the two errors, magnetometer offset and off-center camera error, are coupled together, if we remove the off-center camera error from the recorded sun position, only the magnetometer offset is left. This way, we can calculate the magnetometer offset from the actual sun position calculated from the recorded time. The process of calculating the magnetometer offset is shown in Appendix B.

DoCP Calculation:

We are using two cameras to measure light's LP and CP polarization information. We call these two cameras LP and CP camera. To calculate the Degree of Circular Polarization or DoCP, we use the LP intensity and CP intensity measured by these two cameras. The critical point to notice is that the distance of these two cameras from the center is also not the same. LP camera is 0.83", and CP camera is 1.33'' from the center. So, the off-center error introduced for these two cameras is also different. To compensate for this, we follow the steps detailed in Appendix B to calculate the DoCP

We think that the source of the CP (circularly polarized) light in low turbidity swimming pool measurement is because of the surrounding environment, such as the buildings, chairs, tables on the side of the pool etc. So, this DoCP can help us remove some of the measured AoP data and help us increase the accuracy of finding the sun position from fitting.

6.6.6 Fitting Underwater Polarization Measured Data: Swimming Pool

To do the fitting, we need to do some post-processing of the measured AoP. First, we need to remove the over-exposed data from consideration. To do this, we need to consider the intensities recorded in each of the polarization gratings (0˚, 90˚, 45˚, and 135˚) of the image

sensor and remove any intensity if it is larger than 0.99. Then we use the rest of the intensities from different gratings to calculate the full stokes parameters and finally calculate AoP from stokes parameters Q and U. Second, as we do not expect to see much DoCP in clear water with very low turbidity, we can use the high DoCP to remove some AoP data. This can help us achieve higher accuracy. The modified numerical fitting procedure for this data is presented in Figure 73. Third, we do the fitting to find the sun position, and then we need to apply the magnetometer offset and off-center camera correction on the obtained fitted sun position to get the final fitted sun position.

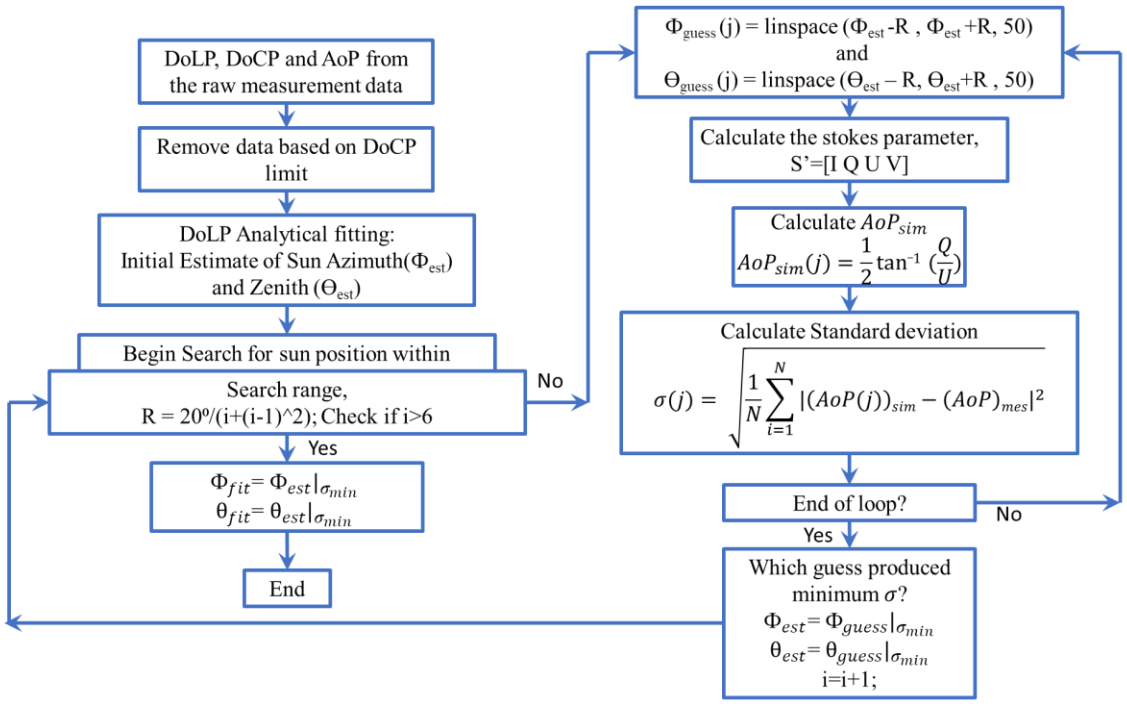


Figure 73: Flow Chart Describing the Numerical Fitting Procedure of Underwater Polarization Measurement Data from Swimming Pool.

The water in the swimming pool was clean, and the measurement was done in a clear sky condition. Figure 74a shows the underwater polarization imaging system inside swimming

pool water. Figure 74b shows the obtained AoP mapping of the partially polarized light inside water. We can see that some of the data is overexposed because the camera looks directly at the sun (red circular portion) for those areas. This can be removed by considering the overexpose intensity of each polarization grating on the camera image sensor.

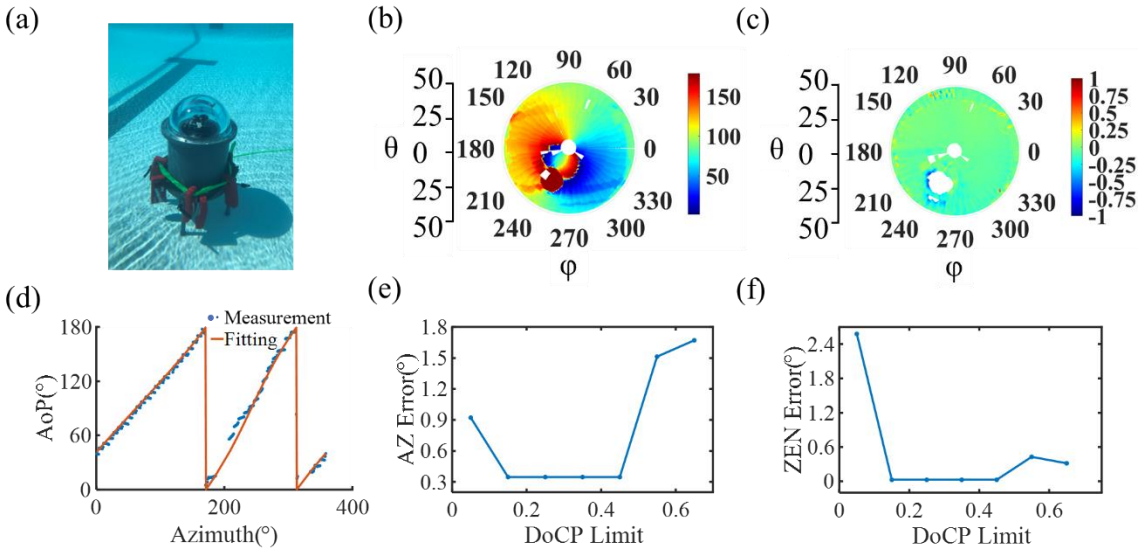


Figure 74: Swimming Pool Measurement on 04/19/2021 at 2.28pm. (a) Measurement Setup Inside Swimming Pool (b) AoP Mapping (c) DoCP Mapping of Partially Polarized Light Inside Swimming Pool Water (d) Fitting AoP Mapping Result with Rayleigh Scattering Model. Inset: AoP Mapping after Removing Data Corresponding to Overexposer and High DoCP. (e) Azimuth Angle (f) Zenith Angle Error for Different Docp Limits Used to Remove Data from AoP Mapping.

As the sky was clear and the water was clean, we did not expect to see much DoCP inside the swimming pool water. However, Figure 74c, the measured DoCP inside swimming pool water, shows that there is a significant amount of DoCP detected. This DoCP could arise from light scattering from the surrounding objects near the swimming pool, the edge of the swimming pool, and buildings near the swimming pool. So, the data corresponding

to large DoCP can be removed to remove some of the noise we see in AoP mapping in Figure 74b. A representative fitting result is shown in Figure 74d, after removing the overexposed data and some of the data corresponding to high DoCP. The inset of Figure 74d shows the corresponding AoP mapping after the data removal. AoP mapping after removing data with different DoCP limits is shown in Figure 75. A summary of the fitting errors obtained with this method is presented in Figure 74e and f, corresponding to the Azimuth angle fitting error and Zenith angle fitting error, respectively. We can see that when very little data is removed, corresponding to DoCP larger than 0.45; the error is high for both azimuth and zenith angles. After we remove some data corresponding to DoCP larger than 0.4 to 0.15, the error becomes very small for the azimuth angle, and the zenith angle error is zero here. Then if we remove too much data, the fitting error becomes high for both azimuth and zenith angle. So, we need to balance the number of data we can remove and the expected fitting error.

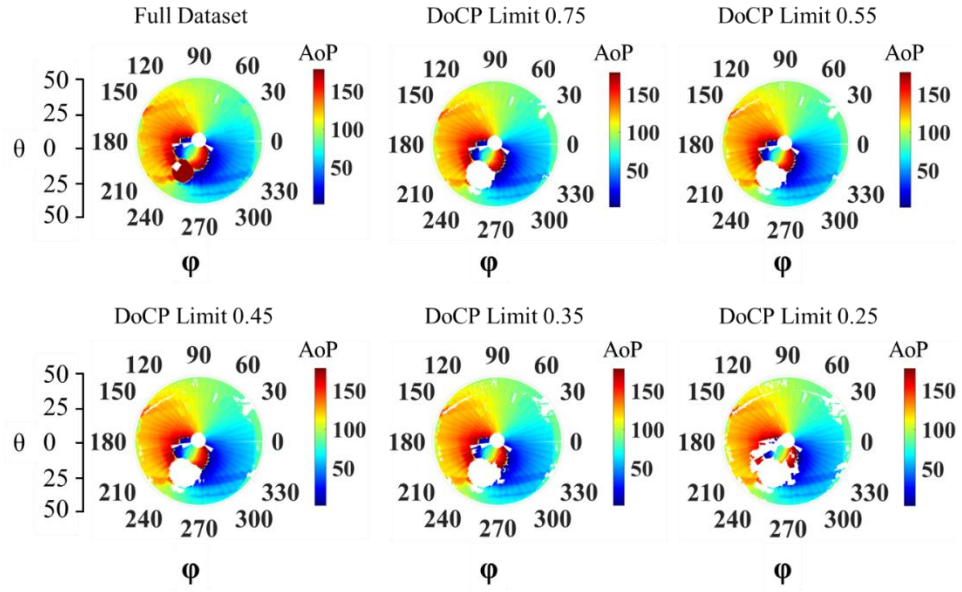


Figure 75: Removed Data for Different DoCP. Data from Swimming Pool Measurement on 04/19/2021 at 2.28pm.

The location prediction error is ~23Km, for minimum azimuth and zenith angle error presented in Figure 74. Detail location prediction error is presented in Figure 76a and b.

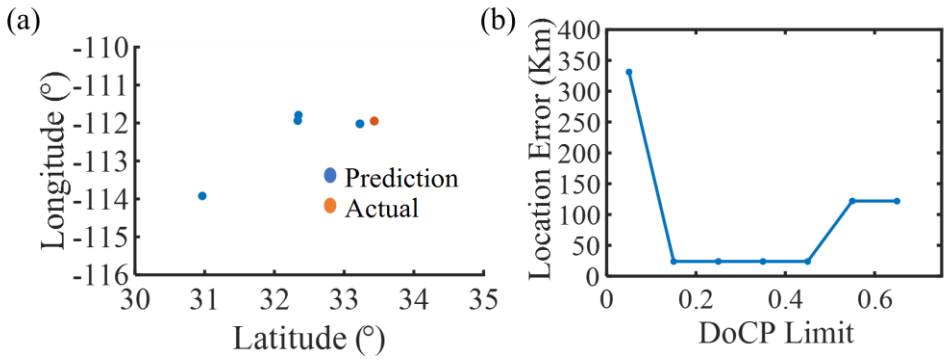


Figure 76: Swimming Pool Measurement on 04/19/2021 at 2.28pm (a) Location Error Shown in Terms of Longitude and Latitude. (b) Location Prediction Error (Km) in Terms of Absolute Distance from the Actual Measurement Point Is Shown for Different DoCP Limits.

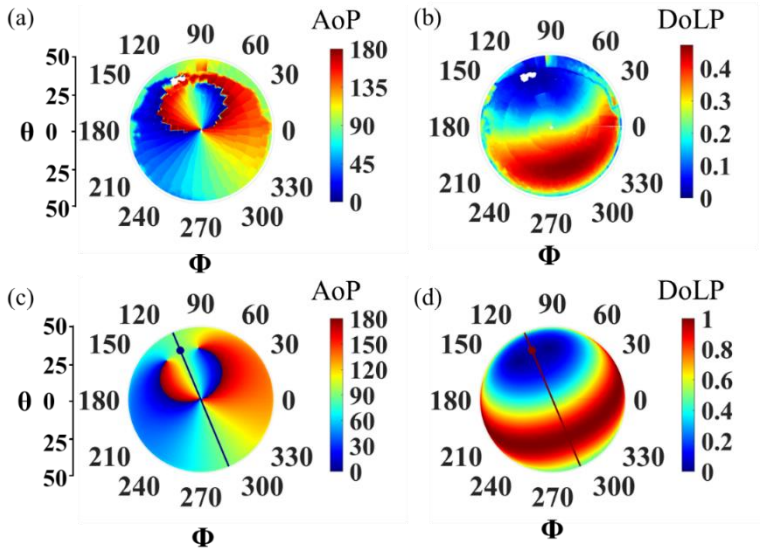


Figure 77: Underwater Polarization Measurements in ASU Swimming Pool on 09/08/2021 at 9.18am (a) Measured AoP (b) Measured DoLP (c-d) Corresponding Simulated AoP and DoLP

We have also performed other measurements in the swimming pool and processed the data similarly.

6.6.5 Underwater Polarization Measurement in Tempe Town Lake

We have also done underwater polarization measurements in Tempe town lake water. To do this, we have utilized a large water tank that can hold a lot of water. We usually put our imaging setup inside the tank and then pump water from the tempe town lake to fill it up. After doing the measurement, we pump out the water from the tank. A detailed image of the setup is shown in Figure 78.

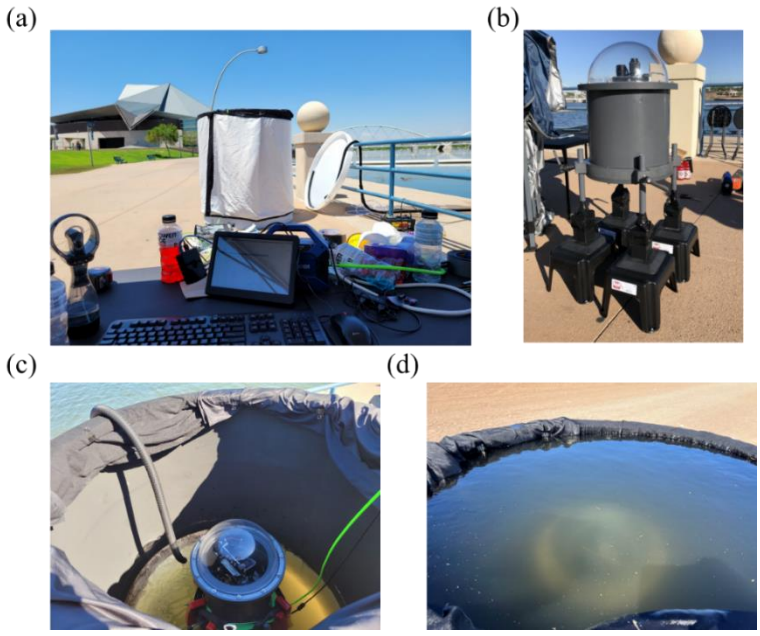


Figure 78: (a) Underwater Measurement Inside a Water Tank with Water from Tempe Town Lake (b) Underwater Housing with Attached Stools to Raise Height (c) Imaging Setup Inside the Water Tank and Pumping Water from the Lake (d after Pumping the Whole Tank with Water from Town Lake. We Can Still See, Barely, the Imaging Setup under the Water.

With this setup, we can see the whole hemisphere from under the water within Snell's window. Also, we can cover some parts outside of the Snell's window. So, overall, we can cover from 0˚ to 70˚ zenith angle and 0˚ to 360˚ azimuth angle.

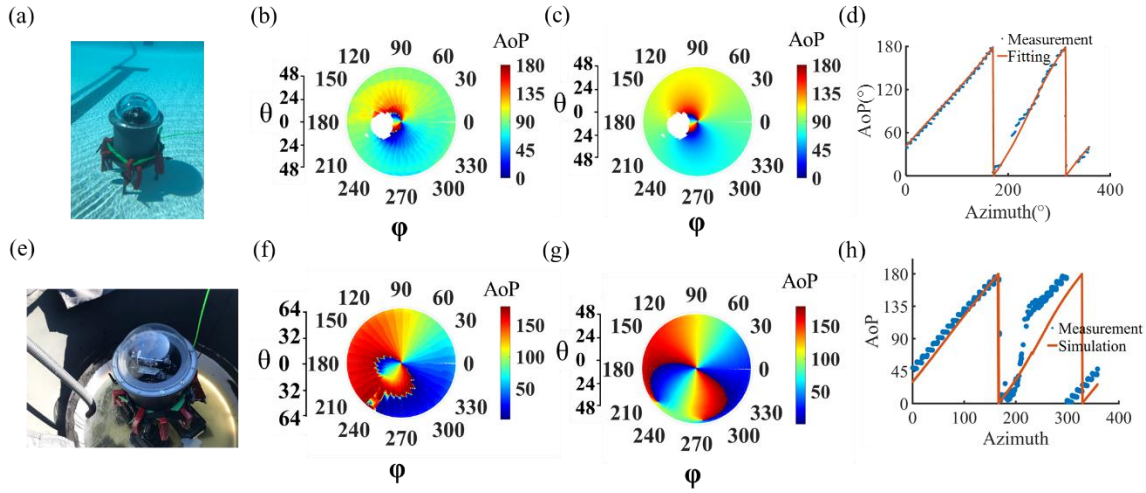


Figure 79: (a) Measurement Setup in the Swimming Pool (b) Measured AoP in the Swimming Pool (c) Simulated AoP in the Swimming Pool (d) Fitting for the Swimming Pool Measurement (e) Underwater Measurement in Tempe Town Lake Water (f) Measured AoP in the Tempe Town Lake Water (g) Simulated AoP for the Tempe Town Lake Water Using the Simple Model Described in Previous Section (h) Fitting Measured AoP (in Tempe Town Lake) with the Simple Model.

We have taken a lot of measurements in the lake water. Figure 79 shows the comparison with the measurement done in the swimming pool. The measured AoP in Tempe town lake is very different from the AoP measured in the swimming pool. Also, for swimming pool measurement, the simulation seems good enough. However, the simulated AoP (with the simple model considering Skylight refraction inside the water) for the Tempe town lake water, Figure 79(g), does not quite match the measured AoP. Figure 79h shows the fitting of measured AoP with the simple underwater polarization model. The fitting is not very

good. For the measurement in tempe town lake, the Actual sun position was: AZ= 256.19˚ and ZEN=84.06˚. The fitted sun position with the simple model: Fit AZ= 248.23˚, Fit ZEN=86.28˚. The fitting error is around 7.97˚ for the azimuth angle and 2.22˚ for the zenith angle. This is obtained after doing all the post-processing and considering the magnetometer offset and off-center camera correction. So, the fitting error is quite large. Why is the fitting error large? Well, the simulation model we used for fitting is not suited for Tempe town lake water. Why? Well, there are many more scatters in the town's lake water than in the swimming pool. It is well depicted in Figure 80. We can clearly see the red weights we have used to drown the underwater housing for swimming pool water, even though the depth is ~20 inches. However, we cannot see the red weights very clearly for Tempe town lake water even though the depth is only ~10 inches.

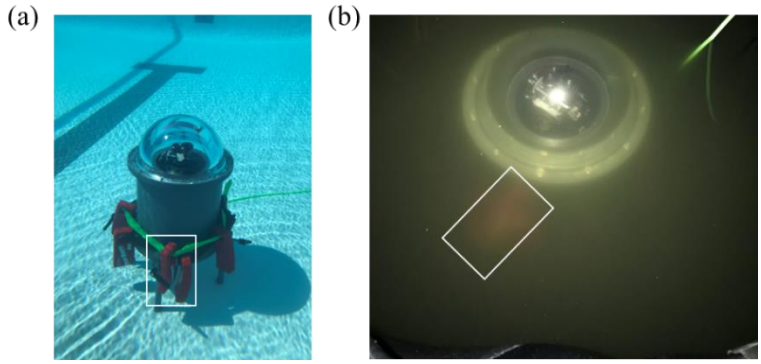


Figure 80: Underwater Imaging Setup Inside (a) Swimming Pool Water (b) in Town Lake Water

Figure 80 clearly indicates that the tempe town lake water is more turbid than swimming pool water, and we need to consider the effect of the increased scatteres in the water and how it can change the light polarization even more.

6.7 Monte Carlo Simulation to Get Underwater Polarization Distribution

Light travel in water is highly dependent on turbidity or relative clarity of the water body. Turbidity is dependent on the presence of different types of scatterer particles in water. The more the scatterer the more turbid the water. Increased turbidity increases the probability of multiple scattering of light. Which (multiple scattering) is not very prominent in skylight due to the small amount of scatterer density in air. So, Monte Carlo method is used to handle the multiple scattering of light, which is very prominent in turbid water.

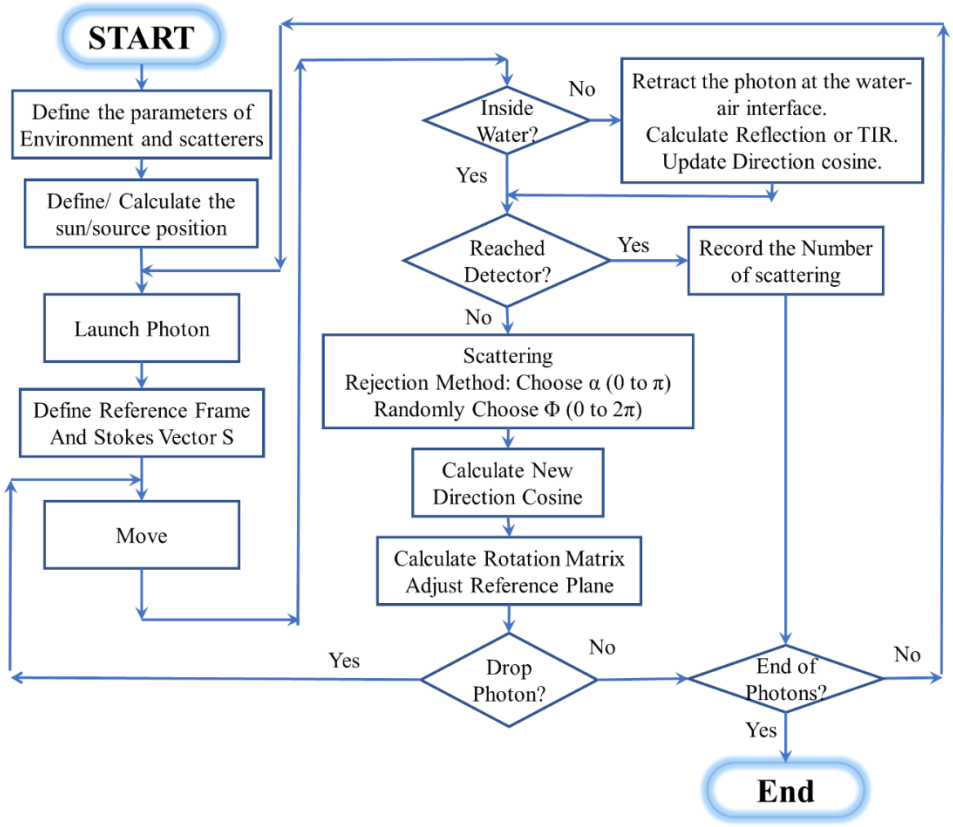


Figure 81: Flow Chart of Different Processes to Simulate the Polarization Mapping From Sunlight Scattering in Underwater with MC Simulation.

The polarized MC simulation steps are presented in Figure 81. The simulation starts by launching a photon from the sun's position. With respect to the meridian plane of the

traveling photon, we define an initial reference frame and the initial stokes parameters. This part is the same for all the photons, as they all come from the sun's position in the sky.

We move the photons, and it doesn't get scattered in the air. It gets refracted in the air-water interface and becomes partially polarized. After entering water body, the light may get scattered depending on the depth of the measurement. If the depth is very large, then the light is scattered more; if the depth is not large (comparable or smaller than the mean free path of the photon), then it may or may not get scattered.

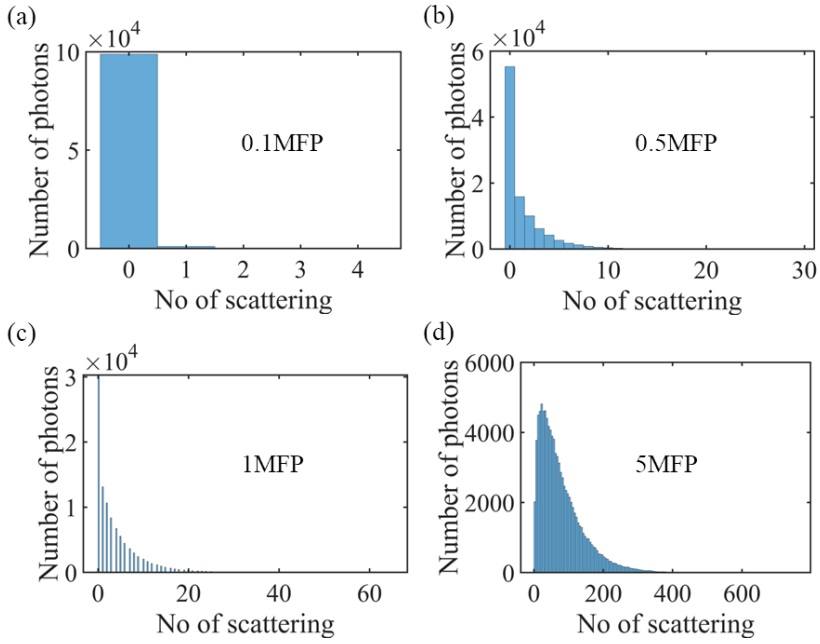


Figure 82: Considering Polystyrene Particles (Index, N=1.572) of Radius 400nm and Concentration 1.1770e+12m^{-3}, Number of Scatterings for Different Depth of Water Indicated in Terms of Mean Free Path (MFP) (a) .1 MFP (b) .5 MFP (c) 1 MFP (d) 5 MFP. For the Particle Size and Concentration Used in the Simulation, the Mean Free Path (MFP) Is 70cm.

When in the water, the light can get scattered a random number of times (within 0 to some preset number). For each scattering event, we need to track the reference frame of the scattered photon and the calculate the polarization state after the scattering. For each scattering event, we randomly choose the scattering angle and the rotation angle from the scattering phase function of mie scatterers with the help of the rejection method[153]. After we find the stokes parameter of the scattered light, we can multiply it with the scattering albedo to account for the scattering and absorption coefficient of the medium. If we can do this calculation for enough photons, then we can eventually find the mapping of the polarization components of light in the water. We have used 100,000 photons here.

Using the Monte Carlo algorithm, we can simulate for an ensemble of photons how many of them will scatter how many times for different depths of water. It is clearly depicted in Figure 82. In this depiction, we considered polystyrene particles (index, n=1.572) of radius 400nm and concentration $1.1770e+12m^{-3}$. We see that, for shallow water, the light encounters almost no scattering events (zero scattering), but for deeper water, the photons scatter many times.

As the particle size density is very important to determine the number of scattering and the probability of each scattering event, we measured the particle size distribution in lake water with DLS (Dynamic Light Scattering). But DLS measurement does not provide the number density for each particle size. To obtain the number density for each size of the particles, we need two pieces of information, one, the mean free path of the water, and two, the size distribution of the particles. We obtained the second one from DLS measurement. And we can calculate the mean free path of the lake water by measuring the change of light intensity

when it goes through the lake water. So, we collected some water during the measurement and conducted mean-free path measurements in the lab. It is detailed in Appendix D.

After going through these calculations, we can get the PSD with the number density, as illustrated in Figure 83a. Furthermore, we can also calculate the light scattering probability and the number of scattering for 1MFP depth of the water with the help of Monte Carlo algorithm explained above, illustrated in Figure 83b.

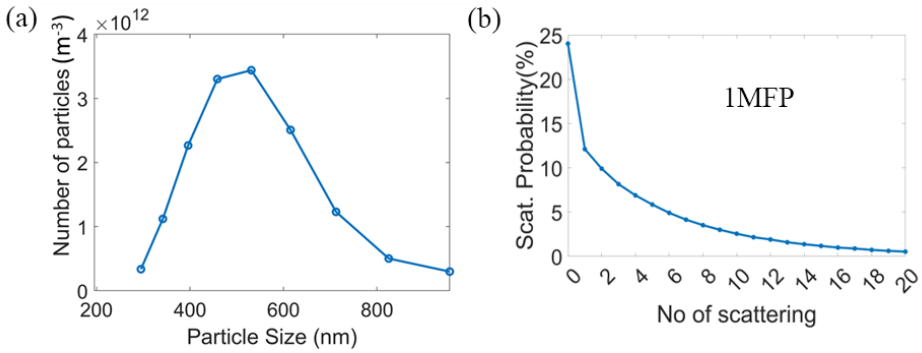


Figure 83: (a) Particle Size Distribution and Number Density in Tempe Town Lake Water. (b) Scattering Probability vs. Number of Scattering for 1MFP Depth of Water.

To understand how light scattering would change the light polarization inside water, we simulated the polarization map for a sun position at Sun position AZ= 246.76˚ and ZEN= 71.83˚. Sunlight is unpolarized light source (S=[1 0 0 0]). For an increasing number of scattering, we can see, from Figure 84, that the polarization pattern is changing. Also, the pattern starts to disappear in the noise after a few scattering events.

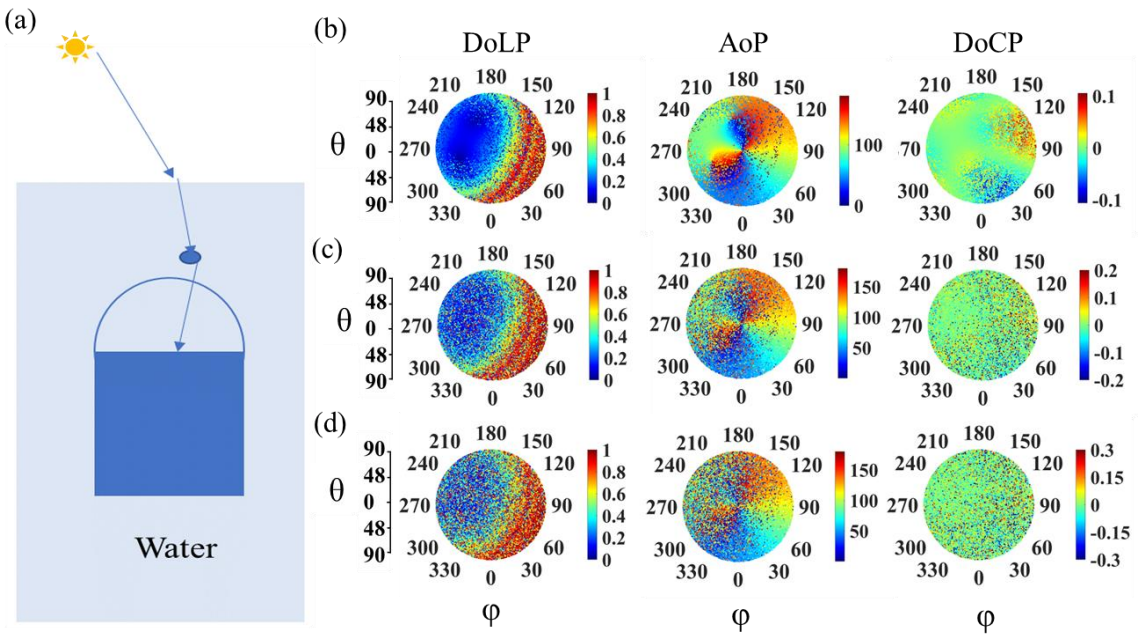


Figure 84: (a) Illustration of Sunlight Going into Water and Scattering Due to Small Scatters Inside. (b) DoLP, AoP and DoCP for 1 Scattering Event (c) DoLP, AoP and DoCP for 2 Scattering Events (d) DoLP, AoP and DoCP for 3 Scattering Events.

If we combine the refracted skylight with the multiple sunlight scattering, we can obtain simulation results that somewhat agree with the measured DoLP and DoCP, illustrated in Figure 85. Some parts of the polarization patterns inside of the Snell's window doesn't quite match with the simulation which may happen because we didn't consider other contributions such as skylight scattering inside the Snell's window. Furthermore, the sunlight scattering inside Snell's window is highly dependent on exact particle size distribution and density of the particles, also the sun and skylight ratio of the day is another unknown parameter in our calculation which I assumed to be 70:30. The particle size distribution and density are measured in our lab with unconventional methods which can lead to erroneous size distribution and density calculation. So, part of the simulated pattern inside of the Snell's window does not match the measurement.

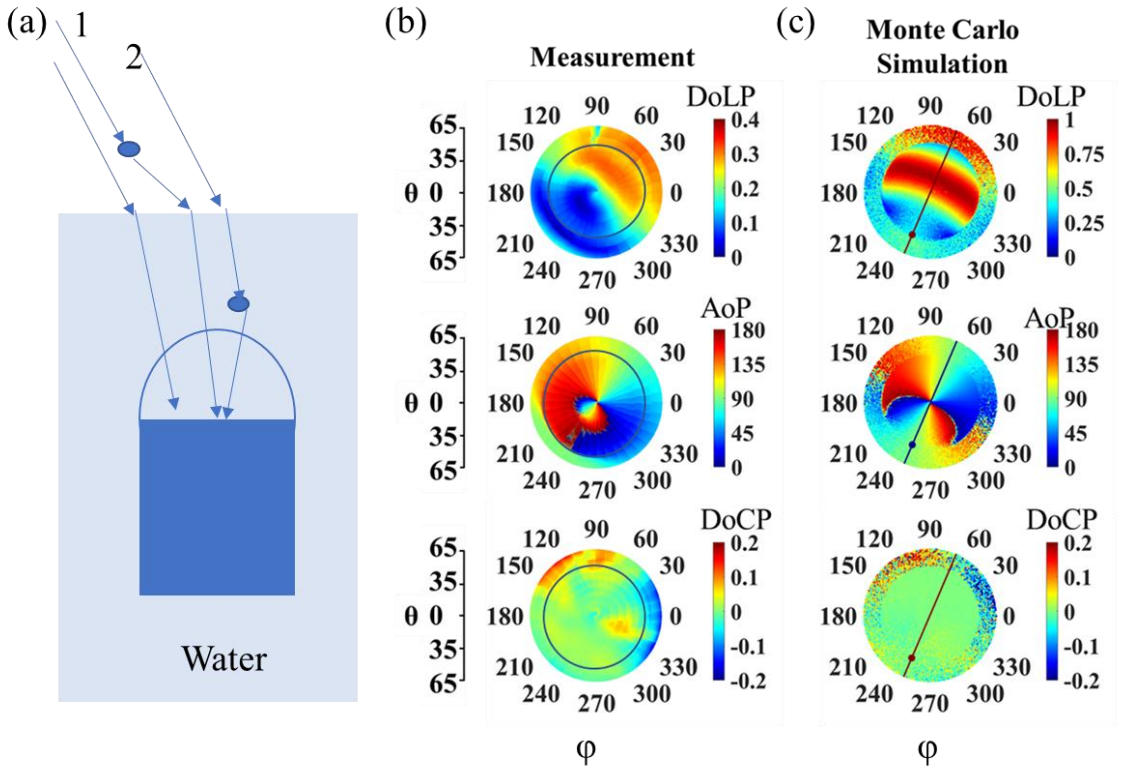


Figure 85: (a) Illustrating Simulation Model Including Skylight Refraction Indicated by 1 and Sunlight Refraction and Scattering Inside Water Indicated by 2 (b) Measured DoLP, AoP, and DoCP in Lake Water Measurement (c) Simulated DoLP, AoP, and DoCP with the Particle Size Density and Distribution Obtained from Lake Water.

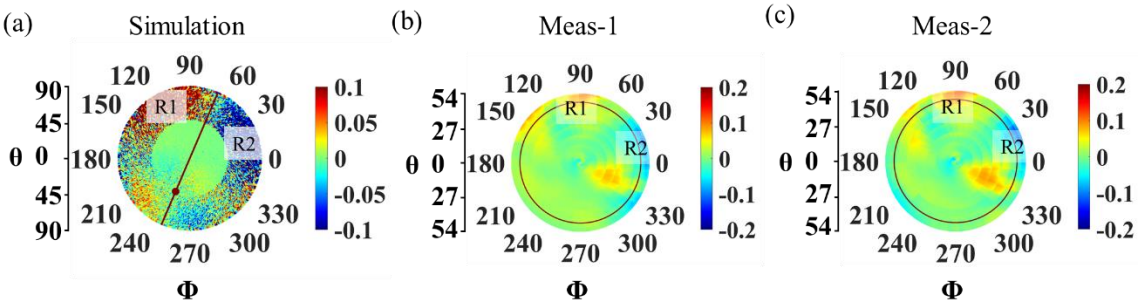


Figure 86: (a) DoCP Obtained Considering Multiple Scattering of Sunlight and Skylight Refraction Inside Water. (b, c) Measured DoCP. The Black Circle Indicates the Snell's

Window. Outside of Snell's Window Shows Pattern Similar to the Simulation Indicated by R1 and R2.

In Figure 86, two measured DoCP, obtained from two different measurements in lake water, show that the detected DoCP in underwater measurement was not an error and it was repeatable. Furthermore, the outside of the Snell's window in measurement matches quite well with the simulation. R1 and R2 indicate the matching part of the simulation and measurement.

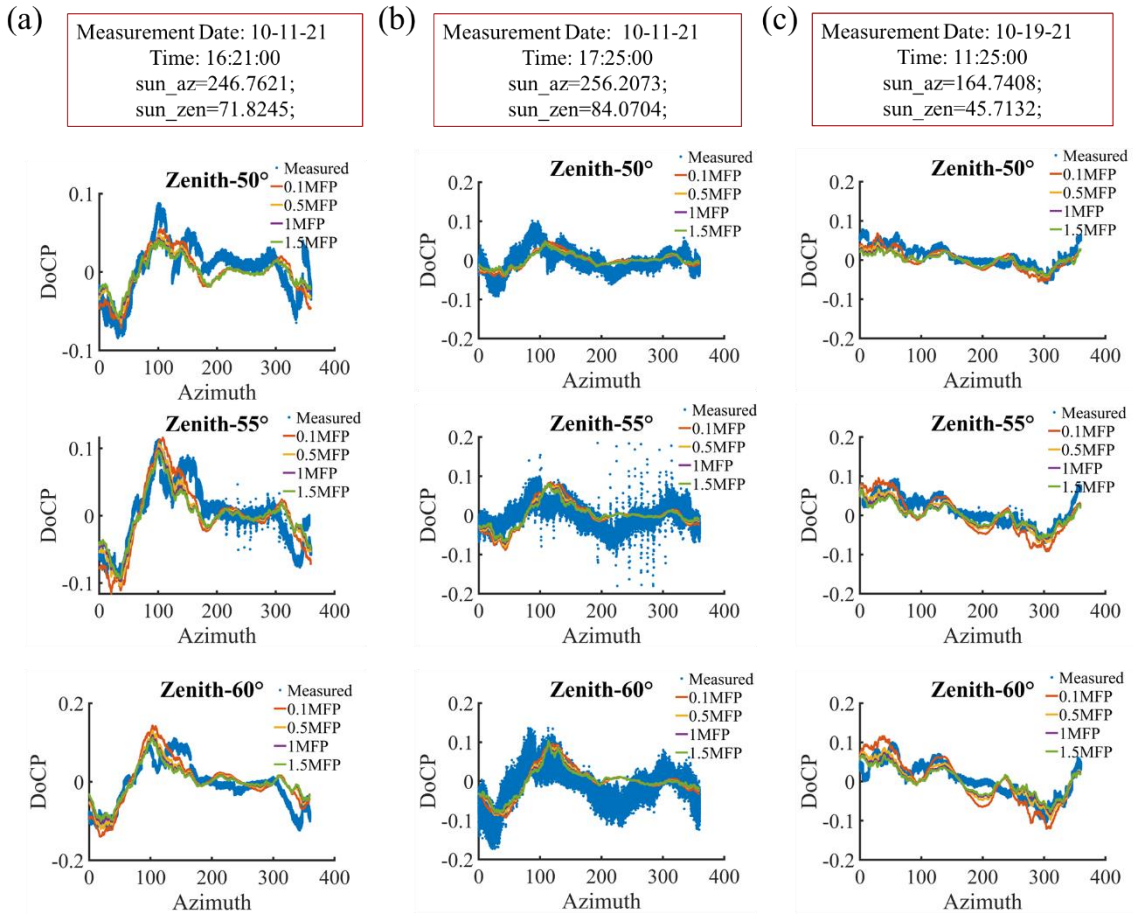


Figure 87: DoCP from Three Different Underwater Measurements and Corresponding Simulations at Zenith 50˚, 55˚, and 60˚. (a, b, c) Are Three Different Measurements Done in 10-11-2021 and 10-19-2021.

Figure 87 shows a detailed line plot showing the match between the simulation and the measured DoCP outside of Snell's window for multiple measurements. It confirms that DoCP outside of Snell's window is generated mostly due to multiple scattering of sunlight, and our model can correctly model it.

Figure 87 shows that including the multiple sunlight scattering with the skylight refraction model has improved the mathematical model. But it is still not enough to explain the measured AoP mapping. To have a complete picture, we need to consider skylight scattering inside the water.

6.8 New Mathematical Model to Describe Light Polarization in Turbid water

This model consists of three different parts. First is the underwater scattering probability calculation from the water turbidity (PSD and particle density). Second, tracking the light polarization change for multiple scattering of sunlight (and skylight) inside water. Furthermore, third, combining the effect of light polarization change due to multiple scattering of sunlight (and skylight) inside water and skylight refraction inside water considering different ratios and probability of each event.

Underwater light scattering is a random process that depends on the sun's position, refractive index ratio of scatterer and water, scatterer's size, scatterer concentration, and water depth. We can emulate this real-world random process with the help of the Monte Carlo method[153–158]. An illustration of the detailed Monte Carlo flowchart is presented in Figure 81. In addition, a detailed calculation method of the mathematical model is presented in Figure 88.

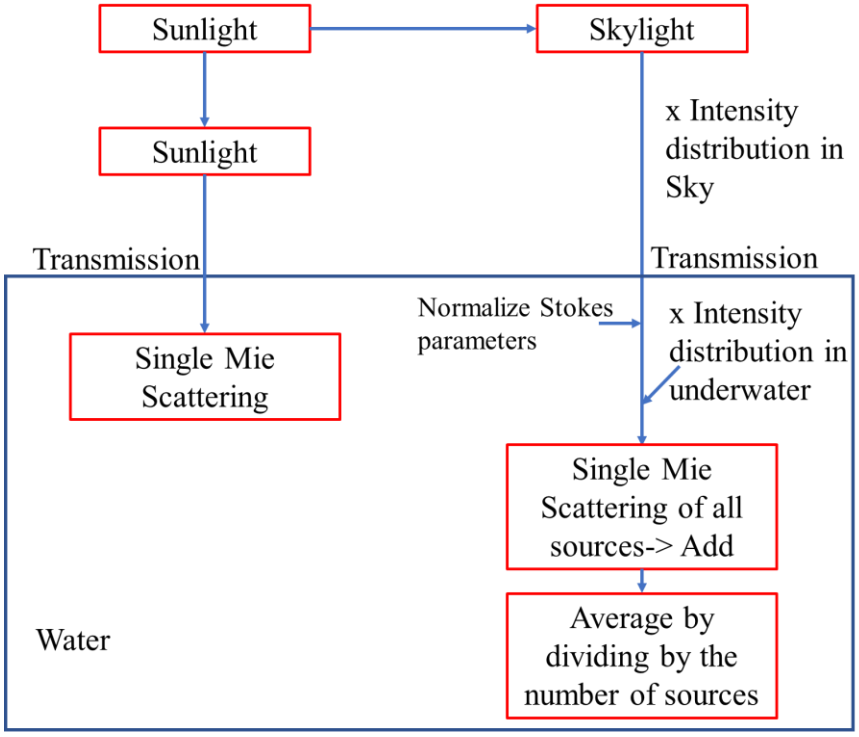


Figure 88: Detailed Calculation Method of the Mathematical Model to Describe Turbid Water Light Polarization

The light polarization states are calculated using muller matrics related to each event, and the calculation for each of these events are done separately (details are provided in the methods section). These processes contribute to forming the light polarization distribution that we see underwater. So, in the end, they are all added up after weighting them with different ratios or probability corresponding to each of these events, as shown below,

$$S = R_{sun}\, P_{sun\_scat}\, S_{sun\_scat} + R_{sky}\,(S_{sky\_scat}\, P_{sky\_scat} + S_{sky\_refract}\,(1\text{-}P_{sky\_scat})) \qquad (57)$$

Here, S = Final Stokes parameter of light; R_{sun} = Sunlight rati; R_{sky} = Skylight ratio; S_{sun_scat} = Stokes parameter after sunlight scattering inside water; S_{sky_scat} = Stokes parameter after Skylight scattering inside water; $S_{sky\_refract}$ = Skylight refracting inside water; P_{sky_scat} =

Skylight scattering probability inside water; P_{sun_scat} = sunlight scattering probability inside water.

We calculate P_{sun_scat} and P_{sky_scat} using the Monte Carlo simulation. This simulation considers the particle size distribution (PSD) and particle density (m^{-3}), representing water turbidity. PSD is obtained from Dynamic Light Scattering (DLS) measurement. DLS measurement does not provide the particle number density. But we can calculate the particle density (m^{-3}) from PSD, Beer-Lambert Law, and Mie scattering theory as explained in the previous sections, Figure 117 and Figure 83.

Skylight scattering inside water:

Stokes parameter of light after skylight (coming from different zenith and azimuth angle) gets scattered,

$$S_{sky\_scat}(\theta,\varphi) = \int_{p=1}^{N}\int_{\varphi'=0}^{2\pi}\int_{\theta'=0}^{\frac{\pi}{2}} P_p M^p(\theta,\varphi,\theta',\varphi')\, S_{Sky\_refract}(\theta',\varphi')\, d\theta' d\varphi' dn$$

$$= \sum_{p=1}^{N}\sum_{\varphi'=0}^{2\pi}\sum_{\theta'=0}^{\frac{\pi}{2}} P_p M^p(\theta,\varphi,\theta',\varphi')\, S_{Sky\_refract}(\theta',\varphi')$$

$$= \sum_{p=1}^{N}\sum_{\varphi'=0}^{2\pi}\sum_{\theta'=0}^{\frac{\pi}{2}} P_p \left[R(\sigma')\left\{M(\theta,\varphi,\theta',\varphi',p)\left(R(\sigma)\, S_{Sky\_refract}(\theta',\varphi')\right)\right\}\right]$$

Here, $S_{Sky\_refract}(\theta',\varphi')$ = Stokes parameters of Skylight sources for different zenith and azimuth angles; $S_{sky\_scat}(\theta,\varphi)$= Stokes parameters after skylight scattering inside the water; P_p = Scattering probability for each size of particles; p = Pseudo particle size;

$M(\theta, \varphi, \theta', \varphi', p) = M(\mu, p)$=Muller Matrices for each particle size; $R(\sigma')\ and\ R(\sigma)\ are$ the rotation matrices; and μ = scattering angle.

Also,

$$M(\mu,p) = \begin{bmatrix} M_{11}(\mu,p) & M_{12}(\mu,p) & 0 & 0 \\ M_{21}(\mu,p) & M_{22}(\mu,p) & 0 & 0 \\ 0 & 0 & M_{33}(\mu,p) & M_{34}(\mu,p) \\ 0 & 0 & M_{43}(\mu,p) & M_{44}(\mu,p) \end{bmatrix} =$$

$$\begin{bmatrix} M_{11}(\mu,p) & M_{12}(\mu,p) & 0 & 0 \\ M_{12}(\mu,p) & M_{11}(\mu,p) & 0 & 0 \\ 0 & 0 & M_{33}(\mu,p) & M_{34}(\mu,p) \\ 0 & 0 & -M_{34}(\mu,p) & M_{33}(\mu,p) \end{bmatrix}$$

Where, $M_{11} = M_{22} = \frac{1}{2}(|m_1|^2 + |m_2|^2)$; $M_{12} = M_{21} = \frac{1}{2}(|m_2|^2 - |m_1|^2)$; $M_{33} = M_{44} = \frac{1}{2}({m_1}^* m_2 + m_1 {m_2}^*)$; and $M_{34} = -M_{43} = i\frac{1}{2}({m_1}^* m_2 - m_1 {m_2}^*)$. Here, m_1 and m_2 are the scattering functions representing the far-field solution of maxwell's equations. They were obtained using the Mie theory[71].

<u>Sunlight scattering inside water:</u>

Sunlight refracts inside water, and that refracted sunlight is scattered. Stokes parameter,

$$S_{sun\_scat}(\theta,\varphi) = \int_{p=1}^{N} P_p M^p(\theta,\varphi,\theta_{sun},\varphi_{sun})\, S_{Sun\_refract}\left(\theta_{sun\_refract},\varphi_{sun}\right) dn$$

$$= \sum_{p=1}^{N} P_p M^p(\theta,\varphi,\theta_{sun},\varphi_{sun})\, S_{Sun}\left(\theta_{sun\_refract},\varphi_{sun}\right)$$

$$= \sum_{p=1}^{N} P_p\, [R(\sigma')\, \{M(\theta,\varphi,\theta_{sun},\varphi_{sun},p)\, \left(R(\sigma)\, S_{Sun}\left(\theta_{sun\_refract},\varphi_{sun}\right)\right)\}]$$

Here, $S_{Sun\_refract}\left(\theta_{sun\_refract},\varphi_{sun}\right)$ = Stokes parameters of sunlight after refracting inside water and $S_{sun\_scat}(\theta,\varphi)$= Stokes parameters after sunlight scattering inside water.

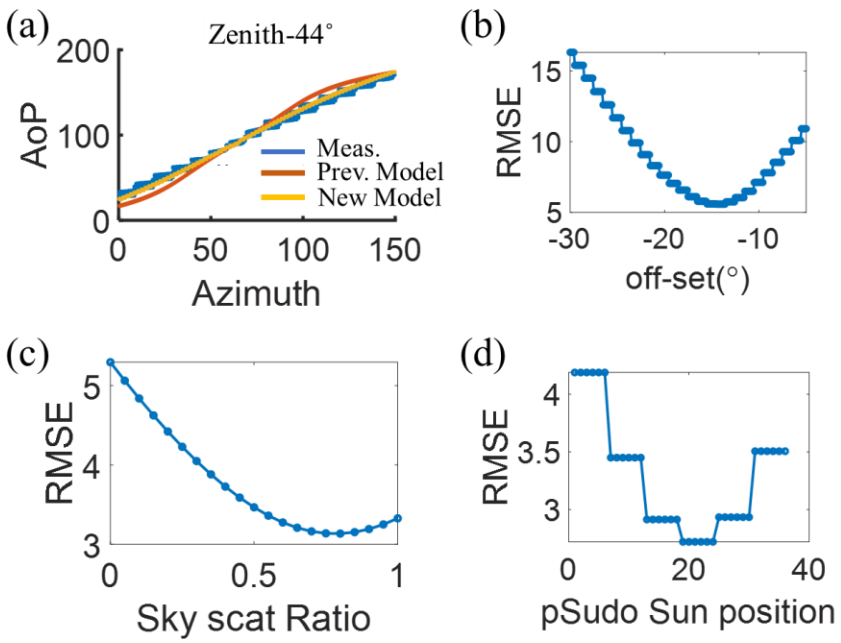


Figure 89: (a) Fitting AoP after Adding Skylight Scattering in the Model (b) Changing the offset to Find the Best Fitting (c) Changing the Sky Scattering Ratio to Find Best-Fitting (d) Change the Sun Position to Find the Best Fitting

After adding the skylight scattering with skylight refraction and sunlight scattering inside water, we have better match between the simulation and measurement, as shown in Figure 89a. During the underwater measurement there are some mechanical and unexpected errors added to the recorded directions (azimuth and zenith angles) of measurements. To remedy that, we consider some offset which includes the magnetometer offset and off-center camera error. We change the offset and calculate the root mean square error (RMSE) to compare the similarity between measurement and simulation. When RMSE is minimum that is the offset for the measurement, Figure 89b. Also, we have a parameter that can be used as a handle to change the simulation result, the skylight scattering rate. This skylight scattering rate depends on the turbidity of water. However, the turbidity of water may not be known to us. So, we change the skylight scattering ratio and find the minimum RMSE

for the best fit. After these two steps, we run the simulation for different sun positions and find the sun position that produces minimum RMSE compared to measured AoP. That is our fitted sun position.

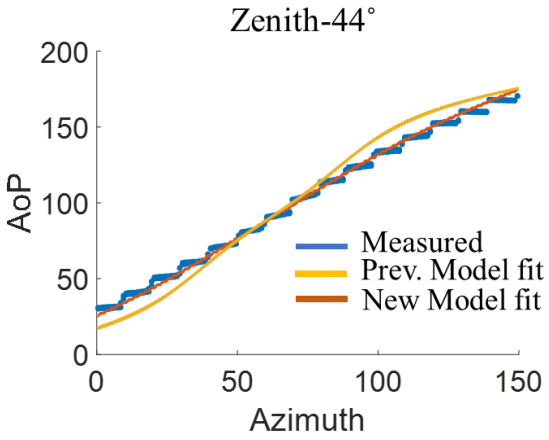


Figure 90: Measured AoP on 10-11-21 and Fitting It with Previous Simple Models (Considering Only Skylight Refraction) and New Model at Zenith Angle 44.

Utilizing this method, we conducted fitting on different measured data. A comparison in terms of fitting the measured data with the previous model (a simple skylight refraction model) and the new model considering the sunlight scattering, skylight refraction, and skylight scattering is shown in Figure 90.

From, Figure 90, we can see that the new model provides better fitting compared to the simple skylight refraction model. We have done this for multiple measured AoP, and the fitting error is smaller for the new model for all of the cases, illustrated in Figure 91.

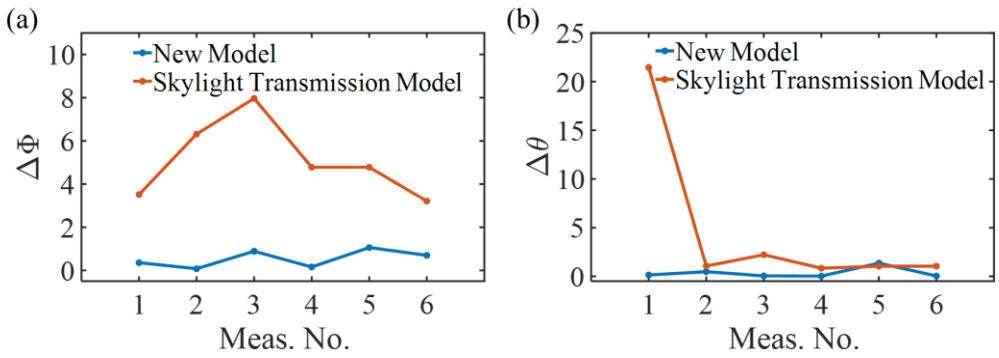


Figure 91: Measured AoP (Tempe Town Lake) Fitting Error (a) Azimuth Angle (b) Zenith Angle. Two Different Simulation Models are Considered for Fitting: The New Model and Previous Simple Model. The Error in the Sun position is Very Small, and We Can Achieve ~100km Location Error Detection with the Fitted Sun positions Shown Here.

6.9 Summary

In summary, we have demonstrated navigation capability with skylight polarization mapping, underwater polarization mapping from swimming pool water, and more turbid Tempe town lake water. Also, we have established a new model to describe the changes in the polarization map considering sunlight scattering, skylight refraction, and skylight scattering. Overall, we could achieve a smaller error in sun position fitting with the new model, enabling us to achieve a navigation error of around 100Km in the turbid water. Furthermore, the simulation model developed here is flexible and can be tailored for different types of water turbidity and weather conditions.

CHAPTER 7

DETECTING MIRROR SOILING LEVEL UTILIZING LIGHT SCATTERING BY NANOPARTICLES AND POLARIMETRIC IMAGING FOR CSP POWER PLAN MAINTENANCE

Heliostat mirrors are used, in concentrated solar power (CSP) plants, to redirect and focus sunlight on receiver tubes to produce electricity. Due to wind, dust storms, and many other natural environmental effects, these mirrors become soiled with nanoparticles. Therefore, sequential cleaning and maintenance are required to maintain maximum reflectivity of the mirrors. This dissertation demonstrates a fast and field deployable inspection method to measure the heliostat mirror soiling levels detection based on polarization images in regular daylight settings. Under sunny and clear sky conditions, accurate measurement of reflection efficiency (with error ~1%) for mirrors with different soiling levels is achieved.

7.1 Introduction

Mirror soiling could significantly reduce the optical reflection efficiency (below 90%) of heliostats. The heliostat mirror cleaning cost is ~25% of the total Operation and Maintenance cost of the Concentrated Solar Power (CSP) plant [89]. Currently, the soiling levels are usually determined by manual measurement using hand-held reflectometers, which are slow and only cover a small portion of the mirror surface. A fast and field-deployable method for detecting the soiling levels of heliostat mirrors is highly desirable for planning CSP field collector cleaning to maintain low optical loss with minimal

cleaning cost. Moreover, it can also facilitate the study of mirror soiling patterns in CSP fields. Here we show that polarization images of heliostat mirrors, esp. the degree of linear polarization (DoLP) images, can be used under sunny and clear sky conditions to measure the mirror soiling levels. We performed outdoor tests and determined mirror soling levels up to reflection efficiencies of 98%, which is deemed challenging by conventional imaging-based methods[90,91] without needing high accuracy sensors[92]. The polarization imaging method can be deployed in CSP collector fields for in-situ characterization of the heliostat mirror soiling levels with high throughput and low cost.

7.2 Mirror Soiling Detection

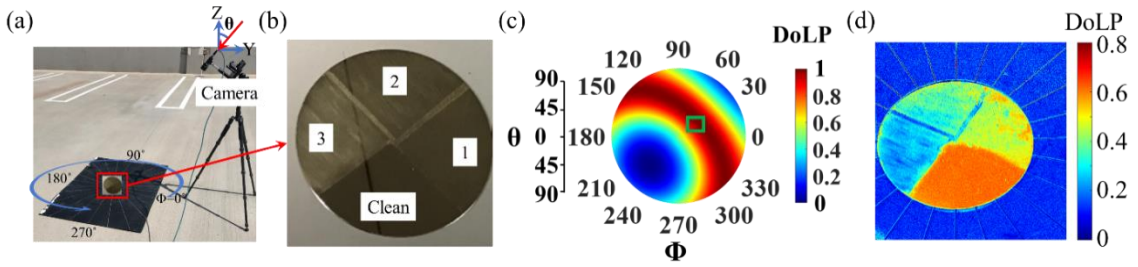


Figure 92: (a) Outdoor Measurement Setup with Polarization Sensitive Camera and Mirror Sample. (b) Prepared Soiled Mirror Sample with Different Soiling Levels in Each Quadrant. (c) Degree of Linear Polarization (DoLP) Mapping of the Sky Seen from the Mirror. (d) DoLP Image of the Soiled Mirror Sample

For mirror soiling detection, we can use the degree of linear polarization (DoLP) of the light coming from the soiled mirror. To demonstrate that, we have set up a polarization-sensitive camera to take pictures of the mirror from different azimuth (Φ) and zenith (θ) angles. The setup is illustrated in Figure 92(a). For this demonstration, we prepared a soiled mirror with the small sand/dust particles collected from a nearby riverbank (Tempe Town

Lake). The prepared soiled mirror, Figure 92(b), has four different regions with different soiling levels. The number sequence on Figure 92(b) represents the increasing soiling levels for each section.

As we know, the sunlight entering our atmosphere gets scattered by small particles (Rayleigh scatterer) and produces a partially polarized skylights. We can model this polarization pattern of the sky using the Rayleigh scattering theory. One such simulated polarization pattern, DoLP, of the sky observed through the mirror at the time of the measurement is shown in the inset of Figure 92(c). The DoLP map is shown in the same coordinate system as the measurement setup. We can look at any portion of the sky through the mirror and see the reflected skylight and the scattered sunlight from that direction by changing the azimuth (Φ) and zenith (θ) angle of the imaging setup.

On the soiled mirror, there are many small particles that can scatter the incoming light and decrease the degree of polarization of that light. Which means if there are large number of particles on the mirror then the incoming light is scattered more and the degree of linear polarization of the light is decrease more compared to a mirror with smaller number of particles (i.e., smaller soiling). Consequently, we try to position the imaging setup towards the maximum DoLP region to obtain maximum contrast between two different soiling levels. One such DoLP image is shown in Figure 92(d). Here, we can clearly see the difference in DoLP in different soiled sections of the mirror. The maximum DoLP is observed in the clean part, and it gradually decreases with increasing soiling levels, as discussed before. In the following sections, we establish a relationship between the observed DoLP and the soiling level (or reflection) of the mirror by combining outdoor

measurements and a mathematical model and eventually predict the soiling level from this relationship for mirrors with unknown soiling levels.

7.3 Mathematical Model

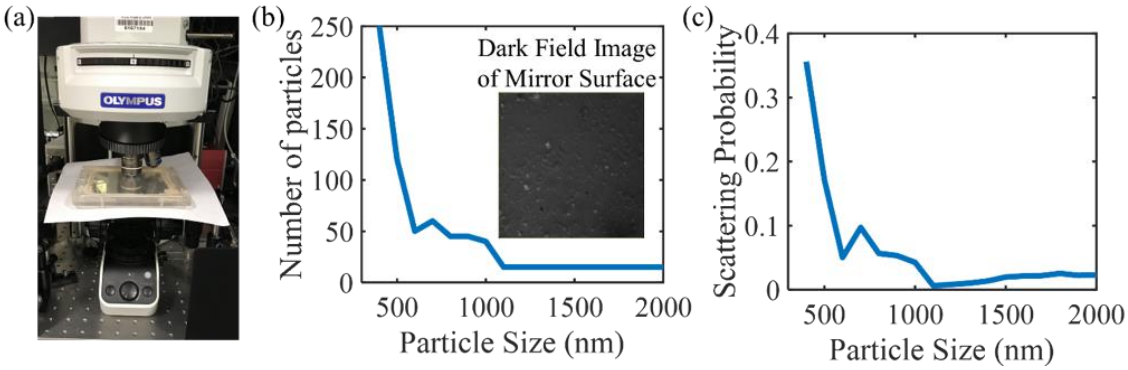


Figure 93: (a) Mirror Soiling Particle Imaging System. (b) Number of Particles vs. Particle Size (nm). Inset Showing the Dark Field Image of the Mirror Surface. (c) Scattering Probability for Different Particle Size. Dark Field Image was Taken by Me, and the Particle Size Distribution was Obtained by my Labmate Ahmad with Image Processing.

To understand how the soiled mirror changes the light polarization, we need to understand how the small particles on the mirrors scatter light. For that, we need to know the distribution of the particles on the mirror. We have studied the particles on the mirror through dark-field imaging using a microscope, illustrated in Figure 93a. The dark field images were processed using a Matlab Image processor to identify different particles and determine their sizes. From this image analysis, we obtained the particle size distribution on the mirror, illustrated in Figure 93b. A dark field image of the mirror surface is shown in the inset of Figure 93b showing these small particles. Different particles have different scattering capabilities. It depends on the wavelength of light, the particles' size and index, and the surrounding environment. All of these are bound together in the well-established

Mie scattering theory [71]. The wavelength of light considered here is 530nm, the same as the working wavelength of our polarization-sensitive camera. The particle index is 1.57, and the mirror glass is 1.52. The attenuation coefficients for each of the particles are calculated from the scattering and absorption cross-sections using the Mie scattering theory. Furthermore, considering both the attenuation coefficient (U_t) and the number (N_D) of each particle size, we have calculated the scattering probability (P_i) of each of these particles, shown in Figure 93c, as follows

$$P_i = \frac{U_{ti} N_{Di}}{\sum_{i=1}^{N} U_{ti} N_{Di}} \tag{58}$$

The scattering probability of each particle size is an important parameter necessary to establish the mathematical model for the mirror polarization pattern. The core consideration for this mathematical model is that if we put a soiled mirror outside lying on the ground in a sunny day, the unpolarized sunlight will be scattered by the small particles, and the partially polarized skylight will be reflected from the mirror surface, as illustrated in Figure 94a. The final polarization pattern seen from the soiled mirror is a combination of these two effects. Sunlight reflection, which produces overexposed images, and the skylight scattering by the particles are ignored as they produce less effects in the overall polarization map but comes with a very large computational drag.

7.3.1 Sunlight Scattering

When sunlight falls on the particle on the mirror, it gets scattered. And some part of the scattered light will be reflected from the mirror, and this reflected scattered light would interact with the scattered light, illustrated in Figure 94a inset.

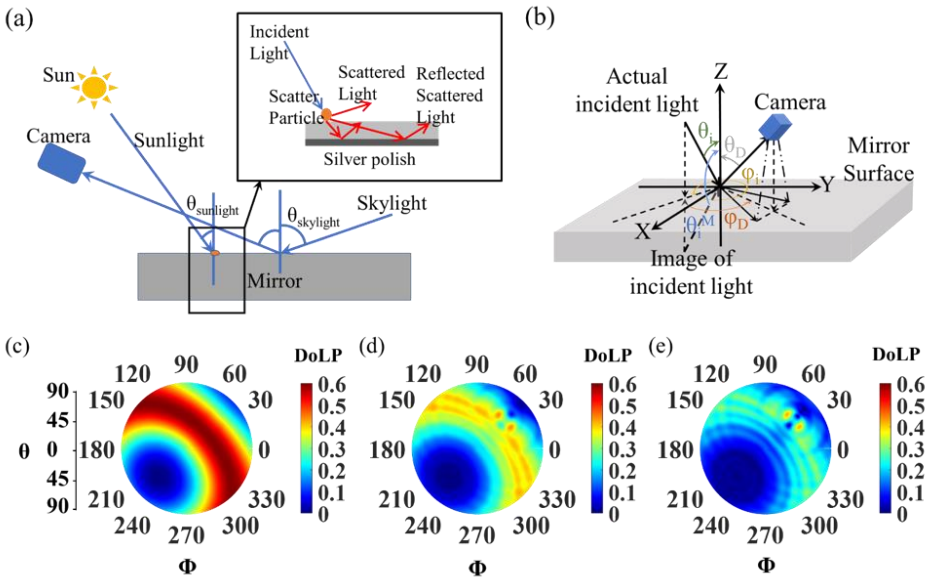


Figure 94: (a) Schematic of Mathematical Model Description. (b) Coordinate System for the Mie Scattering Simulation of Sunlight. XY Plane is on the Mirror Surface. (c-e) DoLP Map Seen from the Mirror for Mirror Having (c) 100% Reflection (d) 90% Reflection (e) 80% Reflection.

The interaction between scattered light and reflected scattered light can happen either coherently or incoherently, depending on the source's coherence length and the mirror glass's thickness. Here, we consider that the coherence length of sunlight is very small compared to the mirror thickness. So, the reflected scattered light will interact with scattered light incoherently and we would see an increase in the intensity of the light when they interact (i.e., simple addition will suffice, no phase needs to be considered here). For simplicity, we can consider a mirror image of the particle and the incoming light on the back of the mirror to mimic the reflected scattered light, indicated as the image of the incident light on Figure 94b. Furthermore, as the distance between the camera and the mirror surface is larger than the glass thickness, we can ignore the glass thickness and

propagation of scattered light inside the glass. The scattered light and the reflected scattered lights are added (incoherent interaction) to obtain the polarization map for the sunlight scattering for each particle size. The total polarization pattern is obtained by weighted summation (considering scattering probability, Figure 93c) of the polarization patterns of each particle in the particle size distribution.

The light scattering pattern is highly dependent on the position of the light source, and for sunlight scattering, it is the sun position. To simulate the sunlight scattering event and obtain the polarization pattern for sunlight scattering from the particles on the mirror, we have considered the coordinate system illustrated in Figure 94b. The XY plane is the mirror surface containing the particles, and Z-axis is the surface normal to the mirror. Incident light coming at a zenith angle, θ_i and azimuth angle, φ_i; gets scattered, and the camera at zenith angle, θ_D and azimuth angle, φ_D; collects the scattered light. To account for the reflected scattered light, an image of the incident light source is considered at the zenith angle, $\theta_i^{M,}$ and azimuth angle, φ_i.

The Stokes parameter of light, S, which properly describes the polarization state, is changed after scattering from a particle. The change in light polarization after scattering at different scattering angle, α, can be calculated using the following equation[71],

$$S_S = R(-\psi)M^p(\alpha)R(\psi)\,S_I \tag{59}$$

Here, S_I is the stokes parameter containing the polarization information of the incident light. Scattering angle, α, is calculated from the zenith and azimuth angles of the incoming and scattered lights as follows,

$$cos\alpha = sin\theta_i\, sin\theta_D\, cos(\varphi_i - \varphi_D) + cos\theta_i\, cos\theta_D$$

After scattering, we obtain the stokes parameter of the scattered light S_S. The scattering matrix, $M^p(\alpha)$, works as a muller matrix that describes the change in the polarization state of light after scattering from a particular particle. It is dependent on the properties of the particle i.e. size of particle, index of particle, and also the light scattering angle. As it is different for each particle so we indicate it with p notation. For a spherical particle, it is defined as below[71],

$$M^p(\alpha) = \begin{bmatrix} M_{11}(\alpha) & M_{12}(\alpha) & 0 & 0 \\ M_{21}(\alpha) & M_{11}(\alpha) & 0 & 0 \\ 0 & 0 & M_{33}(\alpha) & M_{34}(\alpha) \\ 0 & 0 & -M_{34}(\alpha) & M_{33}(\alpha) \end{bmatrix}$$

Where, $M_{11} = M_{22} = \frac{1}{2}(|m_1|^2 + |m_2|^2)$, $M_{12} = M_{21} = \frac{1}{2}(|m_2|^2 - |m_1|^2)$, $M_{33} = M_{44} = \frac{1}{2}(m_1{}^* m_2 + m_1 m_2{}^*)$, $M_{34} = -M_{43} = i\frac{1}{2}(m_1{}^* m_2 - m_1 m_2{}^*)$. m_1 and m_2 are the scattering functions representing the far-field solution of maxwell's equations. They were obtained using the Mie theory[159].

And $R(\psi)$ is the rotation matrix[71], shown below, and ψ is the rotation angle.

$$R(\psi) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & cos2\psi & sin2\psi & 0 \\ 0 & -sin2\psi & cos2\psi & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

After sunlight is scattered, some part of the scattered light will be reflected from the mirror, which would interact with the scattered light, illustrated in Figure 94 inset. The interaction can happen either coherently or incoherently, depending on the source's coherence length and the mirror glass's thickness. Here, we consider that the coherence length of sunlight is

very small compared to the mirror thickness. So, the reflected scattered light will interact with scattered light incoherently, and we would see an increase in the intensity of the light when they interact (i.e., simple addition will suffice, no phase needs to be considered here). For simplicity, we can consider a mirror image of the particle and the incoming light on the back of the mirror to mimic the reflected, scattered light, indicated as the image of the incident light on Figure 94b. Furthermore, as the distance between the camera and the mirror surface is larger than the glass thickness, we can ignore the glass thickness and propagation of scattered light inside the glass. The scattered light and the reflected, scattered lights are added (incoherent interaction) to obtain the polarization map for the sunlight scattering for each particle size. The total polarization pattern is obtained by weighted summation (considering scattering probability, Figure 93c) of the polarization patterns of each particle in the particle size distribution. So, the overall tokes parameter for sunlight scattering is,

$$S_{sun\_scat}(\theta,\varphi) = \int_{p=1}^{N} P_p\, S_S(\theta_D,\varphi_D)\, dn = \sum_{p=1}^{N} P_p\, S_S(\theta_D,\varphi_D)$$

$$= \sum_{p=1}^{N} P_p\, [[R(\psi')\,\{M^p(\alpha)\,(R(\psi)\,S_{Sun}(\theta_i,\varphi_i)\,)\}]$$

$$+\ [R(\psi')\,\{M^p(\alpha)\,(R(\psi)\,S_{Sun}(\theta_i^M,\varphi_i)\,)\}]]$$

Here, $S_{sun\_scat}(\theta,\varphi)$ is the stokes parameters after sunlight scattering, P_p is the scattering probability for each size of particles, p is particle size counter.

7.3.2 Skylight Reflection

The skylight polarization map is obtained from the Rayleigh scattering theory. It is broadly explained in Chapter 6.

7.4 Mirror Soiling Prediction Methodology

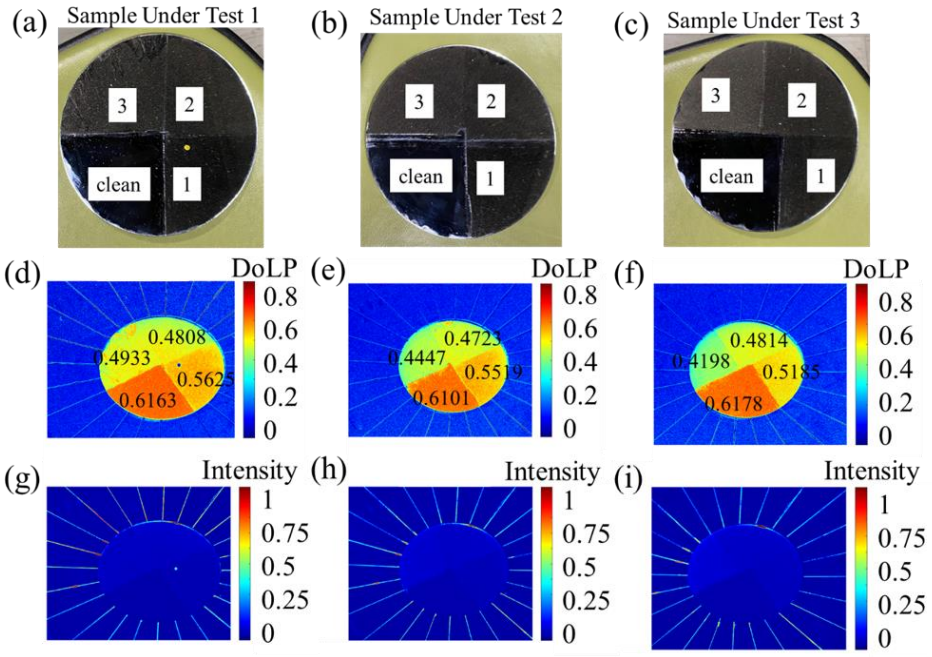


Figure 95: (a-c) Soiled Mirror Samples with Unknown Soiling Levels in Each Quadrant. (d-f) Dolp Image of the Mirror Samples (g-i) Intensity Image of the Samples.

To demonstrate the capability to predict soiling, we measured the DoLP of three more samples using our measurement setup illustrated in Figure 92a. The samples were made from the same dust collected from the local riverbank (Tempe Town Lake) and had four different parts in each mirror same as the previous sample. So, it is safe to say that the particle size distribution and scattering probability of the particles on these samples are also the same as presented in Figure 93b, c. The soiling levels of the samples were unknown during the measurement. The measurement was done on a rooftop under clear sky (no

clouds) conditions, and the looking direction (Φ, θ) was at the maximum DoLP region of skylight mapping. The sample images (sample under test 1, 2 and 3), processed DoLP image, and the intensity images from the measurement are shown in Figure 95. The DoLP images of the samples shows decrease in DoLP values with the increase of soiling level, similar to the earlier demonstration in Figure 92d. On the other hand, the intensity images, Figure 95g-i, show very small change for different soiling levels.

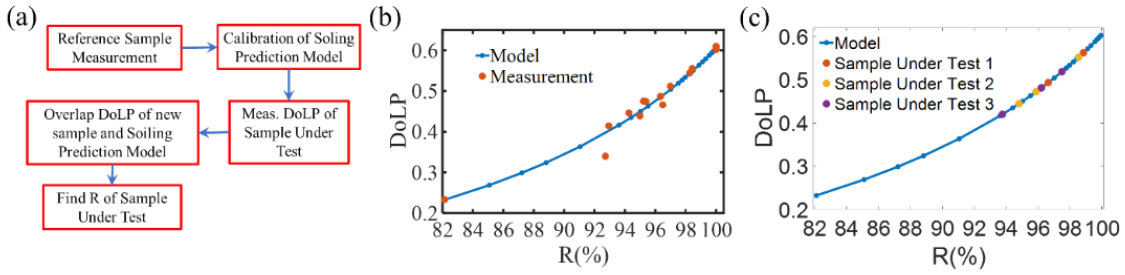


Figure 96: (a) Overall Procedure of Soiling Prediction (b) Overlap of Simulation and Measured DoLP vs Reflection of Reference Soiled Mirror (c) Average DoLP of the Three Soiled Mirrors Overlapped on the DoLP vs Reflection Simulation.

Now, to quantify the soiling levels of the samples in, Figure 95, we measured the DoLP of some reference samples with known soiling levels. This DoLP vs reflection, R, (or soiling level) of the known samples are used to calibrate our mathematical model to account for different weather condition and the ratio of skylight and sunlight during the measurement. The whole process of soiling level prediction is described in the flow chart in Figure 96a. The DoLP vs R curve for the reference samples and the model is shown in Figure 96b. After the calibration, the mathematical model generated DoLP vs R curve is used to find the reflection efficiency of the different soiled mirror sections using the DoLP found in Figure 95d-f. This is shown in Figure 96c. The predicted median soiling level of SUT1 is

98.8415%, 96.1894% and 96.6317%; for SUT2 they are 98.5304%, 95.8734%, 94.8042% and for SUT3 they are 97.4889%, 96.2117%, 93.7566%.

7.5 Accuracy Analysis

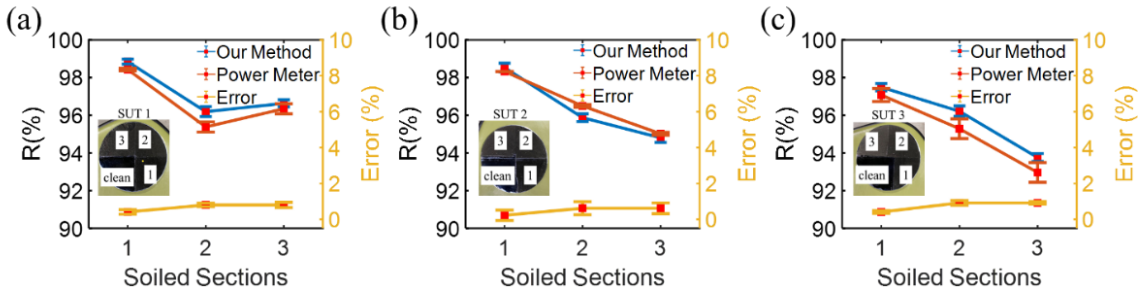


Figure 97: Soiling or Reflection Measured with Our Method and Large Area Reflection Measurement with Power Meter and Error for (a) Sample under Test 1 (SUT 1) (b) SUT 2 (c) SUT 3

To evaluate the accuracy of the predicted reflection or soiling levels of the mirror samples, we directly measured the reflection of the samples using a power meter and an unpolarized light source in the lab. The reflection of all three samples obtained by our polarization method and direct reflection measurement is illustrated on the same plot for comparison in Figure 97. The error is also calculated and shown on the plot. The error is less than 1% for all the samples, as seen in the figure.

7.6 Summary

In conclusion, we have demonstrated a polarimetry-based mirror soiling level detection method in regular daylight conditions. It is field deployable and can be applied to monitor the soiling of the heliostat mirrors in the CSP plant to provide information to make maintenance schedules and also soiling pattern study of that region.

CHAPTER 8

DETECTING DEFECTS IN CSP SOLAR POWER PLANT RECEIVER TUBES UTILIZING LIGHT SCATTERING BY SMALL PARTICLES AND POLARIZATION IMAGING

8.1 Introduction

Receiver tubes[93–96] are used in concentrated solar power plant[97,98] to collect the heat generated by the sunlight. The sunlight is focused on the receiver tubes by heliostats or mirrors[98]. There are different types of receiver tubes, and the design can be different[95] but it is usually a hollow rod where the energy transfer material can flow. The receiver tubes usually contain molten salts that can store the heat or energy for a long time and can be used later when there is no sun shining on the sky (cloudy weather) or at night. The receiver tubes are usually painted with some heat-resistant paints such as carbon black or pyromark[99,100]. They can withstand thousands of degrees of Celsius temperatures. However, with time the paint on the receiver tubes can degrade, fall off or it can be scratched due to storm or wind, also it can have leakage of molten salt. All these defects in receiver tubes can cause a significant decrease of the efficiency of the power plant and cost a lot of money for repairs and maintenance. So, it would be beneficial if we could detect these defects earlier and take precautions.

People usually use thermal camera[101,102], different types of sensors and signal processing[103,104], visual inspection[104], and intensity imaging[104] to detect these defects. But they are slow and depend on good lighting conditions, costly equipment, etc. We are

proposing to improve the defect detection and inspection technique of receiver tubes by utilizing light scattering by the nanoparticles on the receiver tubes and the polarization information of the scattered light. The nanoparticles on the receiver tubes scatter light, and comparing the scattered light polarization from a perfect receiver tube with the receiver tube under inspection, we can determine the defects in the receiver tube.

8.2 Construction and Polarimetric Imaging of Receiver Tubes

To demonstrate the concept, we took some images of the receiver tubes outside in the skylight. The receiver tubes used in this measurement are homemade or, rather labmade(!). To make the receiver tubes, we used one foot (12 inches) long Aluminum hollow rods[160], epoxy putty[161] (used as glue) to hold the hollow Aluminum rods in place, and black Pyromark paint[162]. The tubes are shown in Figure 98.

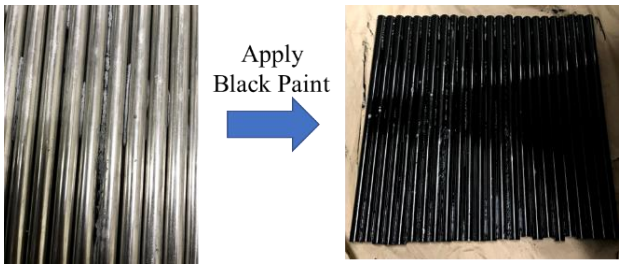


Figure 98: Aluminum Rods Hold Together with Epoxy Putty and Apply Black Paint on Top.

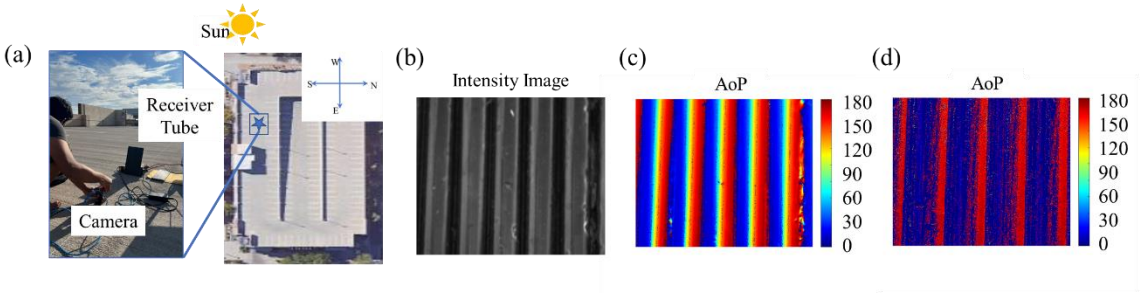


Figure 99: (a) Measurement Setup for Polarimetric Imaging of Homemade Receiver Tubes (b) Intensity Image Taken by the Polarization Camera (c) AoP Image of the Receiver Tube

When Sun is on the Back of the Tubes (d) AoP Image of the Receiver Tube When Sun is Shining on the Tube from Right Side.

Figure 99 shows the setup used in polarimetric imaging and corresponding images of the receiver tube and the processed AoP image. The measurement was done on top of the Church parking lot in ASU (Latitude=33.419258, Longitude=-111.929590). The receiver tubes were positioned in an upright position where it is standing on the ground with some support from the back. The camera is positioned horizontally and looks directly toward the receiver tubes. Figure 99(b) shows the intensity image, which is a typical image taken by any camera. The first measurement was done when the sun was behind the receiver tubes. In this condition, no unpolarized sunlight falls on the receiver tubes, and only the skylight falls on the tubes. So, the skylight works as a partially polarized light source that is scattered off of the nanoparticles (mostly from the paints) of receiver tubes. The camera used (Lucid Vision) in this measurement can capture the intensity of light polarized at 0˚, 45˚, 90˚, and 135˚ degrees with the help of the nano-gratings on top of the CMOS image sensor[152]. So, from the captured image, we can extract the S0, S1, and S2 stokes parameters which are used to calculate the Angle of polarization (AoP), shown in Figure 99(c). The AoP image shows a very clear pattern of the receiver tube, small AoP on the left side and larger AoP on right side. This AoP pattern presents a quite interesting avenue for us to use it for inspection because if any receiver tube deviate from this usual pattern, then we can detect the defect. A second measurement was done when sun was directly shining on the tubes from the right side. The AoP image for this condition is shown in Figure 99(d). Here, we don’t see the pattern that we have seen in Figure 99(c) and also it is very noisy which means the sunlight scattering makes the image more noisy and the pattern is lost.

8.3 Mathematical Model of Observed Polarization Pattern in Receiver Tubes

To get an understanding of the polarization pattern that we see for the receiver tube, we tried to come up with a mathematical model. The model is developed to understand how the pattern is formed, how it changes, and how we can utilize it.

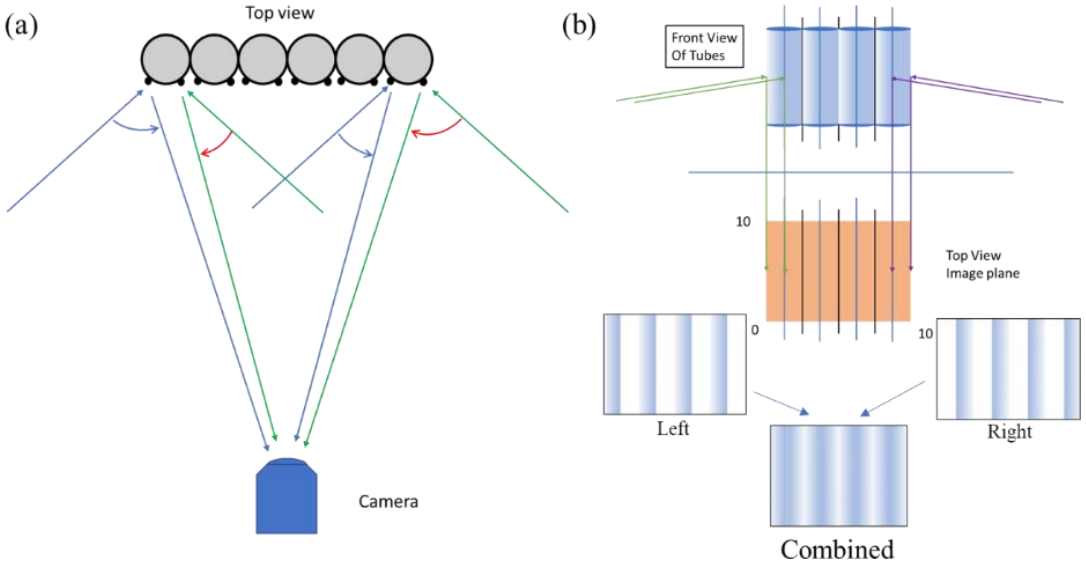


Figure 100: (a) Top View of Receiver Tubes and Scattered Light Collection by Camera (b) Close-up View of the Image Formation on the Image Plane.

In this model, there are several considerations. First, we consider that the receiver tubes are covered with very small nanoscale (~20- 50nm), particles which are mostly paint particles. These nanoparticles scatter off the light that falls on the tubes. No reflection is considered because the black paint usually absorbs light on the receiver tubes. We also consider that the patterns are formed when the light is partially polarized or under skylight because we have observed, in Figure 99, that the pattern disappears when sun shines on the tubes directly which is unpolarized sunlight. Another consideration is that the light coming from the left side of the tubes are scattered only by the left side and light coming from the right

side of the tubes are scattered only by the right side. All the scattering is modeled by the Mie scattering theory.

The scattering process and image formation is illustrated in Figure 100. Figure 100(a) shows the top view of the receiver tubes and nanoparticles attached to the tubes scattering light towards the camera. The nanoparticles are around 20-50nm radius and the refractive index is considered to be 1.75+0.65i, same as the carbon black particles[99,163]. Figure 100(b) shows the process of image formation, the light rays coming from the left side are scattered by the left side of the receiver tubes and the scattered light is collected by the camera and it produces the left side of the receiver tube in image plane. The same thing happens for the right side of the tubes also, as illustrated in Figure 100(b). The combination of these two sides are used to make the full image in the model. The calculation process of this model is described in the flowchart presented in Figure 101. The time, location are used to calculate the sun position, and from the sun position we calculate the skylight polarization pattern using Rayleigh scattering theory. Then utilizing the position of the receiver tube (in terms of zenith and azimuth angle range covered by the structure) we determine which part of the skylight is going be used as source for which side of the receiver tubes. After we have the sources for each side, we utilize the Mie scattering theory to calculate the change in light polarization after scattering for each source. We continue the calculations for each side and for each source and store the resulting unnormalized stokes parameters. Then we add all the unnormalized stokes parameters. We can do that because the Stokes parameters are just intensities. From the resulting stokes parameters for the sides of the receiver tubes, we calculate the angle of polarization (AoP), degree of linear polarization (DoLP), and degree of circular polarization (DoCP).

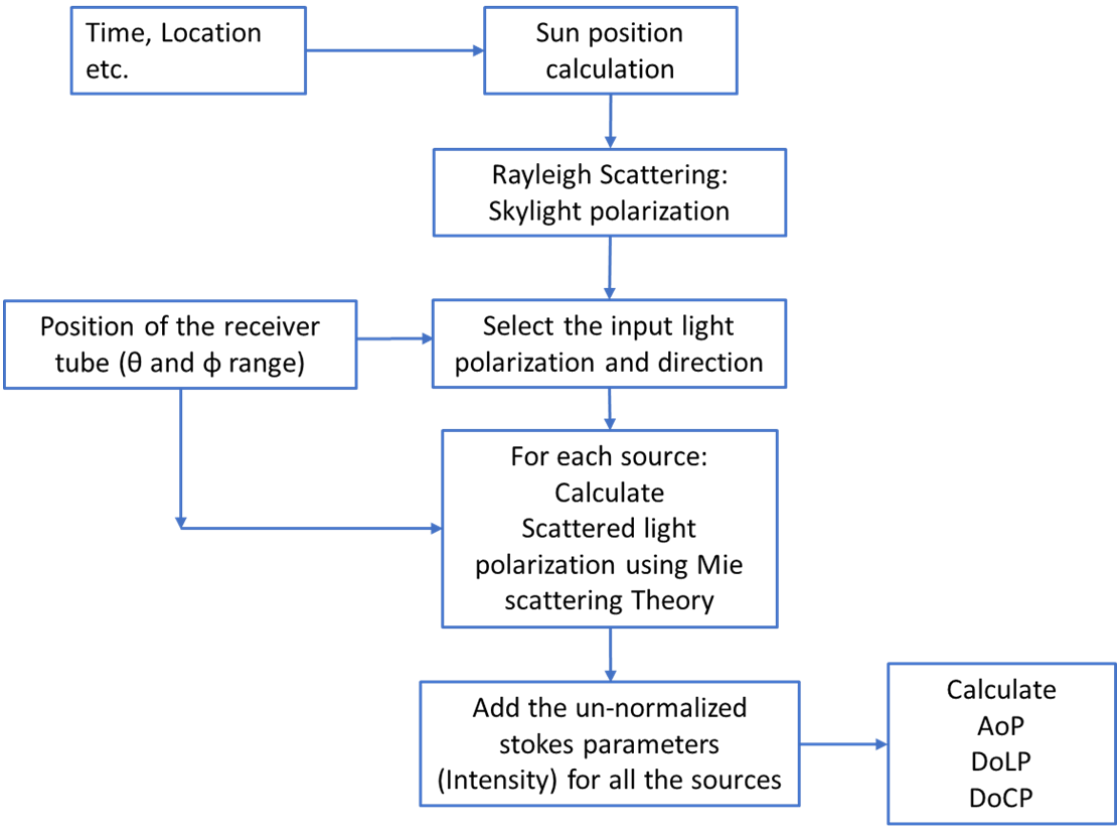


Figure 101: Flow Chart of the Calculations Done in the Receiver Tube Polarization Model

Figure 102(a) shows the model's coordinate system used in calculation. Figure 102(c-e) shows the simulated AoP, DoLP and DoCP of five receiver tubes. The sun position for this calculation was zenith =62.8711˚, azimuth=217.4035˚, which is the same sun position during the measurement. Figure 102(f) shows the receiver tube used in the measurement. Figure 102(g) shows the measured AoP under the same conditions and Figure 102(h) shows the measured DoLP. During this measurement, the sun was blocked by the clouds so only the skylight fell on the receiver tubes, just as we considered in the model. This measurement was done in Sandia, New Mexico in the experimental CSP plant Sandia National Lab. The patterns of the simulated and measured AoP and DoLP both match with

each other. So, we have been able to model the scattering process and image formation process quite correctly in this model.

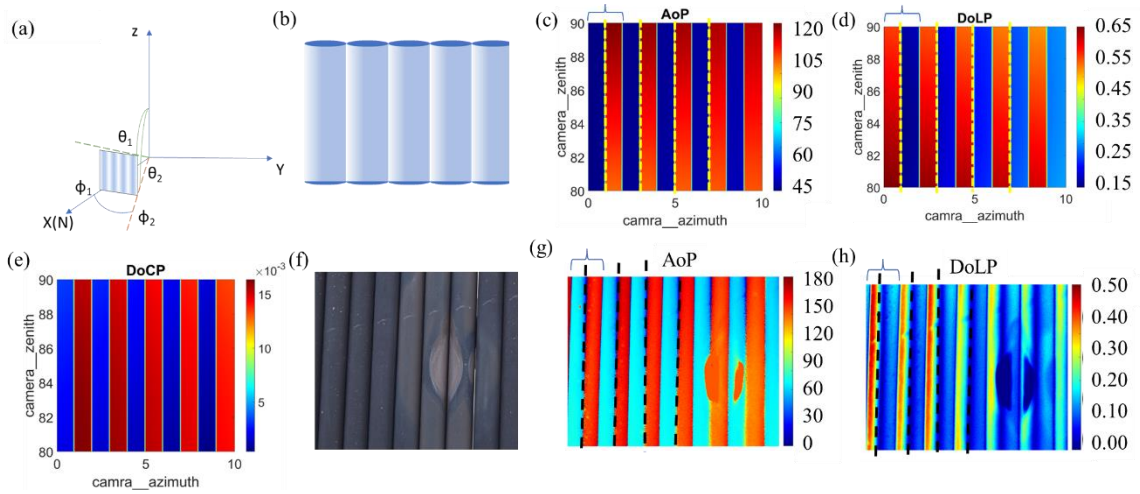


Figure 102: (a) The Coordinate System Used in Calculation (b) Visualized Receiver Tubes Drawings (c) Simulated AoP (d) Simulated DoLP (e) Simulated DoCP (f) the Receiver Tube Used in the Measurement (g) Measured AoP (h) Measured DoLP. Sun Position for This Calculation was Zenith =62.8711˚, Azimuth=217.4035˚.

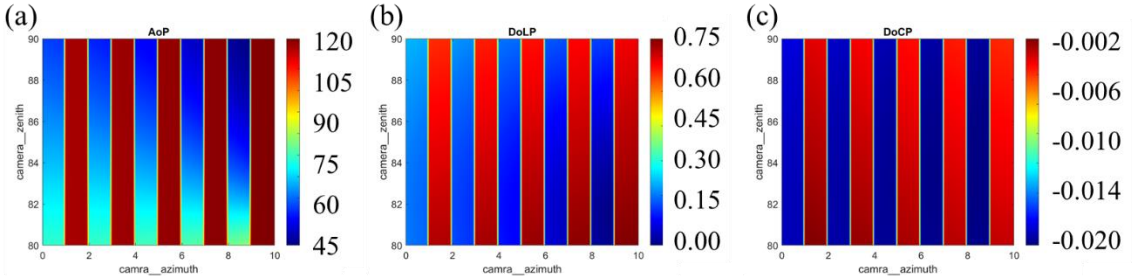


Figure 103: Simulated (a) AoP (b) DoLP (c) DoCP for Sun Position Zenith=65.7412˚, Azimuth=137.8030˚.

If the sun position is changed then the AoP pattern stays the same which is smaller AoP on the left and larger AoP on the right side of receiver tube as seen in Figure 102(c) and Figure

103(a). But if we compare Figure 102(d) and Figure 103 (b) we can see that the DoLP pattern is changed or flipped. The only difference between these two simulations is the sun's position. Also, the DoCP pattern is the same, but the value of DoCP is so small that we wouldn't be able to measure it utilizing the available polarization camera due to low extinction ratio and efficiency. So, the AoP pattern does not change with sun position, but the DoLP pattern might change with the sun position.

The AoP and DoLP images show defects in receiver tubes, as seen in Figure 102(f-h). The receiver tube used in the measurement had paint degradation, Figure 102(f). And we can see that the usual pattern of AoP and DoLP is changed due to this paint degradation, Figure 102(g, h).

8.4 Different Types of Defect Detection Utilizing Polarimetric Imaging

There are other types of defects that can occur in receiver tubes, such as the molten salt contained in them can leak out. To investigate if we can detect the leaking, we tried to emulate the situation by dropping some water droplets on the tubes. In Figure 104, we compare the images of the receiver tube without any water droplets and with water droplets. The intensity image doesn't show good contrast, as we can see from Figure 104(b). But from the AoP image in Figure 104(d), we can see that the droplets show up as protrusions in the usual pattern of AoP (Figure 104(c)). So, with the right image processing, we can detect the leaking.

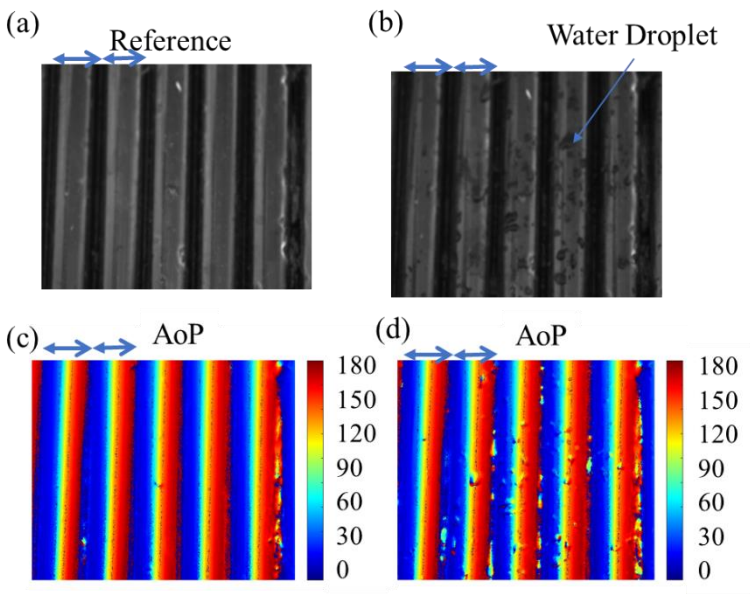


Figure 104: (a) Intensity Image of the Reference Receiver Tubes (b) Receiver Tube with Water Droplets on Them. (c) AoP Image of the Receiver Tube Without Any Water Droplets (d) AoP Image of the Receiver Tube with the Water Droplets.

The water droplets look like protrusions because the receiver tube is actually hydrophobic, so the water droplet does not spread out but stays in place like a water drop.

8.5 Summary

In summary, in this chapter, we have demonstrated the capability of defect detection with polarization imaging. The AoP pattern that we found in measurement can be reproduced with the mathematical model that we developed here. The pattern appears due to the scattering of partially polarized skylights, not sunlight. The AoP pattern stays the same for changing sun position, but the DoLP pattern can change with changing sun position. So, we can use the AoP pattern and DoLP pattern to detect different types of defects in receiver tubes by comparing the pattern with a reference AoP or DoLP pattern of the tubes without any defects.

CHAPTER 9

CONCLUSION AND OUTLOOK

I have completed different projects falling into three different categories. Ultrafast optical modulation techniques for laser and communication technologies and optical signal processing. The graphene plasmonic hybrid metasurface design in NIR and MIR serves as an efficient way to develop ultra-compact, subwavelength, ultrafast saturable absorber and optical modulator that require ultralow pump fluence. In this dissertation, it is demonstrated through simulation, numerical analysis, experiment, and data analysis, that it is possible to achieve 2-3 times smaller saturation fluence and ultrafast recovery time for saturable absorber in NIR, 1-2 orders of magnitude lower pump fluence to obtain MIR all-optical modulation with ultrafast response time and lastly, presented some initial results on saturable absorption in MIR with graphene plasmonic hybrid metasurface design. We need to investigate more about the time constant of these devices, how it is affected by the plasmonic metasurface and does plasmonic metasurface introduce new thermalization channels so that the excited carrier in graphene relaxes faster compared to other substrates. Also, developing a MIR saturable absorber and demonstrating the working principle and characteristics through simulation and experiments is a future goal that can change the MIR ultrafast laser industry forever.

Properties of light can change after scattering by a small particle. The changes of light polarization due to scattering are a fascinating topic and have many different applications. One of the applications is in the ocean, which is underwater navigation. It is demonstrated

that the developed model and data analysis procedures here can lead the way for developing a novel underwater navigation technique. In clear water conditions, a 23Km location prediction error is demonstrated. A location prediction error of ~100Km is demonstrated for dirty or turbid water conditions. The learning and knowledge gathered from this project can help lead other projects like underwater bubble detection, aerosol detection, object detection etc.

In the energy sector, concentrated solar power (CSP) plants are one of the most effective alternatives which can help us switch to clean energy. Heliostat cleanliness and receiver tubes are in the heart of the efficiency and operation of a CSP plant. Light scattering and the polarization pattern emerging from light scattering is an effective way to detect heliostat mirror soiling and receiver tube defect detection. It is demonstrated in this thesis that mirror soiling level detection is possible with the help of DoLP within 1% error. Further investigations include setting up a drone with a polarization-sensitive camera to take images of the CSP plant and determine the soiling levels of the field and conditions of receiver tubes from the processed DoLP images.

REFERENCES

1 Bonaccorso F, Sun Z, Hasan T, Ferrari AC. Graphene photonics and optoelectronics. *Nature Photon* 2010; **4**: 611–622.

2 Grigorenko AN, Polini M, Novoselov KS. Graphene plasmonics. *Nature Photon* 2012; **6**: 749–758.

3 Bao Q, Loh KP. Graphene Photonics, Plasmonics, and Broadband Optoelectronic Devices. *ACS Nano* 2012; **6**: 3677–3694.

4 Graphene - What Is It? Graphenea. https://www.graphenea.com/pages/graphene (accessed 28 Jul2022).

5 Castro Neto AH, Guinea F, Peres NMR, Novoselov KS, Geim AK. The electronic properties of graphene. *Rev Mod Phys* 2009; **81**: 109–162.

6 Novoselov KS, Geim AK, Morozov SV, Jiang D, Katsnelson MI, Grigorieva IV *et al.* Two-dimensional gas of massless Dirac fermions in graphene. *Nature* 2005; **438**: 197–200.

7 Gusynin VP, Sharapov SG, Carbotte JP. Magneto-optical conductivity in graphene. *J Phys: Condens Matter* 2006; **19**: 026222.

8 Falkovsky LA, Varlamov AA. Space-time dispersion of graphene conductivity. *The European Physical Journal B* 2007; **56**: 281–284.

9 Lim G-K, Chen Z-L, Clark J, Goh RGS, Ng W-H, Tan H-W *et al.* Giant broadband nonlinear optical absorption response in dispersed graphene single sheets. *Nature Photon* 2011; **5**: 554–560.

10 Emani NK, Chung T-F, Ni X, Kildishev AV, Chen YP, Boltasseva A. Electrically Tunable Damping of Plasmonic Resonances with Graphene. *Nano Lett* 2012; : 5.

11 Brida D. Ultrafast collinear scattering and carrier multiplication in graphene. *NATURE COMMUNICATIONS* 2013; : 9.

12 Breusing M, Ropers C, Elsaesser T. Ultrafast Carrier Dynamics in Graphite. *Phys Rev Lett* 2009; **102**: 086809.

13 Plo T. Experimental Verification of Carrier Multiplication in Graphene. *Nano Lett* 2014; : 5.

14 Tielrooij KJ. Photoexcitation cascade and multiple hot-carrier generation in graphene. *NATURE PHYSICS* 2013; **9**: 5.

15 Kim J, Son H, Cho DJ, Geng B, Regan W, Shi S *et al.* Electrical Control of Optical Plasmon Resonance with Graphene. *Nano Lett* 2012; : 5.

16 Fang Z, Thongrattanasiri S, Schlather A, Liu Z, Ma L, Wang Y *et al.* Gated Tunability and Hybridization of Localized Plasmons in Nanostructured Graphene. *ACS Nano* 2013; **7**: 2388–2395.

17 Chen F, Xia J, Ferry DK, Tao N. Dielectric screening enhanced performance in graphene FET. *Nano letters* 2009; **9**: 2571–2574.

18 Fregonese S, Magallo M, Maneux C, Happy H, Zimmer T. Scalable electrical compact modeling for graphene FET transistors. *IEEE Transactions on Nanotechnology* 2013; **12**: 539–546.

19 Generalov AA, Andersson MA, Yang X, Vorobiev A, Stake J. A 400-GHz graphene FET detector. *IEEE Transactions on Terahertz Science and Technology* 2017; **7**: 614–616.

20 Hamed A, Asad M, Wei M-D, Vorobiev A, Stake J, Negra R. Integrated 10-GHz Graphene FET Amplifier. *IEEE Journal of Microwaves* 2021; **1**: 821–826.

21 Wang H, Yang N, Chang L, Zhou C, Li S, Deng M *et al.* CMOS-compatible all-optical modulator based on the saturable absorption of graphene. *Photon Res* 2020; **8**: 468.

22 Basiri A, Rafique MZE, Bai J, Zuo J, Choi S, Yao Y. Ultrafast Mid-Infrared Optical Modulator Based on Optically Controlled Graphene-Integrated Metasurface. In: *CLEO: QELS_Fundamental Science*. Optical Society of America, 2020, pp FTh3Q-2.

23 Zhong C, Li J, Lin H. Graphene-based all-optical modulators. *Front Optoelectron* 2020; **13**: 114–128.

24 Ono M, Hata M, Tsunekawa M, Nozaki K, Sumikura H, Chiba H *et al.* Ultrafast and energy-efficient all-optical switching with graphene-loaded deep-subwavelength plasmonic waveguides. *Nat Photonics* 2020; **14**: 37–43.

25 Hwang MT, Park I, Heiranian M, Taqieddin A, You S, Faramarzi V *et al.* Ultrasensitive detection of dopamine, IL-6 and SARS-CoV-2 proteins on crumpled graphene FET biosensor. *Advanced materials technologies* 2021; **6**: 2100712.

26 Krsihna BV, Ahmadsaidulu S, Teja SST, Jayanthi D, Navaneetha A, Reddy PR *et al.* Design and development of graphene FET biosensor for the detection of SARS-CoV-2. *Silicon* 2021; : 1–9.

27 Xia Y, Sun Y, Li H, Chen S, Zhu T, Wang G *et al.* Plasma treated graphene FET sensor for the DNA hybridization detection. *Talanta* 2021; **223**: 121766.

28 Li J, Wu D, Yu Y, Li T, Li K, Xiao M-M *et al.* Rapid and unamplified identification of COVID-19 with morpholino-modified graphene field-effect transistor nanosensor. *Biosensors and Bioelectronics* 2021; **183**: 113206.

29 Koppens FHL. Photodetectors based on graphene, other two-dimensional materials and hybrid systems. *NATURE NANOTECHNOLOGY* 2014; **9**: 14.

30 Mueller T, Xia F, Avouris P. Graphene photodetectors for high-speed optical communications. ; : 20.

31 Ma Z, Kikunaga K, Wang H, Sun S, Amin R, Maiti R *et al.* Compact Graphene Plasmonic Slot Photodetector on Silicon-on- Insulator with High Responsivity. *ACS Photonics* 2020; : 9.

32 Wei J. Zero-bias mid-infrared graphene photodetectors with bulk photoresponse and calibration-free polarization detection. ; : 9.

33 Yao Y, Shankar R, Rauter P, Song Y, Kong J, Loncar M *et al.* Mid-infrared graphene detectors with antenna enhanced light absorption and photo-carrier collection. 2014; : 2.

34 Xie Y, Meng X, Chang Y, Mao D, Qin Z, Wan L *et al.* Heteroatom Modification Enhances Corrosion Durability in High-Mechanical-Performance Graphene-Reinforced Aluminum Matrix Composites. *Advanced Science* 2022; : 2104464.

35 Indukuri CSR, Nerella R. Enhanced transport properties of graphene oxide based cement composite material. *Journal of Building Engineering* 2021; **37**: 102174.

36 Graphene applications: what is graphene used for? | Graphene-Info. https://www.graphene-info.com/graphene-applications (accessed 28 Jul2022).

37 Yao Y, Shankar R, Kats MA, Song Y, Kong J, Loncar M *et al.* Electrically tunable metasurface perfect absorbers for ultrathin mid-infrared optical modulators. *Nano letters* 2014; **14**: 6526–6532.

38 Basiri A, Rafique MZE, Bai J, Choi S, Yao Y. Ultrafast low-pump fluence all-optical modulation based on graphene-metal hybrid metasurfaces. *Light Sci Appl* 2022; **11**: 102.

39 Verdeyen JT. *Laser electronics*. 3rd ed. 1995.

40 Bao Q, Zhang H, Ni Z, Wang Y, Polavarapu L, Shen Z *et al.* Monolayer graphene as a saturable absorber in a mode-locked laser. *Nano Res* 2011; **4**: 297–307.

41 Bao Q, Zhang H, Wang Y, Ni Z, Yan Y, Shen ZX *et al.* Atomic-Layer Graphene as a Saturable Absorber for Ultrafast Pulsed Lasers. *Advanced Functional Materials* 2009; **19**: 3077–3083.

42 Xing G, Guo H, Zhang X, Sum TC, Huan CHA. The Physics of ultrafast saturable absorption in graphene. *Opt Express* 2010; **18**: 4564.

43 Zhou G-J, Wong W-Y, Cui D, Ye C. Large Optical-Limiting Response in Some Solution-Processable Polyplatinaynes. *Chem Mater* 2005; **17**: 5209–5217.

44 O'Flaherty S m., Hold S v., Cook M j., Torres T, Chen Y, Hanack M *et al.* Molecular Engineering of Peripherally And Axially Modified Phthalocyanines for Optical Limiting and Nonlinear Optics. *Advanced Materials* 2003; **15**: 19–32.

45 Jia W, Douglas EP, Guo F, Sun W. Optical limiting of semiconductor nanoparticles for nanosecond laser pulses. *Appl Phys Lett* 2004; **85**: 6326–6328.

46 Feng M, Zhan H, Chen Y. Nonlinear optical and optical limiting properties of graphene families. *Appl Phys Lett* 2010; **96**: 033107.

47 Sun Z, Martinez A, Wang F. Optical modulators with 2D layered materials. *Nature Photonics* 2016; **10**: 227–238.

48 Brunner D, Soriano MC, Mirasso CR, Fischer I. Parallel photonic information processing at gigabyte per second data rates using transient states. *Nature communications* 2013; **4**: 1–7.

49 Chen J-H, Zheng B-C, Shao G-H, Ge S-J, Xu F, Lu Y-Q. An all-optical modulator based on a stereo graphene–microfiber structure. *Light: Science & Applications* 2015; **4**: e360–e360.

50 Guo Q, Yao Y, Luo Z-C, Qin Z, Xie G, Liu M *et al.* Universal near-infrared and mid-infrared optical modulation for ultrafast pulse generation enabled by colloidal plasmonic semiconductor nanocrystals. *ACS nano* 2016; **10**: 9463–9469.

51 Kuramochi H, Takeuchi S, Yonezawa K, Kamikubo H, Kataoka M, Tahara T. Probing the early stages of photoreception in photoactive yellow protein with ultrafast time-domain Raman spectroscopy. *Nature Chemistry* 2017; **9**: 660–666.

52 Ding L, Morits D, Bakker R, Li S, Eschimese D, Zhu S *et al.* All-Optical Modulation in Chains of Silicon Nanoantennas. *ACS Photonics* 2020; **7**: 1001–1008.

53 Afinogenov BI, Bessonov VO, Soboleva IV, Fedyanin AA. Ultrafast all-optical light control with Tamm plasmons in photonic nanostructures. Supporting Information. ; : 6.

54 Shen L, Healy N, Mitchell CJ, Penades JS, Nedeljkovic M, Mashanovich GZ *et al.* Two-photon absorption and all-optical modulation in germanium-on-silicon waveguides for the mid-infrared. *Optics Letters* 2015; **40**: 2213–2216.

55 Wu R, Collins J, Chekulaev D, Kaplan A. All-optical modulation and ultrafast switching in MWIR with sub-wavelength structured silicon. *Applied Sciences* 2019; **9**: 1808.

56 Yu J-P, Chen S, Fan F, Xu S-T, Cheng J-R, Chen X-F *et al.* Accelerating terahertz all-optical modulation by hot carriers effects of silver nanorods in PVA film. *AIP Advances* 2019; **9**: 075017.

57 Reed GT, Mashanovich G, Gardes FY, Thomson Dj. Silicon optical modulators. *Nature photonics* 2010; **4**: 518–526.

58 Schönenberger S, Stöferle T, Moll N, Mahrt RF, Dahlem MS, Wahlbrink T *et al.* Ultrafast all-optical modulator with femtojoule absorbed switching energy in silicon-on-insulator. *Optics express* 2010; **18**: 22485–22496.

59 Manolatou C, Lipson M. All-optical silicon modulators based on carrier injection by two-photon absorption. *Journal of lightwave technology* 2006; **24**: 1433.

60 Almeida VR, Barrios CA, Panepucci RR, Lipson M. All-optical control of light on a silicon chip. *Nature* 2004; **431**: 1081–1084.

61 Liu A, Liao L, Rubin D, Nguyen H, Ciftcioglu B, Chetrit Y *et al.* High-speed optical modulation based on carrier depletion in a silicon waveguide. *Optics express* 2007; **15**: 660–668.

62 Xu Q, Manipatruni S, Schmidt B, Shakya J, Lipson M. 12.5 Gbit/s carrier-injection-based silicon micro-ring silicon modulators. *Optics express* 2007; **15**: 430–436.

63 Feng D, Liao S, Liang H, Fong J, Bijlani B, Shafiiha R *et al.* High speed GeSi electro-absorption modulator at 1550 nm wavelength on SOI waveguide. *Optics express* 2012; **20**: 22224–22232.

64 Baba T, Nguyen HC, Yazawa N, Terada Y, Hashimoto S, Watanabe T. Slow-light Mach–Zehnder modulators based on Si photonic crystals. *Science and technology of advanced materials* 2014.

65 Li W, Chen B, Meng C, Fang W, Xiao Y, Li X *et al.* Ultrafast all-optical graphene modulator. *Nano letters* 2014; **14**: 955–959.

66 Sun F, Xia L, Nie C, Shen J, Zou Y, Cheng G *et al.* The all-optical modulator in dielectric-loaded waveguide with graphene-silicon heterojunction structure. *Nanotechnology* 2018; **29**: 135201.

67 Zhou F, Du W. Ultrafast all-optical plasmonic graphene modulator. *Applied Optics* 2018; **57**: 6645–6650.

68 Afinogenov BI, Bessonov VO, Soboleva IV, Fedyanin AA. Ultrafast All-Optical Light Control with Tamm Plasmons in Photonic Nanostructures. *ACS Photonics* 2019; **6**: 844–850.

69 Rafique MZE, Rafique MZE, Basiri A, Basiri A, Bai J, Bai J *et al.* Graphene-Plasmonic Hybrid Metasurface Saturable Absorber. In: *OSA Advanced Photonics Congress 2021 (2021), paper NoM5B.3*. Optica Publishing Group, 2021, p NoM5B.3.

70 Dash A, Palanchoke U, Gely M, Jourdan G, Hentz S, Selvaraja SK *et al.* Enhanced all-optical cavity-tuning using graphene. *Opt Express* 2019; **27**: 34093.

71 Bohren CF, Huffman DR. *Absorption and Scattering of Light by Small Particles*. John Wiley & Sons, 2008.

72 Demos SG, Alfano RR. Optical polarization imaging. *Applied optics* 1997; **36**: 150–155.

73 Polarization-division multiplexing. Wikipedia. 2022.https://en.wikipedia.org/w/index.php?title=Polarization-division_multiplexing&oldid=1094023428 (accessed 28 Jul2022).

74 Herard C, Lacourt A. New multiplexing technique using polarization of light. *Applied optics* 1991; **30**: 222–231.

75 Namer E, Schechner YY. Advanced visibility improvement based on polarization filtered images. In: *Polarization Science and Remote Sensing II*. SPIE, 2005, pp 36–45.

76 Specht HP, Nölleke C, Reiserer A, Uphoff M, Figueroa E, Ritter S *et al.* A single-atom quantum memory. *Nature* 2011; **473**: 190–193.

77 Hamaoui M. Polarized skylight navigation. *Appl Opt* 2017; **56**: B37.

78 Smith GS. The polarization of skylight: An example from nature. *American Journal of Physics* 2007; **75**: 25–35.

79 Horváth G, Barta A, Pomozi I, Suhai B, Hegedüs R, Åkesson S *et al.* On the trail of Vikings with polarized skylight: experimental study of the atmospheric optical prerequisites allowing polarimetric navigation by Viking seafarers. *Philosophical Transactions of the Royal Society B: Biological Sciences* 2011; **366**: 772–782.

80 Hegedüs R, Åkesson S, Wehner R, Horváth G. Could Vikings have navigated under foggy and cloudy conditions by skylight polarization? On the atmospheric optical prerequisites of polarimetric Viking navigation under foggy and cloudy skies. *Proc R Soc A* 2007; **463**: 1081–1095.

81 Marchant J. Did Vikings navigate by polarized light? *Nature* 2011; : news.2011.58.

82 Zhang X, Wan Y, Li L. Bio-inspired polarized skylight navigation: a review. In: Liu J, Sun H (eds). . Enshi, China, 2015, p 981517.

83 Labhart T, Meyer EP. Detectors for polarized skylight in insects: a survey of ommatidial specializations in the dorsal rim area of the compound eye. *Microscopy research and technique* 1999; **47**: 368–379.

84 Kaufman YJ, Gitelson A, Karnieli A, Ganor E, Fraser RS, Nakajima T *et al.* Size distribution and scattering phase function of aerosol particles retrieved from sky brightness measurements. *Journal of Geophysical Research: Atmospheres* 1994; **99**: 10341–10356.

85 Rayleigh sky model - Wikipedia. https://en.wikipedia.org/wiki/Rayleigh_sky_model (accessed 28 Jul2022).

86 Mie scattering. Wikipedia. 2022.https://en.wikipedia.org/w/index.php?title=Mie_scattering&oldid=1100562394 (accessed 28 Jul2022).

87 Pomozi I, Horváth G, Wehner R. How the clear-sky angle of polarization pattern continues underneath clouds: full-sky measurements and implications for animal orientation. *Journal of Experimental Biology* 2001; **204**: 2933–2942.

88 Suhai B, Horváth G. How well does the Rayleigh model describe the E-vector distribution of skylight in clear and cloudy conditions? A full-sky polarimetric study. *J Opt Soc Am A* 2004; **21**: 1669.

89 Picotti G, Binotti M, Cholette ME, Borghesani P, Manzolini G, Steinberg T. Modelling the soiling of heliostats: Assessment of the optical efficiency and impact of cleaning operations. Casablanca, Morocco, 2019, p 030043.

90 Wolfertstetter F, Fonk R, Prahl C, Röger M, Wilbert S, Fernández-Reche J. Airborne soiling measurements of entire solar fields with Qfly. Daegu, South Korea, 2020, p 100008.

91 Meyen S, Lupfert E, Center DGA, Fernandez-Garcia A. Standardization of Solar Mirror Reflectance Measurements - Round Robin Test: Preprint. ; : 11.

92 Wolfertstetter F, Pottler K, Merrouni AA, Mezrhab A, Pitz-Paal R. A Novel Method for Automatic Real-Time Monitoring of Mirror Soiling Rates. ; : 9.

93 Ho CK. Advances in central receivers for concentrating solar applications. *Solar Energy* 2017; **152**: 38–56.

94 Kulkarni K, Havaldar S, Bhattacharya N. Review on advance tubular receivers for central solar tower system. *IOP Conf Ser: Mater Sci Eng* 2021; **1136**: 012018.

95 Luceño JA, de la Fuente E, Martín M. Optimal Design of Solar Receivers in CSP Plants: Effects of Facility Location. *Ind Eng Chem Res* 2021; **60**: 7218–7231.

96 Vittitow MP. Concentrating Solar Power - Receivers. ; : 28.

97 Concentrated solar power. Wikipedia. 2022.https://en.wikipedia.org/w/index.php?title=Concentrated_solar_power&oldid=1100535781 (accessed 4 Aug2022).

98 Power Tower System Concentrating Solar-Thermal Power Basics. Energy.gov. https://www.energy.gov/eere/solar/power-tower-system-concentrating-solar-thermal-power-basics (accessed 4 Aug2022).

99 Liu F, Yon J, Fuentes A, Lobo P, Smallwood GJ, Corbin JC. Review of recent literature on the light absorption properties of black carbon: Refractive index, mass absorption cross section, and absorption function. *Aerosol Science and Technology* 2020; **54**: 33–51.

100 Rubin EB, Chen Y, Chen R. Optical properties and thermal stability of Cu spinel oxide nanoparticle solar absorber coatings. *Solar Energy Materials and Solar Cells* 2019; **195**: 81–88.

101 Jalón A, Pérez D, Benito-Cia P, Zaversky F. Inspection Receiver Tubes Device for CSP Plants. *Energy Procedia* 2015; **69**: 1868–1876.

102 Price H, Forristall R, Wendelin T, Lewandowski A, Moss TA, Gummo C. Field Survey of Parabolic Trough Receiver Thermal Performance. *Proceedings of ISEC2006, ASME International Solar Energy Conference* 2006; **2006**. doi:10.1115/ISEC2006-99167.

103 Gómez Muñoz CQ, Arcos Jimenez A, García Marquez FP, Kogia M, Cheng L, Mohimi A *et al.* Cracks and welds detection approach in solar receiver tubes employing electromagnetic acoustic transducers. *Structural Health Monitoring* 2018; **17**: 1046–1055.

104 Papaelias M, Cheng L, Kogia M, Mohimi A, Kappatos V, Selcuk C *et al.* Inspection and Structural Health Monitoring techniques for Concentrated Solar Power plants. *Renewable Energy* 2016; **85**: 1178–1191.

105 Zhang H, Lu SB, Zheng J, Du J, Wen SC, Tang DY *et al.* Molybdenum disulfide (MoS_2) as a broadband saturable absorber for ultra-fast photonics. *Opt Express* 2014; **22**: 7249.

106 Yuan J, Mu H, Li L, Chen Y, Yu W, Zhang K *et al.* Few-layer platinum diselenide as a new saturable absorber for ultrafast fiber lasers. *ACS applied materials & interfaces* 2018; **10**: 21534–21540.

107 Yan P, Liu A, Chen Y, Chen H, Ruan S, Guo C *et al.* Microfiber-based WS_2-film saturable absorber for ultra-fast photonics. *Opt Mater Express* 2015; **5**: 479.

108 He M, Quan C, He C, Huang Y, Zhu L, Yao Z *et al.* Enhanced Nonlinear Saturable Absorption of MoS_2 /Graphene Nanocomposite Films. *J Phys Chem C* 2017; **121**: 27147–27153.

109 Okazaki D, Morichika I, Arai H, Kauppinen E, Zhang Q, Anisimov A *et al.* Ultrafast saturable absorption of large-diameter single-walled carbon nanotubes for passive mode locking in the mid-infrared. *Opt Express* 2020; **28**: 19997.

110 Xiao P, Wu K, Mao D, Chen J. A pulsewidth measurement technology based on carbon-nanotube saturable absorber. *Opt Express* 2019; **27**: 4188.

111 Gladush Y, Mkrtchyan AA, Kopylova DS, Ivanenko A, Nyushkov B, Kobtsev S *et al.* Ionic Liquid Gated Carbon Nanotube Saturable Absorber for Switchable Pulse Generation. *Nano Lett* 2019; **19**: 5836–5843.

112 Zhao C, Zhang H, Qi X, Chen Y, Wang Z, Wen S *et al.* Ultra-short pulse generation by a topological insulator based saturable absorber. *Appl Phys Lett* 2012; **101**: 211106.

113 Mu H, Wang Z, Yuan J, Xiao S, Chen C, Chen Y *et al.* Graphene–Bi_2Te_3 Heterostructure as Saturable Absorber for Short Pulse Generation. *ACS Photonics* 2015; **2**: 832–841.

114 Demongodin P, El Dirani H, Lhuillier J, Crochemore R, Kemiche M, Wood T *et al.* Ultrafast saturable absorption dynamics in hybrid graphene/Si_3N_4 waveguides. *APL Photonics* 2019; **4**: 076102.

115 Zubyuk VV, Vabishchevich PP, Shcherbakov MR, Shorokhov AS, Fedotova AN, Liu S *et al.* Low-Power Absorption Saturation in Semiconductor Metasurfaces. *ACS Photonics* 2019; **6**: 2797–2806.

116 Wang J, Coillet A, Demichel O, Wang Z, Rego D, Bouhelier A *et al.* Saturable plasmonic metasurfaces for laser mode locking. *Light Sci Appl* 2020; **9**: 50.

117 Li P, Chen Y, Yang T, Wang Z, Lin H, Xu Y *et al.* Two-Dimensional $CH_3NH_3PbI_3$ Perovskite Nanosheets for Ultrafast Pulsed Fiber Lasers. *ACS Appl Mater Interfaces* 2017; **9**: 12759–12765.

118 Wang K, Szydłowska BM, Wang G, Zhang X, Wang JJ, Magan JJ *et al.* Ultrafast Nonlinear Excitation Dynamics of Black Phosphorus Nanosheets from Visible to Mid-Infrared. *ACS Nano* 2016; **10**: 6923–6932.

119 Sotor J, Sobon G, Macherzynski W, Paletko P, Abramski KM. Black phosphorus saturable absorber for ultrashort pulse generation. *Appl Phys Lett* 2015; **107**: 051108.

120 Lim G-K, Chen Z-L, Clark J, Goh R, Ng W-H, Tan H-W *et al.* Giant broadband nonlinear optical absorption response in dispersed graphene single sheets. *Nature Photonics* 2011; **5**: 554–560.

121 Spühler GJ, Weingarten KJ, Grange R, Krainer L, Haiml M, Liverini V *et al.* Semiconductor saturable absorber mirror structures with low saturation fluence. *Appl Phys B* 2005; **81**: 27–32.

122 Bao Q, Zhang H, Ni Z, Wang Y, Polavarapu L, Shen Z *et al.* Monolayer graphene as a saturable absorber in a mode-locked laser. *Nano Res* 2011; **4**: 297–307.

123 Mu H, Wang Z, Yuan J, Xiao S, Chen C, Chen Y *et al.* Graphene–Bi2Te3 heterostructure as saturable absorber for short pulse generation. *Acs Photonics* 2015; **2**: 832–841.

124 Xing G, Guo H, Zhang X, Sum TC, Huan CHA. The Physics of ultrafast saturable absorption in graphene. *Opt Express* 2010; **18**: 4564.

125 Winzer T, Knorr A, Mittendorff M, Winnerl S, Lien M-B, Sun D *et al.* Absorption saturation in optically excited graphene. *Appl Phys Lett* 2012; **101**: 221115.

126 Malic E, Winzer T, Wendler F, Brem S, Jago R, Knorr A *et al.* Carrier dynamics in graphene: ultrafast many-particle phenomena. *Annalen der Physik* 2017; **529**: 1700038.

127 Kador L. Kramers–Kronig relations in nonlinear optics. *Applied physics letters* 1995; **66**: 2938–2939.

128 Md Zubair Ebne Rafique, Ali Basiri, Jing Bai, Zuo J, Yu Yao. Ultra-fast Graphene-Plasmonic Hybrid Metasurface Saturable Absorber with Low Saturation Fluence. *Manuscript in preparation* 2022.

129 Garmire E. Resonant optical nonlinearities in semiconductors. *IEEE J Select Topics Quantum Electron* 2000; **6**: 1094–1110.

130 Falkovsky LA, Pershoguba SS. Optical far-infrared properties of a graphene monolayer and multilayer. *Physical Review B* 2007; **76**: 153410.

131 Chen C-J, Wai PKA, Menyuk CR. Self-starting of passively mode-locked lasers with fast saturable absorbers. *Opt Lett* 1995; **20**: 350.

132 Li L, Li Z, Li K, Blarel L, Wendisch M. A method to calculate Stokes parameters and angle of polarization of skylight from polarized CIMEL sun/sky radiometers. *Journal of Quantitative Spectroscopy and Radiative Transfer* 2014; **149**: 334–346.

133 Ma T, Hu X, Zhang L, Lian J, He X, Wang Y *et al.* An Evaluation of Skylight Polarization Patterns for Navigation. *Sensors* 2015; **15**: 5895–5913.

134 Tang J, Zhang N, Li D, Wang F, Zhang B, Wang C *et al.* Novel robust skylight compass method based on full-sky polarization imaging under harsh conditions. *Opt Express* 2016; **24**: 15834.

135 Powell SB, Garnett R, Marshall J, Rizk C, Gruev V. Bioinspired polarization vision enables underwater geolocalization. *Sci Adv* 2018; **4**: eaao6841.

136 Hu P, Yang J, Guo L, Yu X, Li W. Solar-tracking methodology based on refraction-polarization in Snell's window for underwater navigation. *Chinese Journal of Aeronautics* 2022; **35**: 380–389.

137 Zhao H, Xu W, Zhang Y, Li X, Zhang H, Xuan J *et al.* Polarization patterns under different sky conditions and a navigation method based on the symmetry of the AOP map of skylight. *Opt Express* 2018; **26**: 28589.

138 Wu L, Gao J, Fan Z, Zhang J. Measurements of skylight polarization: a case study in urban region with high-loading aerosol. *Appl Opt* 2015; **54**: B256.

139 Stokes parameters. Wikipedia. 2022.https://en.wikipedia.org/w/index.php?title=Stokes_parameters&oldid=1098598981 (accessed 3 Aug2022).

140 Hulst HC, van de Hulst HC. *Light scattering by small particles*. Courier Corporation, 1981.

141 On the Composition and Resolution of Streams of Polarized Light from different Sources - Mathematical and Physical Papers. https://www.cambridge.org/core/books/abs/mathematical-and-physical-papers/on-the-composition-and-resolution-of-streams-of-polarized-light-from-different-sources/A021426E1F3E9BD366E8E1F967FBE3CE (accessed 15 Aug2022).

142 Chandrasekhar S. *Radiative transfer*. Courier Corporation, 2013.

143 Wendisch M, Yang P. *Theory of Atmospheric Radiative Transfer: A Comprehensive Introduction*. John Wiley & Sons, 2012.

144 Overview :: Ocean Optics Web Book. https://www.oceanopticsbook.info/view/introduction/overview#Eq:rotations1 (accessed 3 Aug2022).

145 Power From The Sun :: Chapter 3. http://powerfromthesun.net/Book/chapter03/chapter03.html (accessed 11 Sep2022).

146 Zerubavel E. The Standardization of Time: A Sociohistorical Perspective. *American Journal of Sociology* 1982; **88**: 1–23.

147 Milne R. Note on the Equation, of Time. *Math Gaz* 1921; **10**: 372–375.

148 Voss KJ, Gleason ACR, Gordon HR, Kattawar GW, You Y. Observation of non-principal plane neutral points in the in-water upwelling polarized light field. *Opt Express* 2011; **19**: 5942.

149 Hu P, Yang J, Guo L, Yu X, Li W. Solar-tracking methodology based on refraction-polarization in Snell's window for underwater navigation. *Chinese Journal of Aeronautics* 2022; **35**: 380–389.

150 Ding Y, Pau S. Circularly and elliptically polarized light under water and the Umov effect. *Light Sci Appl* 2019; **8**: 32.

151 Zhou G, Xu W, Niu C, Zhao H. The polarization patterns of skylight reflected off wave water surface. *Opt Express* 2013; **21**: 32549.

152 Beyond Conventional Imaging: Sony's Polarized Sensor - Polarsens | LUCID Vision Labs. https://thinklucid.com/tech-briefs/polarization-explained-sony-polarized-sensor/ (accessed 4 Aug2022).

153 Ramella-Roman JC, Prahl SA, Jacques SL. Three Monte Carlo programs of polarized light transport into scattering media: part I. *Opt Express* 2005; **13**: 4420.

154 Ramella-Roman JC, Prahl SA, Jacques SL. Three Monte Carlo programs of polarized light transport into scattering media: part II. *Opt Express* 2005; **13**: 10392.

155 Leathers RA, Downes TV, Davis CO, Mobley CD. Monte Carlo Radiative Transfer Simulations for Ocean Optics: A Practical Guide: Defense Technical Information Center: Fort Belvoir, VA, 2004 doi:10.21236/ADA426624.

156 Prahl SA. A Monte Carlo model of light propagation in tissue. In: Mueller GJ, Sliney DH, Potter RF (eds). . Berlin, Germany, 1989, p 1030509.

157 Tynes HH, Kattawar GW, Zege EP, Katsev IL, Prikhach AS, Chaikovskaya LI. Monte Carlo and multicomponent approximation methods for vector radiative transfer by use of effective Mueller matrix calculations. *Appl Opt* 2001; **40**: 400.

158 Xu M. Electric field Monte Carlo simulation of polarized light propagation in turbid media. *Opt Express* 2004; **12**: 6530.

159 Matzler C. MATLAB functions for Mie scattering and absorption. *IAP Res Rep* 2002; **8**.

160 McMaster-Carr. https://www.mcmaster.com/ (accessed 4 Aug2022).

161 Amazon.com: JB Weld 8297 -DEU HighHeat Heat-Resistant Epoxy Metal Putty for All High Temperature Work : Industrial & Scientific. https://www.amazon.com/Weld-8297-DEU-HighHeat-Heat-Resistant-Temperature/dp/B010LMLS3E (accessed 4 Aug2022).

162 PK25FBLKGL Pyromark 2500 Flat Black - 1 Gallon-25010. AMMRO. https://americanmademro.com/products/labeling-and-marking/high-heat-indicating/pyromark/pk25fblkgl-pyromark-2500-flat-black-1-gallon-25010/ (accessed 4 Aug2022).

163 Zhang X, Mao M, Yin Y. Optically effective complex refractive index of coated black carbon aerosols: from numerical aspects. *Atmos Chem Phys* 2019; **19**: 7507–7518.

164 Grange R, Haiml M, Paschotta R, Spühler GJ, Krainer L, Golling M *et al.* New regime of inverse saturable absorption for self-stabilizing passively mode-locked lasers. *Appl Phys B* 2005; **80**: 151–158.

165 Thoen ER, Koontz EM, Joschko M, Langlois P, Schibli TR, Kärtner FX *et al.* Two-photon absorption in semiconductor saturable absorber mirrors. *Appl Phys Lett* 1999; **74**: 3927–3929.

APPENDIX A

SIMULATION, FABRICATION AND MEASUREMENT METHODS OF ULTRAFAST AND LOW SATURATION FLUENCE SATURABLE ABSORBER

A1: Numerical Simulation:

All numerical simulation is done using Matlab with a phenomenological graphene carrier dynamics model[42], considering the non-equilibrium carrier accumulation and subsequent relaxation of the non-equilibrium carriers upon laser excitation. To model these processes, we can consult the dynamic interplay between the occupation probability of electrons in the conduction band ($f_C(t, \omega)$) and the valence band ($f_V(t, \omega)$) upon laser excitation. These dynamics of the carrier occupation probabilities are explained sufficiently by two semi-empirical coupled differential equations relating the density of states of graphene at the excitation photon energy (D(E)), incident light intensity (I (t, ω)) and the associated relaxation process, τ_1, which are provided below[42],

$$\frac{\partial f_V(t,\omega)}{\partial t} = -f_V(t,\omega)\,\frac{\pi\alpha_f}{D(E)\,\hbar\omega}\,I_0 exp(-\frac{t^2}{\Delta^2}) + f_C(t,\omega)\,\frac{\pi\alpha_f}{D(E)\,\hbar\omega}\,I_0 exp(-\frac{t^2}{\Delta^2}) + \frac{1-f_V(t,\omega)}{\tau_1} \quad (1)$$

$$\frac{\partial f_C(t,\omega)}{\partial t} = f_V(t,\omega)\,\frac{\pi\alpha_f}{D(E)\,\hbar\omega}\,I_0 exp(-\frac{t^2}{\Delta^2}) - f_C(t,\omega)\,\frac{\pi\alpha_f}{D(E)\,\hbar\omega}\,I_0 exp(-\frac{t^2}{\Delta^2}) - \frac{f_C(t,\omega)}{\tau_1} \quad (2)$$

Where, $I(t) = I_0 exp(-2.76t^2/\Delta^2)$ is the incident laser intensity, α_f is the fine structure coefficient of graphene, $D(E) = D(\hbar\omega/2) = \hbar\omega/(\pi\hbar^2 v_F^2)$ is the density of states of graphene on the CB or VB at incident photon energy, and Δ is the incident laser pulse width (FWHM). These two coupled differential equations explain the absorption, stimulated

emission, and the subsequent relaxation of the excited non-equilibrium carriers. For saturable absorption, we are primarily interested in the ultrafast accumulation of non-equilibrium photoexcited carriers from VB to CB and the subsequent relaxation of the carriers from the non-equilibrium states to the equilibrium Fermi-Dirac distribution near the Dirac point. In this process, the most contributing relaxation channels are the carrier-carrier and carrier-optical phonon scattering, which happens within 10-100fs. The scattering contributions from other origins such as carrier-acoustic phonon and phonon-phonon scattering are not considered here as they have a picosecond or larger time constant[11]. So, the relaxation time, τ_1, only consists of the excited carriers' intraband electron-electron and electron-optical phonon scattering rates. Furthermore, as τ_1 is ultrafast, it is only valid within the energy band of half of the photon energy covered by the bandwidth of the incident laser light. This is the energy band of our interest to investigate the operation principle of the saturable absorber.

The detailed simulation process can be found in Chapter 2.

A2: FDTD Simulation

All FDTD simulations are done using commercially available Lumerical Inc. FDTD software. The optical surface conductivity of non-pumped graphene is extracted from the numerical model included in the software and the time dependent optical surface conductivity of pumped graphene is calculated using the numerical model described above. All other material parameters (Gold, Aluminum, and Aluminum Oxide-Al_2O_3) are also obtained from the simulation software database. Time-dependent FDTD simulation process details can be found in Chapter 2 and Chapter 3.

We sample optical surface conductivity ($\sigma_S(t,\omega)$) of graphene at different points of time in accordance with the change of non-equilibrium carrier accumulation, $N(t,\omega)$, in graphene CB (which is responsible for the instantaneous change in optical conductivity). Then we use these $\sigma_S(t,\omega)$, at different time, as conductivity input for the graphene 2D conductivity model in Lumerical FDTD to do the time dependent FDTD simulation of our device. The process of selecting optical surface conductivity at different points of time is depicted in Figure 105. Pump fluence used in this simulation is 0.0566μJ/cm^2.

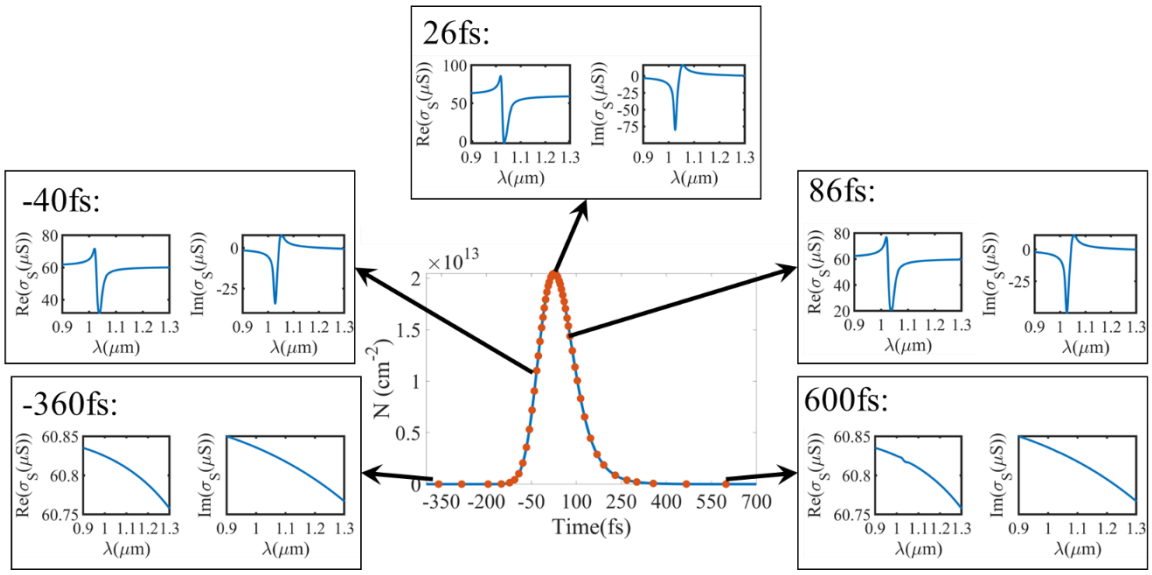


Figure 105: Process of Selecting Different Point of Time on the Carrier Density Plot and Finding Graphene Conductivity at Each of Those Times. FDTD Simulation Is Done with Corresponding Surface Conductivity of Graphene for Each of the Selected Points. This Way We Can Perform a Time-dependent Simulation for Our Graphene-plasmonic Hybrid metasurface Device (GPMSA).

A3: Time dependent FDTD Simulation in a Self-Consistent way

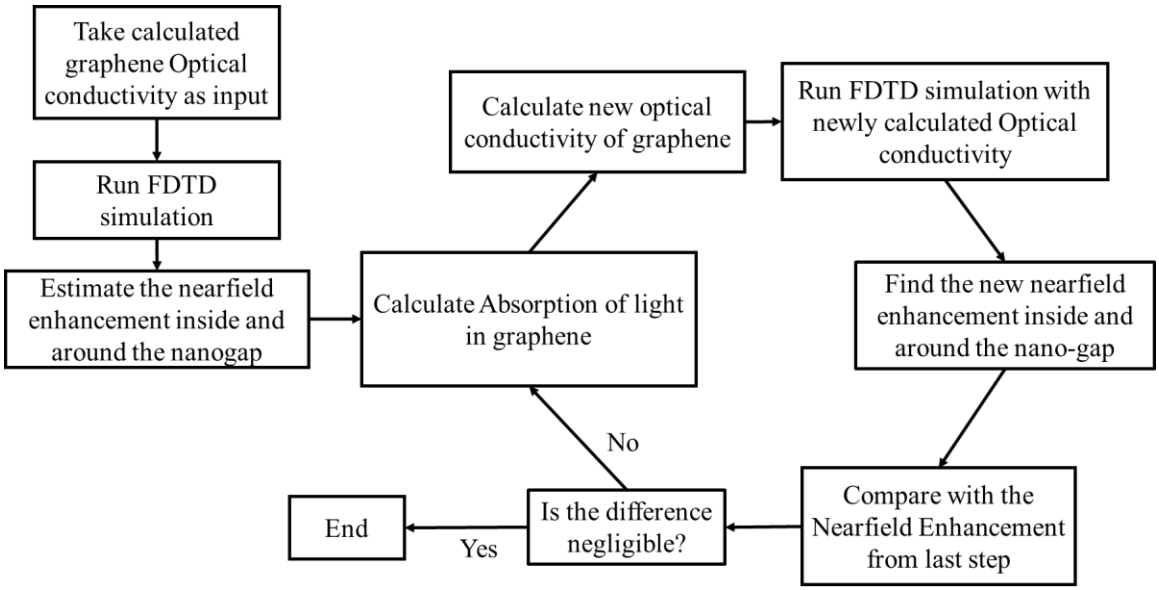


Figure 106: Flow Chart of Time Dependent FDTD Simulation in a Self-consistent Way

The process of doing time dependent FDTD simulation in a self-consistent way is shown in Figure 106. We have conducted the simulation for some preselected time points on the input pulse, which we selected in Figure 105. Graphene conductivity is calculated based on the process described in section 2.3.1. After the initial interaction of laser light with our device (GPMSA), there will be a large near field enhancement around the nanogap.

Graphene on the plasmonic metasurface is modelled considering four different regions according to the enhanced near field in different regions around the nano-gap. These regions are indicated by R1, R2, R3 and R4. For a plasmonic nano-antenna design with gap 30nm and width of 100nm; area of R1 is 15nmx 105nm, R2 is 7.5nmx 105nm and R3 is 120nmx 31nm and the rest is R4. These are illustrated in Figure 107.

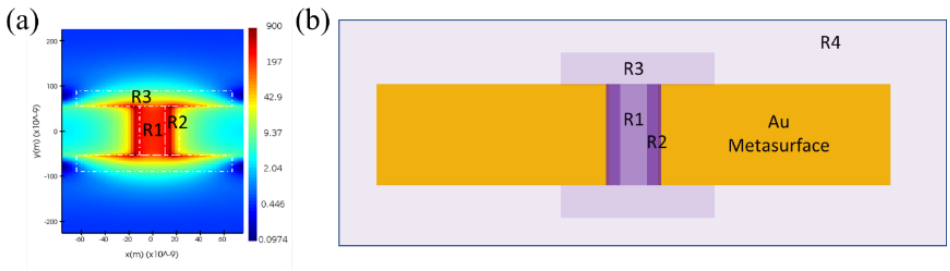


Figure 107: (a) Different Regions of Graphene Experience Different near Field Enhancement. (b) Based on (a) Graphene Is Modeled in Four Different Regions. R1- Middle of Nano Gap, R2- near the Edge of Nano-gap, R3- on the Side of Nano-gap and Antenna, R4-rest of the Graphene Regions Covering the GPMSA and Surroundings. GPMSA Device Parameters Used in All Simulations Presented in This Manuscript: Au Nanobar Length, L=120nm, Width, W=100nm, Gap, G=30nm, Thickness, t_{Au}=40nm, X-period, P_x=150nm, Y-period, P_y=450nm and Al_2O_3 Thickness, t_{AlO}=40nm

The average near field enhancement is calculated for the regions discussed in Figure 107. This near field enhancement is multiplied with the incident pulse intensity to get the new incident peak intensity. This new pulse intensity is used to calculate graphene's new optical conductivity for each indicated regions (R1, R2 and R3). Then we use this new optical conductivity to simulate our device in FDTD again to obtain the new near field enhancement in those regions. Comparing the difference in the near field enhancement, between this step and the previous, we can set a stop to this loop. The entire process is illustrated in Figure 108 for incident pump fluence of 0.0566μJ/cm2.

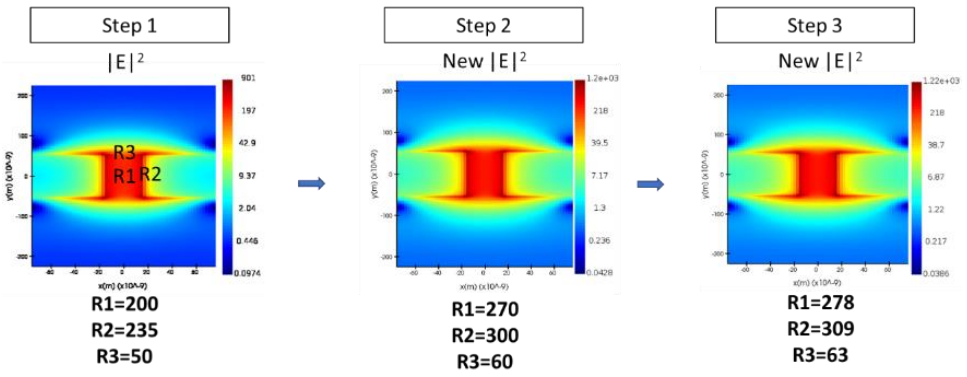


Figure 108: Step 1: Near Field Enhancement Around the Nanogap for the Input Optical Conductivity of Graphene. The Average near Field Enhancement for Different Regions Are Indicated below the near Field-enhancement for Each Step. New Optical Conductivity Is Calculated Based on the Near-field Enhancement Obtained in Step 1 . Step 2- the Resulted near Field Enhancement after Using the Newly Calculated Optical Conductivity of Graphene Is Shown Here. As the near Field Enhancement in Different Regions Are Quite Different Compared to Step 1, We Use This New near Field Enhancement to Calculate the New Optical Conductivity of Graphene. Step 3- We Use the Newly Calculated Optical Conductivity to Do Another Simulation of Our Device. It Seems the near Field Enhancement after This Simulation Is Very Close to the Results in Step 2.

A4: Graphene Complex Refractive Index (n+ik) and Absorption Coefficient Calculation

The transmission, reflection and absorption from the monolayer graphene depends on the multiple reflections happening inside the cavity formed inside graphene surrounded by air, illustrated in Figure 109a and b.

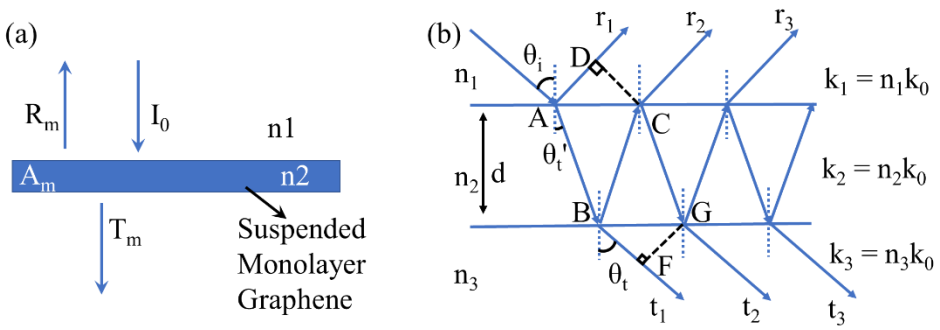


Figure 109: (a) Suspended Monolayer Graphene Surrounded by Air. (b) Reflection, Transmission, and Absorption Inside Monolayer Graphene Depends on the Multiple Reflection in the Cavity Formed Inside Graphene.

Total reflection coefficient considering multiple reflection inside the cavity,

$$r_{tot} = r_{12} + t_{12}r_{23}t_{21}e^{i(2\varphi_h - \triangle\varphi_r)}\frac{1}{1 - r_{21}r_{23}e^{i(2\varphi_h - \triangle\varphi_r)}}$$

Total transmission coefficient considering multiple reflection inside the cavity,

$$t_{tot} = t_{12}\, e^{i\varphi_h}\, t_{23}\frac{1}{1 - r_{23}r_{21}e^{i(2\varphi_h - \triangle\varphi_t)}}$$

Here, r_{ij} = reflection coefficient of an interface when light going from i medium to j medium and t_{ij} = transmission coefficient of an interface when light going from i medium to j medium

Path traversed inside material with index n2 (graphene) is AB and BC. Phase accumulated by AB and BC is: $\varphi_h = k_2\, d/\cos(\theta_t')$

Phase difference introduced due to path difference is provided as follows

$\Delta\varphi_r = k_1\,(2d)\tan(\theta_t')\sin(\theta_i) = 2k_2 d\ (\sin^2(\theta_t')/\cos(\theta_t'))$

$\Delta\varphi_t = k_1 (2d) \tan(\theta_t') \sin(\theta_i) = 2k_2 d \ (\sin^2(\theta_t')/\cos(\theta_t'))$

For vertical incidence, $\varphi_h = k_2 d$ and $\Delta\varphi_r = \Delta\varphi_t = 0$

So, total reflection and transmission coefficient becomes

$$r_{tot} = r_{12} + t_{12} r_{21} t_{21} e^{i(2k_2 d)} \frac{1}{1 - r_{21} r_{21} e^{i(2k_2 d)}} \quad (60)$$

$$t_{tot} = t_{12}\, e^{ik_2 d}\, t_{21} \frac{1}{1 - r_{21} r_{21} e^{i(2k_2 d)}} \quad (61)$$

Here, medium 3 and 1 are same, air. So, n1 =1 and n2= n+ik.

Reflection and transmission coefficient from ach interface is provided below

$$r_{12} = \frac{n_1 - n_2}{n_1 + n_2} = \frac{1 - (n + ik)}{1 + (n + ik)} \qquad t_{12} = \frac{2n_1}{n_1 + n_2} = \frac{2}{1 + (n + ik)}$$

$$r_{21} = \frac{n_2 - n_1}{n_1 + n_2} = \frac{(n + ik) - 1}{1 + (n + ik)} \qquad t_{21} = \frac{2n_2}{n_1 + n_2} = \frac{2(n + ik)}{1 + (n + ik)}$$

And propagation constant or wavenumber inside graphene is

$$k_2 = n_2 k_0 = (n + ik) \frac{2\pi}{\lambda}$$

So, $e^{i(2k_2 d)} = e^{i(2(n+ik)\frac{2\pi}{\lambda} d)} = e^{2(in-k)\frac{2\pi}{\lambda} d} = e^{i2n\frac{2\pi}{\lambda} d - 2k\frac{2\pi}{\lambda} d} = e^{i2n\frac{2\pi}{\lambda} d - \alpha d} = e^{i2n\frac{2\pi}{\lambda} d} e^{-\alpha d}$

Absorption coefficient of graphene,

$$\alpha = 2k \frac{2\pi}{\lambda} = 2k \frac{2\pi f}{c} = 2k \frac{\omega}{c}$$

Graphene material properties, under laser excitation, is derived from the coupled differential equation in previous section.

A5: Plasmonic nanoantenna fabrication

On a Silicon substrate, 150nm Aluminum (Al) layer was deposited using electron beam evaporation (PVD 75, Kurt J. Lesker Company®). On top of Aluminum layer, a 20nm Aluminum Oxide (Al_2O_3) layer is deposited using atomic layer deposition (Cambridge Savannah ALD) as a dielectric spacer layer. The dielectric layer was spin coated with two layers of electron beam resists. One layer of poly (methyl methacrylate) (495K PMMA with 2% Anisole, ~100nm) and on top another layer of methyl methacrylate (950K MMA with 2% Anisole, ~70nm). On top of the resists a very thin layer (~5-10nm) of Chromium (Cr) was deposited as a charge dissipation layer. Electron Beam Lithography (EBL, JEOL JBX-6000FS) was used to pattern the double layer resist. After patterning, the sample was developed in cold temperature (4ºC) in a solution with 1-part Methyl Isobutyl Ketone (MIBK) and 3-part Isopropyl alcohol (IPA) to remove the exposed resists. Then a thin layer (~5nm) of Cr and 35nm thick Gold (Au) layer was deposited using thermal evaporation (Edwards Auto 306). The lift-off was done by soaking the sample in hot acetone for 12 hours followed by acetone and IPA rinse.

A6: Graphene Transfer

We used a CVD graphene sample on a Cu foil (covered both sides). We cut a small piece, enough to cover the whole area of the fabricated device and put it on a glass slide. To hold the small piece in its place, we used one drop of water on the glass slide. Then we spin-coated 950K A4 PMMA layer (at 2500 RPM for 1 minute) on top of the graphene piece to protect the graphene layer on that side from the following processing steps. Then we flip the small piece and expose the bare graphene side of the Cu foil. We take another glass slide and use Kapton tape on all four sides of the piece to hold it on the glass slide. Then put it inside an oxygen(O_2) plasma etcher, for 15 minutes to etch the graphene off that side

of Cu foil. After the process is done, we remove the graphene-covered Cu foil (on one side now) and cut off the four edges of the foil as they were protected by the Kapton tape during the plasma etching and still has graphene on both sides. Then we put the graphene-covered Cu foil (Cu side down) in a copper etchant (a mixture of $CuCl_2$ and HCl). After the etching is completed, we can see a translucent piece of graphene (covered with PMMA on top) floating on the etchant. We scooped the floating graphene piece from the etchant with a small piece of SiO_2/Si wafer and put it inside a Deionized (DI) water bath. We provide three separate DI water baths (each for 5 minutes) to the graphene piece to remove all the Cu etchants. After the last bath, we move the graphene sample to another DI water bath. Then we take our device (where we want to transfer graphene) and scoop the graphene sample out of the DI water with it (basically transferring graphene on our device). Then we gently blow the sample with an N_2 gun (optional as it may tear graphene) to remove any water puddle underneath graphene and then let it dry on its own for 24 hours. After that, we put the sample in an acetone bath for 5 minutes and then rinsed it with acetone and IPA to remove all the PMMA from the top. Then blow-dry with an N_2 gun. Raman spectrum of the monolayer graphene is presented in Figure 14.

A7: Device reflection spectra measurement

Bruker Vertex 70 FTIR spectrometer connected to a Hyperion 2000 microscope is used to measure the reflection spectra of the fabricated devices. A linear polarizer was used on the optical light path to align the light polarization along the long axis of nanoantenna. The reflected light from the sample was collected with a 15x objective (NA=0.4). A liquid Nitrogen cooled Mercury Cadmium Telluride (MCT) photodetector was used to detect the collected reflected light from the sample. A background spectrum of the sample substrate

(Al2O3/Al/Si) is taken and subtracted from all the measurements to remove any effect of the substrate.

A8: The Fitting Function (and Why We Are Using It?)

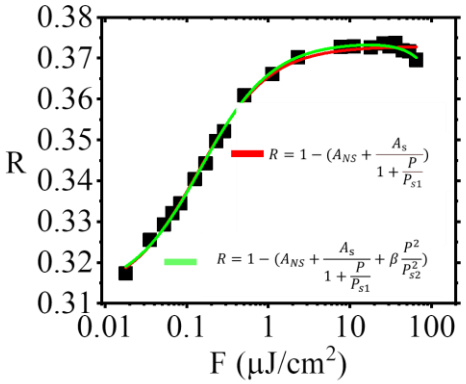


Figure 110: Reflection Profile of Our Hybrid Metasurfae with Changing Pump Fluence. Solid Lines Are the Fitted Curve for Two Different Equations and Solid Green Squares Are the Measured Points.

Most saturable absorption effects can be modeled with simplified two-level atom model described in[129],

$$R = 1 - (A_{NS} + \frac{A_s}{1 + \frac{F}{F_s}}) \tag{62}$$

Here, A_{NS} is non-saturable absorption coefficient. A_s is saturable absorption coefficient and F_s is the saturation fluence.

In Figure 110, fitting with equation (62) deviates from the measured absorption for pump fluence larger than 1μJ/cm^2. It happens because of the large nonlinear effects produced in the hybrid metasurface at larger pump fluences which causes two photon absorption and bends the absorption profile upward around 40μJ/cm^2 [164,165]. This behavior is also seen very prominently in other hybrid metasurface devices (with same design with different

parameters) presented in Figure 19(b) and Figure 20(b) and Figure 21(b). So, we added a two-photon absorption term with equation (62) to fit our measurement at higher pump fluence more accurately[165]. The modified two-atom model with two-photon absorption term is given below,

$$R = 1 - (A_{NS} + \frac{A_s}{1 + \frac{F}{F_{s1}}} + \beta \frac{F^2}{F_{s2}^2}) \tag{63}$$

Here, β is the two-photon absorption coefficient and F_{S2} is the two photon saturation fluence.

Here, two-photon absorption initiates at a very large pump fluence. So, in general, F_{s2} is substantially larger compared to F_{s1}.

For, incident pump fluence close to the first saturation level or when $F \approx F_{s1} \Rightarrow A \approx A_{NS} + \frac{A_s}{1+\frac{F}{F_{s1}}}$; i.e., A two-level system can reasonably explain this part of the curve.

For pump fluence larger than F_{s1} or $F >> F_{s1} \Rightarrow A \approx A_{NS}$; It reaches the non-saturable absorption level.

For power close to the second saturation level or $F \approx F_{s2} \Rightarrow A \approx \beta \frac{F^2}{F_{s2}{}^2}$; This part is dominated by two-photon absorption.

The red curve in Figure 110 shows fitting with equation (63). From the figure we can see, that it can fit the measured absorption very well at all pump fluences.

A9: Degenerate Pump-Probe Measurement

The ultrafast self-modulation is demonstrated by a degenerate (same color) pump-probe measurement. The measurement setup is illustrated in Figure 18a. A femtosecond Ytterbium fiber laser (MenloSystems) is used to generate the pump and probe pulses at

1035nm with 100MHz repetition rate and ~100fs pulse width. The pump beam is focused on the sample at 45˚ angle with a parabolic mirror and probe beam is focused at a normal angle on the sample with a ZeSe objective lens (NA=0.13). The reflected probe pulse is shrunk and focused with two plano-convex lenses on a biased free-space InGaAs photodetector (DET08CL, 5GHz, Thorlabs Inc.) and displayed by a mixed domain oscilloscope (DSA-X 91604A, Agilent Technologies) with 16GHz bandwidth. As the bandwidth and the response time of the photodetector is not enough to resolve the changes in the reflection of the probe pulses, we used a chopper (chopping frequency ~1200Hz) on the pump path and use it as a reference (5f option selected in chopper controller) for a phase sensitive lock-in-amplifier. The output of the photodetector is connected to the lock-in-amplifier and the DC output of the lock-in-amplifier is connected with a source meter and ultimately the source meter reading is stored automatically by a computer using MATLAB instrument control toolbox. The probe pulse delay is controlled by an optical delay line (ODL 220, Thorlabs Inc.). The movement of the ODL and the data recording from the lock-in-amplifier is synchronized and automated with the computer so that first the ODL is moved to change the delay and then a data is recorded.

APPENDIX B

OFF-CENTER CAMERA POSITION CORRECTION, MAGNETOMETER OFFSET, DOCP CALCULATION AND PARTICLE DENSITY CALCULATION

D1: Off-Center Camera Correction:

The Figure 111, Figure 112 and Figure 113 shows the geometric schematics used to derive the mathematical formula to obtain the correction.

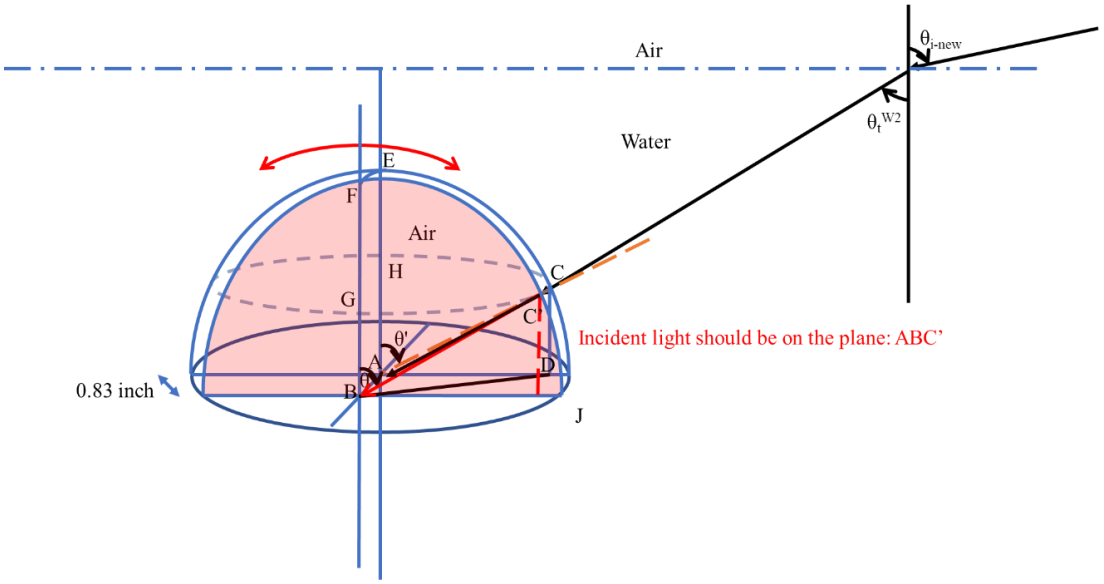


Figure 111: Schematic of the Dome Indicating the Plane of Rotation of the Camera (BJF) and the Plane Which Contains the Actual Light Direction (ABC')

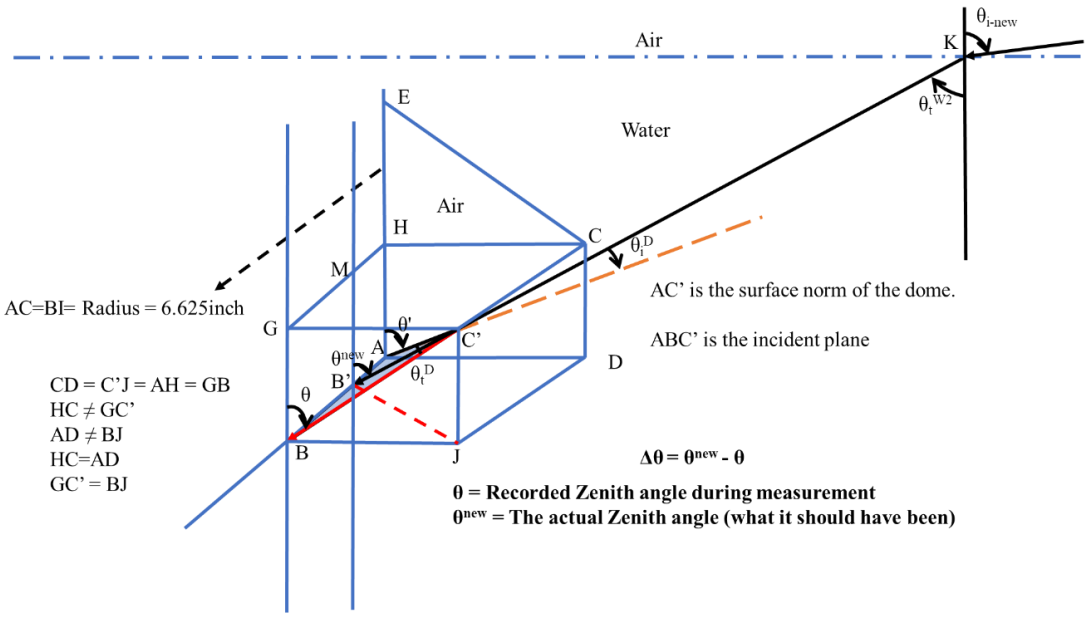


Figure 112: A Schematic Showing the Relationship Between Different Parts of the Structure

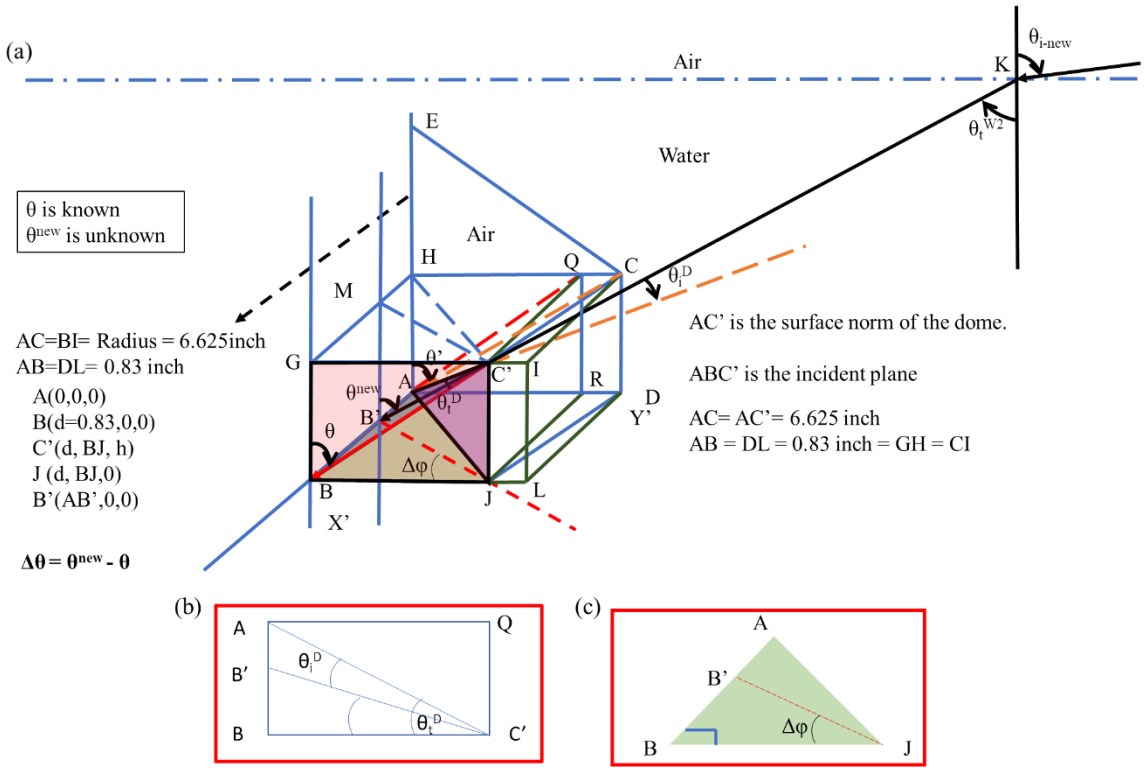


Figure 113: Schematic to Show Different Planes Used in Derivation for (b) Zenith Angle Correction and (c) Azimuth Angle Correction

<u>Zenith Angle Correction for off-center camera position:</u>

$$sin(\theta_t^D) = \frac{AB}{AC'} = \frac{AB}{AC}$$

According to Snell's law,

$$n_{air}\, sin(\theta_t^D) = n_{water}\, sin(\theta_i^D)$$

From Figure 113 (b),

$$tan(\theta_t^D - \theta_i^D) = \frac{BB'}{BC'}$$

In $\Delta ABC'$:

$$AB^2 + BC'^2 = AC'^2$$

$$BC' = \sqrt{(AC'^2 - AB^2)}$$

$$BB' = tan(\theta_t^D - \theta_i^D)BC'$$

$$BB' = tan(\theta_t^D - \theta_i^D)\sqrt{(AC'^2 - AB^2)}$$

$$AB' = AB - BB' = AB - tan(\theta_t^D - \theta_i^D)BC'$$

$$AB' = AB - tan(\theta_t^D - \theta_i^D)\sqrt{(AC'^2 - AB^2)}$$

$$BJ = BC' sin(\theta)$$

$$BG = BC'^{cos(\theta)} = h$$

$$B'C' = \sqrt{((d - AB')^2 + BJ^2 + h^2)}$$

Now,

$$cos(\theta^{new}) = \frac{B'M}{B'C'} = \frac{C'J}{B'C'} = \frac{C'J}{\sqrt{((d - AB')^2 + BJ^2 + h^2)}}$$

$$cos(\theta^{new}) = \frac{BC' cos(\theta)}{\sqrt{((d - (AB - tan(\theta_t^D - \theta_i^D)\sqrt{(AC'^2 - AB^2)}))^2 + BJ^2 + h^2)}}$$

$$cos(\theta^{new})$$

$$= \frac{(\sqrt{(AC'^2 - AB^2)}\, cos(\theta))}{\sqrt{((d - (AB - tan(\theta_t^D - \theta_i^D)\sqrt{(AC'^2 - AB^2)}))^2 + (\sqrt{(AC'^2 - AB^2)}\, sin(\theta))^2 + (\sqrt{(AC'^2 - AB^2)}\, cos(\theta))^2)}}$$

$$cos(\theta^{new})$$

$$= \frac{(\sqrt{(r^2 - d^2)}\, cos(\theta))}{\sqrt{((d - (d - tan(\theta_t^D - \theta_i^D)\sqrt{(r^2 - d^2)}))^2 + (\sqrt{(r^2 - d^2)}\, sin(\theta))^2 + (\sqrt{(r^2 - d^2)}\, cos(\theta))^2)}}$$

$$cos(\theta^{new})$$

$$= \frac{(\sqrt{(r^2 - d^2)}\, cos(\theta))}{\sqrt{((d - d + tan(\theta_t^D - \theta_i^D)\sqrt{(r^2 - d^2)})^2 + (\sqrt{(r^2 - d^2)}\, sin(\theta))^2 + (\sqrt{(r^2 - d^2)}\, cos(\theta))^2)}}$$

$$cos(\theta^{new})$$

$$= \frac{(\sqrt{(r^2 - d^2)}\, cos(\theta))}{\sqrt{((tan(\theta_t^D - \theta_i^D)\sqrt{(r^2 - d^2)})^2 + (\sqrt{(r^2 - d^2)}\, sin(\theta))^2 + (\sqrt{(r^2 - d^2)}\, cos(\theta))^2)}}$$

$$cos(\theta^{new}) = \frac{cos(\theta)}{\sqrt{((tan(\theta_t^D - \theta_i^D))^2 + (sin(\theta))^2 + (cos(\theta))^2)}}$$

$$\theta^{new} = \frac{a\,cos(cos(\theta))}{\sqrt{((tan(\theta_t^D - \theta_i^D))^2 + (sin(\theta))^2 + (cos(\theta))^2)}}$$

So, the zenith error $\Delta\theta = \theta^{new} - \theta$

Azimuth Angle Correction for off-center camera position:

From Figure 113(c),

$$sin(\Delta\varphi) = \frac{BB'}{B'J}$$

$$BB' = tan(\theta_t^D - \theta_i^D)\sqrt{(AC^2 - AB^2)}$$

$$B'J = \sqrt{((d - AB')^2 + BJ^2)}$$

$$AB' = AB - tan(\theta_t^D - \theta_i^D)\sqrt{(AC'^2 - Ab^2)}$$

$$BJ = BC'\,sin(\theta)$$

$$sin(\Delta\varphi) = \frac{(tan(\theta_t^D - \theta_i^D)\sqrt{(AC'^2 - AB^2)})}{\sqrt{((d - AB')^2 + BJ^2)}}$$

$$sin(\Delta\varphi)$$

$$= \frac{(tan(\theta_t^D - \theta_i^D)\sqrt{(AC'^2 - AB^2)})}{\sqrt{((d - \left(AB - tan(\theta_t^D - \theta_i^D)\sqrt{(AC'^2 - AB^2)}\right))^2 + (\sqrt{(AC'^2 - AB^2)}\,sin(\theta))^2)}}$$

$$sin(\Delta\varphi) = \frac{(tan(\theta_t^D - \theta_i^D)\sqrt{(r^2 - d^2)})}{\sqrt{((d - \left(d - tan(\theta_t^D - \theta_i^D)\sqrt{(r^2 - d^2)}\right))^2 + (\sqrt{(r^2 - d^2)}\,sin(\theta))^2)}}$$

$$sin(\Delta\varphi) = \frac{(tan(\theta_t^D - \theta_i^D))}{\sqrt{((tan(\theta_t^D - \theta_i^D))^2 + (sin(\theta))^2)}}$$

So, Azimuth angle error,

$$\Delta\varphi = \frac{a\,sind((tan(\theta_t^D - \theta_i^D))}{\sqrt{((tan(\theta_t^D - \theta_i^D))^2 + (sin(\theta))^2)}}$$

D2: Magnetometer off-set calculation:

A simple process of calculating the magnetometer offset is provided below:

1. Read and record the sun azimuth angle from the magnetometer when pointing toward the sun. Also, record the zenith angle of the sun. Also, record which camera (LP or CP) is used for this step.
2. Record the time of doing step 1.
3. Calculate the sun azimuth angle using the time recorded in step 2 for the measurement location.
4. Subtract the sun azimuth angle calculated in step 3 from step 2. This difference is a mixture of magnetometer deviation and off-center camera error.
5. Calculate the off-center camera error for the zenith angle recorded in step 1. Then subtract (add) the error if using CP camera (LP camera) in step 1.

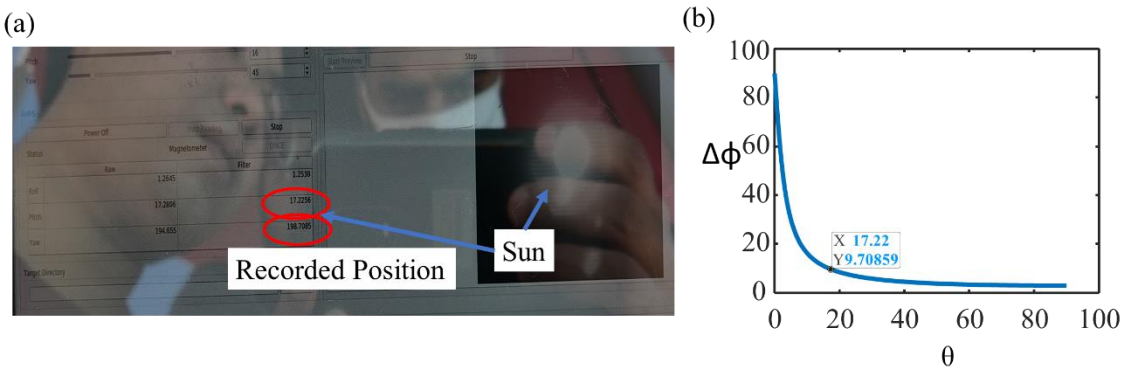


Figure 114: For the Swimming Pool Measurement on 04/19/2021 at 2.28pm (a) Taken Image in the Middle of Measurements Using Cp Camera (1.33 Inch from Center) to Calculate Magnetometer Offset. The Recorded Sun Azimuth is 198.7085º Is the Reference Azimuth and the Zenith is 17.22. Time of Taking This Photo: 12:27:13pm Sun Position at That Time: AZ: 180.1465 and ZEN: 22.5450. (B) Calculated Off-center Camera Error (Azimuth) at 17.22 Zenith Angle.

From the Figure 114, we can calculate the magnetometer offset following the procedure explained, Magnetometer offset: 198.7085 - 180.1465 - 9.71 = 8.852˚.

We can also find the magnetometer offset from CP and LP intensities recorded. The procedure is as follows: Read the sun position from the CP intensity (or LP intensity) (center of the circular overexposed data). Find-out what is the time of taking that image frame. Calculate the sun position at that time. Subtract the actual sun position from the sun position obtained from the CP intensity (or LP intensity). Then subtract the off-center camera correction for CP camera or add for LP camera.

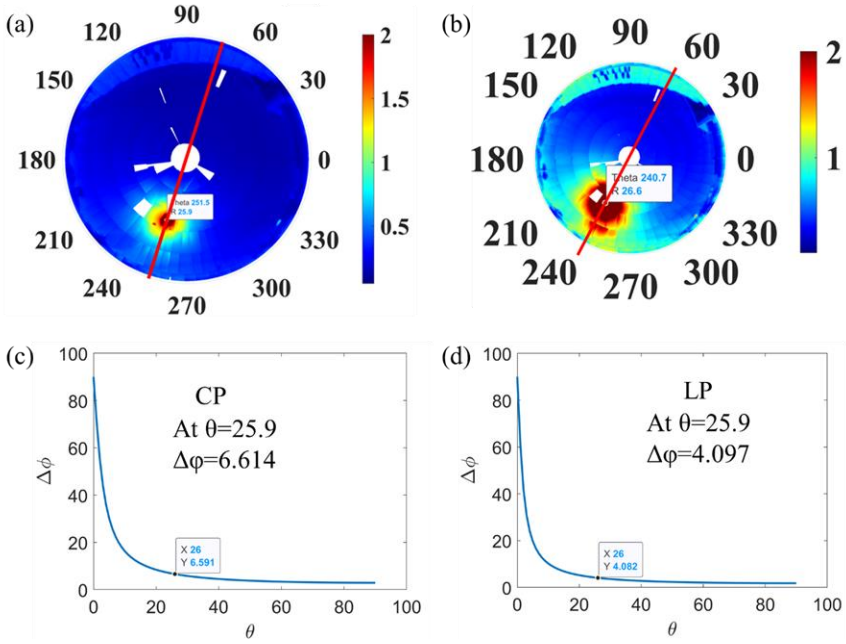


Figure 115: For the Swimming Pool Measurement on 04/19/2021 at 2.28pm (a) CP Intensity (b) CP Intensity (c) Off-center Camera Correction for CP Camera (d) Off-center Camera Correction for LP Camera

From CP Intensity (Figure 115 (a)):

Recorded sun position: AZ: 251.5 and ZEN: 25.9 (from CP intensity profile)

Actual Sun position at 2.28pm: AZ = 237.8330 and ZEN = 35.6424

Off-center camera correction at 25.9 zenith angle: 6.591˚

Difference or the magnetometer deviation: 251.5 – 237.833 – 6.614 (off cent. corr) = 7.053º

From LP Intensity (Figure 115 (b)):

Recorded sun position: AZ: 240.7 and ZEN: 26.6 (from LP intensity profile)

Actual Sun position at 2.28pm: AZ = 237.8330 and ZEN = 35.6424

Off-center camera correction at 26.6 zenith angle: 4˚

Difference or the magnetometer deviation: 240.7 – 237.833 + 4 (off cent. corr) = 6.8670º

The magnetometer offset calculated from CP and LP intensities are almost the same. To be consistent over many calculations, we can use the average of these two numbers.

D3: DoCP Calculation:

The steps to calculate the DoCP:

1. Read LP and CP intensities from LP and CP camera. Even though these two components have the same recorded azimuth and zenith angle (as we only have one gyroscope), they have different off-center errors mixed with them.
2. Calculate the off-center error for both of these cameras.
3. Make corrections to the off-center error for both LP and CP intensities. Now, these two components have different azimuth and zenith angles.
4. Find CP and LP intensities at the same azimuth and zenith angles of the LP camera.
5. Calculate DoCP using the following equation, DoCP= 2*CP intensity/0.9 – LP intensity. We need to divide the CP intensity by 0.9 to compensate for the reduced intensity due to the CP filter used with the CP camera.
6. Add the LP camera off-center error to the DoCP data to reverse back the correction.
7. Then find DoCP at the same azimuth angles recorded with the LP camera in step 1.

D4: Particle Density Calculation from lab measurement:

During our underwater measurement we collected water from the lake and brought it to the laboratory and filled a 27cm x 10cm water tank. For this measurement we used a

unpolarized light source (a battery powered torch), a CMOS camera and the larger side of the tank (27cm side). First, we measured the intensity of light without the water tank (I_{01}) and with the water tank without water (I_{w1}). Second, we measured the intensity of light with the water tank filled with water (I_{w2}). As we are using a quite unstable light source, we measured the light intensity without any water tank (I_{02}) again. The measurement process is illustrated in Figure 116.

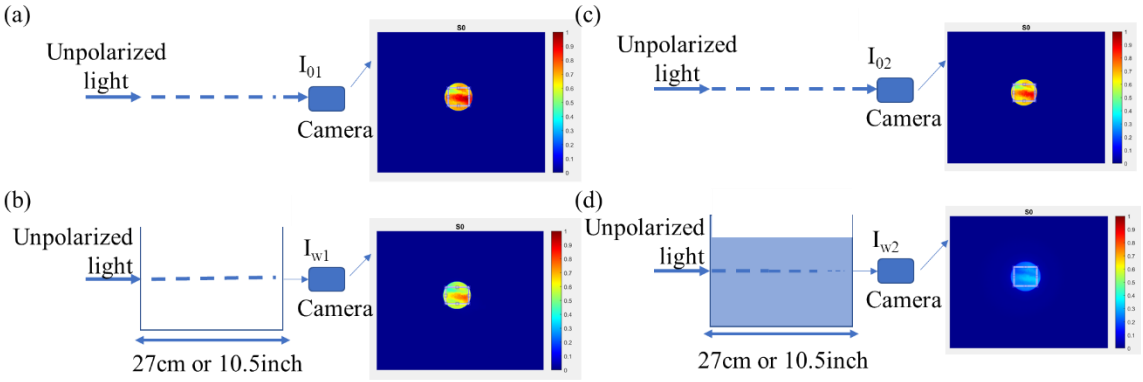


Figure 116: (a, b) Capturing the Intensity of Light Without and with the Water Tank to Set a Baseline for Measurement (c, d) Measuring the Intensity of Light Without and with the Water Tank Full of Lake Water

We calculated the mean free path (MFP) by using the following equation otaiend from Beer-Lambert's law,

$$I = I_0 exp^{-x/MFP}$$

$$\Rightarrow MFP = -\frac{x}{ln\left(\frac{I}{I_0}\right)}$$

$$\Rightarrow MFP = -\frac{x}{ln\left(\frac{I_{w2}/I_{02}}{I_{w1}/I_{01}}\right)}$$

$$\Rightarrow MFP = -\frac{27}{ln\left(\frac{38.99}{81.1012}\right)}$$

$$\Rightarrow MFP = 36.8655\ cm$$

Here, the mean free path is around 37cm.

For the number density we take an elaborate route using the Mie scattering theory. In this case, we can calculate the attenuation coefficient of the water by using the measured intensities and the Beer-Lambert's law,

$$I = I_0 exp^{-L_w \alpha}$$

$$\Rightarrow L_w = -\frac{1}{\alpha} ln\left(\frac{I}{I_0}\right)$$

$$\Rightarrow \alpha = -\frac{1}{L_w} ln\left(\frac{I_{w2}/I_{02}}{I_{w1}/I_{01}}\right)$$

Here, L_w = water tank length = 27cm

So, with this attenuation coefficient and using the Mie theory we can back calculate the number density needed to produce the calculated attenuation coefficient. We just need to run the Mie code obtained from [71,159] by assuming different maximum particle concentration for the size distribution found in DLS measurement. We calculate the attenuation coefficient for different particle concentration dn compare with the attenuation coefficient calculated from the measurement. If the attenuation coefficient match with the calculated on we stop the code. The process is illustrated in Figure 117. We can also calculate the MFP from this process and compare with the one we calculated before to verify the process. In this process I have assumed the particle refractive index is 1.572 (silica) and water refractive index is 1.33.

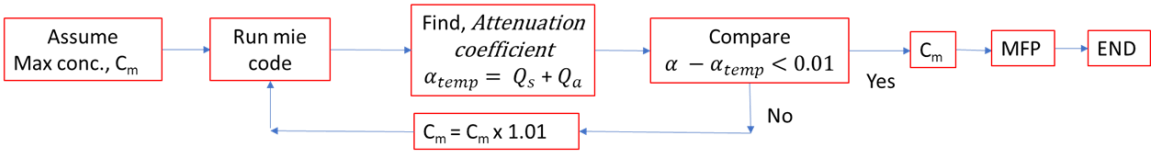


Figure 117: Calculation Process Flow to Obtain the Particle Density. Mie code can be found in[159].

APPENDIX C

MATLAB CODE TO CALCULATE TIME AND WAVELENGTH DEPENDENT GRAPHENE OPTICAL CONDUCTIVITY UNDER INTENSE AND ULTRAFAST LASER EXCITATION

Main Code to calculate Graphene Optical Conductivity:

```
clc
clear all
close all
save_mat_data=1 %make it 1 if you want to save the data to do FDTD simulation
plot_time_dependent=0
plot_wavelength_and_time_dependent=0
near_field_enhancement_all=[1];
sech2_pulse=0;
main_laser_wavelength_all=[1.035]*1e-6;
main_laser_BW=0.017e-6; %in m
la=linspace(0.7,2,1000);
number_of_time=1e3;
light_velocity=3e8;
incident_avg_power_all=[55e3:50e3:300e4]*1e-3; %Average power of laser measured by
power meter in W
pulse_width_all=100*1e-15; %in s
pulse_rep_rate_all=100e6; %Repetition rate in Hz or 1/s
beam_diameter_all=150e-6; %Diameter of beam (in m)
tau1_all=[20]*1e-15;%relaxation time in s
for nfe_c=1:length(near_field_enhancement_all)
   near_field_enhancement=near_field_enhancement_all(nfe_c);
for mlw_c=1:length(main_laser_wavelength_all)
   main_laser_wavelength=main_laser_wavelength_all(mlw_c);
for inc_p_c=1:length(incident_avg_power_all)
   incident_avg_power=incident_avg_power_all(inc_p_c);
for bd_c=1:length(beam_diameter_all)
   beam_diameter=beam_diameter_all(bd_c);
for pr_c=1:length(pulse_rep_rate_all)
   pulse_rep_rate=pulse_rep_rate_all(pr_c);
for pw_c=1:length(pulse_width_all)
   pulse_width=pulse_width_all(pw_c);
for tau1_c=1:length(tau1_all)
   tau1=tau1_all(tau1_c);
   close all
time=linspace(-400,700,number_of_time); %delay in fs
laser_spectrum_from_image=dlmread('1035nm laser spectra.txt');
laser_spectrum_from_image(:,1)=laser_spectrum_from_image(:,1)*1e-
6+(main_laser_wavelength-1.0375*1e-6);
laser_spectrum_from_image_freq=light_velocity./laser_spectrum_from_image(:,1);
%we have to do the integration in frequency or omega
f_ini=light_velocity/la(end)/1e-6;
f_end=light_velocity/la(1)/1e-6;
disp('f_ini=');
disp(f_ini);
disp('f_end=');
```

```
disp(f_end);
f_all=linspace(f_ini,f_end,length(la));
main_laser_frequency=light_velocity/main_laser_wavelength;
main_laser_BW_f=light_velocity*(1/(main_laser_wavelength-main_laser_BW/2)-
1/(main_laser_wavelength+main_laser_BW/2)); %in um
laser_spectrum_from_image_for_interpolation=[laser_spectrum_from_image_freq
laser_spectrum_from_image(:,2)];
laser_spectrum_from_image_for_interpolation=sortrows(laser_spectrum_from_image_fo
r_interpolation,1);
laser_spectrum_freq=incident_avg_power*interp1(laser_spectrum_from_image_for_inter
polation(:,1),laser_spectrum_from_image_for_interpolation(:,2),f_all,'linear','extrap');
h_cut=6.626e-34/2/pi;
fermi_velocity=1e6;
omega=2*pi.*f_all;
en=h_cut*omega;
DoS_1=2*en/(pi*h_cut^2*fermi_velocity^2)/2.*en.*laser_spectrum_freq/max(laser_spec
trum_freq);
[k_collect,alpha_collect,absorption_collect,nn_collect,CB_carrier_density_collect,f_E_t_
VB_Collect,f_E_t_CB_Collect]=get_carrier_dynamics(f_all,laser_spectrum_freq,main_l
aser_BW_f,pulse_width,pulse_rep_rate,beam_diameter,tau1,time,number_of_time,sech2
_pulse,near_field_enhancement);
delta_alpha_collect=(alpha_collect-alpha_collect(1,:));
dummy_central_freq_position=find(abs(f_all-main_laser_frequency)<(f_all(2)-f_all(1)));
dummy=abs(f_all(dummy_central_freq_position)-main_laser_frequency);
[v1,p1]=min(dummy);
central_frequency_position=dummy_central_freq_position(p1);
%Find where (in time) is the maximum carrier density for the central wavelength of the
laser
[max_val,carrier_density_max_pos]=max(CB_carrier_density_collect(:,central_frequenc
y_position));
la_new=light_velocity./f_all;
E_la=en;
if plot_time_dependent==1

figure(5)
plot(time,f_E_t_VB_Collect(:,central_frequency_position),'linewidth',2)
xlabel('time(fs)')
ylabel('VB Ocupation probability')
set(gca, 'FontSize', 14);

figure(6)
plot(time,f_E_t_CB_Collect(:,central_frequency_position),'linewidth',2)
xlabel('time(fs)')
ylabel('CB Ocupation probability')
set(gca, 'FontSize', 14);
figure(7)
```

```
hold on
plot(time,CB_carrier_density_collect(:,central_frequency_position)/1e4,'linewidth',2)
xlabel('time(fs)')
ylabel('Carrier density (cm^-^2)')
set(gca, 'FontSize', 14);

figure(8)
hold on
plot(time,k_collect(:,central_frequency_position+9),'linewidth',2)
xlabel('time(fs)')
ylabel('k')
hYLabel = get(gca,'YLabel');
set(hYLabel,'rotation',0,'VerticalAlignment','middle')
set(gca, 'FontSize', 14);

%%
figure(9)
plot(time,absorption_collect(:,central_frequency_position),'linewidth',2)
xlabel('time(fs)')
ylabel('Absorption')
hYLabel = get(gca,'YLabel');
% set(hYLabel,'rotation',0,'VerticalAlignment','middle')
set(gca, 'FontSize', 14);

figure(10)
plot(time,alpha_collect(:,central_frequency_position),'linewidth',2)
xlabel('time(fs)')
ylabel('\alpha')
hYLabel = get(gca,'YLabel');
set(hYLabel,'rotation',0,'VerticalAlignment','middle')
set(gca, 'FontSize', 14);
figure(11)
plot(time,delta_alpha_collect(:,central_frequency_position),'linewidth',2)
xlabel('time(fs)')
ylabel('\delta\alpha')
hYLabel = get(gca,'YLabel');
set(gca, 'FontSize', 14);

%%
figure(12)
plot(la_new/1e-6,alpha_collect(carrier_density_max_pos,:),'linewidth',2)
xlabel('\lambda(\mum)')
ylabel('\alpha')
hYLabel = get(gca,'YLabel');
set(gca, 'FontSize', 14);
legend('Maximum Excitation')
```

```
figure(13)
plot(la_new/1e-6,alpha_collect(1,:),'linewidth',2)
xlabel('\lambda(\mum)')
ylabel('\alpha')
hYLabel = get(gca,'YLabel');
set(gca, 'FontSize', 14);
legend('No Excitation')

figure(14)
plot(la_new/1e-6,delta_alpha_collect(carrier_density_max_pos,:),'linewidth',2)
xlabel('\lambda(\mum)')
ylabel('\delta\alpha')
hYLabel = get(gca,'YLabel');
set(gca, 'FontSize', 14);

figure(15)
plot(la_new/1e-6,absorption_collect(carrier_density_max_pos,:),'linewidth',10)
xlabel('\lambda(\mum)')
ylabel('Absorption')
xlim([1 1.07])
hYLabel = get(gca,'YLabel');
set(gca, 'FontSize', 45);
set(gca, 'XTick', [1:.02:1.07]);
set(gca, 'FontName', 'Times New Roman')
set(gca, 'linewidth',5)
set(gcf,'color','w');

figure(16)
plot(la_new/1e-6,k_collect(1,:),'linewidth',2)
xlabel('\lambda(\mum)')
ylabel('k')
legend('No Excitation')
hYLabel = get(gca,'YLabel');
set(gca, 'FontSize', 14);
legend('No Excitation')

figure(17)
hold on
plot(la_new/1e-6,k_collect(carrier_density_max_pos,:),'linewidth',2)
xlabel('\lambda(\mum)')
ylabel('k')
hYLabel = get(gca,'YLabel');
set(gca, 'FontSize', 14);
legend('Maximum Excitation')
```

```
figure(18)
surf(la_new/1e-6,time,real(alpha_collect),'Edgecolor','none')
colorbar
xlabel('\lambda(\mum)')
ylabel('time(fs)')
zlabel('\alpha')
hzLabel = get(gca,'ZLabel');
set(gca, 'FontSize', 14);

figure(19)
surf(la_new/1e-6,time,CB_carrier_density_collect,'Edgecolor','none')
colorbar
colormap(parula(512))
xlabel('\lambda(\mum)')
ylabel('time(fs)')
zlabel('N (m^-^2)'); %in Conduction Band
hzLabel = get(gca,'ZLabel');
hyLabel = get(gca,'yLabel');
hxLabel = get(gca,'xLabel');
ylim([-200 310])
xlim([1.017 1.067])
ll=0; %DoCP lower limit (ll)
ml=3e17; % DoCP max limit (ml)
step=5e16; %DoCP colorbar tick step
caxis([ll, ml]); %change colorbar limit
cbh=colorbar();
set(cbh,'YTick',[ll:step:ml]); % Chnage color bar ticks
grid off
set(hyLabel,'rotation',-20,'VerticalAlignment','middle')
set(hxLabel,'rotation',10,'VerticalAlignment','middle')
set(gca, 'FontSize', 30);

figure(20)
surf(E_la/1.6e-19,time,CB_carrier_density_collect,'Edgecolor','none')
colorbar
colormap(parula(512))
xlabel('E(eV)')
ylabel('time(fs)')
zlabel('N (m^-^2)'); %in Conduction Band
hzLabel = get(gca,'ZLabel');
hyLabel = get(gca,'yLabel');
hxLabel = get(gca,'xLabel');
ylim([-200 310])
xlim([1.17 1.23])
ll=0; %DoCP lower limit (ll)
ml=3e17; % DoCP max limit (ml)
```

```
step=5e16; %DoCP colorbar tick step
caxis([ll, ml]); %change colorbar limit
cbh=colorbar();
set(cbh,'YTick',[ll:step:ml]); % Chnage color bar ticks
grid off
set(hyLabel,'rotation',-20,'VerticalAlignment','middle')
set(hxLabel,'rotation',10,'VerticalAlignment','middle')
set(gca, 'FontSize', 36);

figure(210)
surf(la_new/1e-6,time,absorption_collect*100,'Edgecolor','none')
colorbar off
colormap(parula(512))
xlabel('\lambda(\mum)')
ylabel('time(fs)')
zlabel('A(%)'); %in Conduction Band
hzLabel = get(gca,'ZLabel');
hyLabel = get(gca,'yLabel');
hxLabel = get(gca,'xLabel');
ylim([-200 310])
xlim([1 1.07])
zlim([1.38 2.4])
grid off
set(hyLabel,'rotation',-32,'VerticalAlignment','middle')
set(hxLabel,'rotation',18,'VerticalAlignment','middle')
set(gca, 'FontSize', 45);
set(gca,'XTick',[1:0.02:1.07]);
set(gca,'YTick',[-300:150:300]);
set(gca,'ZTick',[1:0.2:2.4]);
set(gca, 'FontName', 'Times New Roman')
set(gca, 'linewidth',5)
set(gcf,'color','w');
figure(21)
contourf(time,E_la/1.6e-19,CB_carrier_density_collect','edgecolor','none','linewidth',5)
colormap(hot(512))
ylabel('E(eV)')
xlabel('time(fs)')
zlabel('N (m^-^2)'); %in Conduction Band
hzLabel = get(gca,'ZLabel');
hyLabel = get(gca,'yLabel');
hxLabel = get(gca,'xLabel');
xlim([-150 200])
ylim([1.17 1.23])
cbh=colorbar();
set(gca,'XTick',[-400:100:400]);
set(gcf,'color','w');
```

```
grid off
set(gca, 'FontSize', 35);
end
 %% KK Relationship
omega_new=2*pi*f_all;
graphene_thickness=0.30e-9;
d_omega=(omega_new(2)-omega_new(1));
delta_n_collect=zeros(number_of_time,length(f_all));
graphene_index_collect=zeros(number_of_time,length(f_all));
graphene_permittivity_collect=zeros(number_of_time,length(f_all));
graphene_conductivity_collect=zeros(number_of_time,length(f_all));
parfor c2=1:length(f_all)
laser_frequency=f_all(c2);
omega=2*pi*laser_frequency;
col_r1=1:c2-1;
col_r2=c2+1:length(f_all);
frequency_range1=f_all(col_r1);
frequency_range2=f_all(col_r2);
delta_alpha_collect1=delta_alpha_collect(:,col_r1);
delta_alpha_collect2=delta_alpha_collect(:,col_r2);
omega_range1=2*pi*frequency_range1;
omega_range2=2*pi*frequency_range2;
A_Y1=zeros(number_of_time,length(omega_range1));
A_Y2=zeros(number_of_time,length(omega_range2));
for i=1:length(omega_range1)
   A_Y1(:,i)=delta_alpha_collect1(:,i)./(omega_range1(i)^2-omega^2);
End
for i=1:length(omega_range2)
   A_Y2(:,i)=delta_alpha_collect2(:,i)./(omega_range2(i)^2-omega^2);
end
if c2==1
   inte_1=0;
else
   inte_1=zeros(number_of_time,1);
   for t_cc1=1:size(A_Y1,1)
      inte_1(t_cc1,1)=sum(A_Y1(t_cc1,:));
   end
end
if c2==length(f_all)
   inte_2=0;
else
   inte_2=zeros(number_of_time,1);
   for t_cc2=1:size(A_Y2,1)
      inte_2(t_cc2,1)=sum(A_Y2(t_cc2,:));
   end
end
```

```
n=light_velocity/pi.*(inte_1+inte_2)*d_omega;
delta_n_collect(:,c2)=n;
end
delta_k_collect=delta_alpha_collect*light_velocity/2./omega_new;
d=0.3e-9; %thickness of graphene (m)
h=6.62607004e-34; %planks constant in m2kg/s or Js
h_cut=h/2/pi;
ec=1.6e-19; %electron charge
kB=1.38e-23; %in JK-1
T=300; %in K
epsilon_0=8.854e-12; %in Fm^-1
Ef=0.2; %in eV
kbt=kB*T;
w=omega_new;
%%Read Sigma obtained from FDTD software
sigma_RPA_temp=dlmread('Graphene_Sigma_From_FDTD_Ef_0.2eV_scat_0.01eV_T_
300K.txt',',0,0);
sigma_RPA_temp(:,1)=3e8./sigma_RPA_temp(:,1)/1e-6;%Convert wavelength to
frequency
sigma_RPA_temp=sortrows(sigma_RPA_temp,1);
sigma_RPA_R=interp1(sigma_RPA_temp(:,1),sigma_RPA_temp(:,2),f_all,'linear','extrap
');
sigma_RPA_I=interp1(sigma_RPA_temp(:,1),sigma_RPA_temp(:,3),f_all,'linear','extrap'
);
sigma_RPA=sigma_RPA_R+1i*sigma_RPA_I;
eps_RPA=1+1i*sigma_RPA./w/epsilon_0/d;
ng_RPA=sqrt(eps_RPA);
n_RPA=real(ng_RPA);
k_RPA=imag(ng_RPA);
%%%%%%%%%%%%%modification (See Appendix A)
ratio_Graphene_n_k =k_collect(1,:)./k_RPA;
for i=1:size(delta_k_collect,1)
   delta_k_collect_mod(i,:)=delta_k_collect(i,:)./ ratio_Graphene_n_k;
   delta_n_collect_mod(i,:)=delta_n_collect(i,:)./ ratio_Graphene_n_k;
end
n_total=n_RPA+delta_n_collect_mod;
k_total=k_RPA+delta_k_collect_mod;
%%%%%%%%%%%%%
graphene_index=n_total+1i*k_total;
epsilon_0=8.854e-12; %in Fm^-1
graphene_permittivity=graphene_index.^2;
graphene_conductivity=(graphene_permittivity-
1).*omega_new*epsilon_0*graphene_thickness./1i;
graphene_index_collect=graphene_index;
graphene_permittivity_collect=graphene_permittivity;
graphene_conductivity_collect=graphene_conductivity;
```

```
if plot_wavelength_and_time_dependent==1
%% plot n and k obtained from FDTD
figure
plot(la_new/1e-6,real(ng_RPA),'linewidth',2)
xlabel('\lambda(\mum)')
ylabel('n_{FDTD}')
hYLabel = get(gca,'YLabel');
set(gca, 'FontSize', 14);

figure
plot(la_new/1e-6,imag(ng_RPA),'linewidth',2)
xlabel('\lambda(\mum)')
ylabel('k_{FDTD}')
hYLabel = get(gca,'YLabel');
set(gca, 'FontSize', 14);

%%
figure
plot(time,real(graphene_index_collect(:,central_frequency_position)),'linewidth',2)
xlabel('time(fs)')
ylabel('n')
hYLabel = get(gca,'YLabel');
set(hYLabel,'rotation',0,'VerticalAlignment','middle')
set(gca, 'FontSize', 14);

figure
plot(time,imag(graphene_index_collect(:,central_frequency_position)),'linewidth',2)
xlabel('time(fs)')
ylabel('k')
hYLabel = get(gca,'YLabel');
set(hYLabel,'rotation',0,'VerticalAlignment','middle')
set(gca, 'FontSize', 14);

figure
plot(time,real(graphene_permittivity_collect(:,central_frequency_position)),'linewidth',2)
xlabel('time(fs)')
ylabel('Re(\epsilon)')
hYLabel = get(gca,'YLabel');
set(hYLabel,'rotation',90,'VerticalAlignment','middle')
set(gca, 'FontSize', 14);

figure
plot(time,imag(graphene_permittivity_collect(:,central_frequency_position)),'linewidth',2
)
xlabel('time(fs)')
```

```
ylabel('Im(\epsilon)')
hYLabel = get(gca,'YLabel');
set(hYLabel,'rotation',90,'VerticalAlignment','middle')
set(gca, 'FontSize', 14);

%%
figure
plot(time,real(graphene_conductivity_collect(:,1))/1e-6,'linewidth',2)
hold on
plot(time,real(graphene_conductivity_collect(:,central_frequency_position))/1e-
6,'linewidth',2)
xlabel('time(fs)')
ylabel('Re(\sigma) (uS)')
hYLabel = get(gca,'YLabel');
set(gca, 'FontSize', 14);

figure
plot(time,imag(graphene_conductivity_collect(:,1))/1e-6,'linewidth',2)
hold on
plot(time,imag(graphene_conductivity_collect(:,central_frequency_position))/1e-
6,'linewidth',2)
xlabel('time(fs)')
ylabel('Im(\sigma) (uS)')
hYLabel = get(gca,'YLabel');
set(gca, 'FontSize', 14);

%%
figure
plot(la_new/1e-6,real(graphene_conductivity_collect(end,:))/1e-6,'linewidth',2)
xlabel('\lambda(\mum)')
ylabel('Re(\sigma) (uS)')
hYLabel = get(gca,'YLabel');
set(gca, 'FontSize', 14);

figure
plot(la_new/1e-6,imag(graphene_conductivity_collect(end,:))/1e-6,'linewidth',2)
xlabel('\lambda(\mum)')
ylabel('Im(\sigma) (uS)')
hYLabel = get(gca,'YLabel');
set(gca, 'FontSize', 14);

figure
plot(la_new/1e-6,real(graphene_index_collect(1,:)),'linewidth',2)
xlabel('\lambda(\mum)')
ylabel('n')
legend('No Excitation')
```

```
hYLabel = get(gca,'YLabel');
set(gca, 'FontSize', 14);

figure
plot(la_new/1e-6,imag(graphene_index_collect(1,:)),'linewidth',2)
xlabel('\lambda(\mum)')
ylabel('k')
legend('No Excitation')
hYLabel = get(gca,'YLabel');
set(gca, 'FontSize', 14);

figure
plot(la_new/1e-6,real(graphene_permittivity_collect(1,:)),'linewidth',2)
xlabel('\lambda(\mum)')
ylabel('re(\epsilon)')
legend('No Excitation')
hYLabel = get(gca,'YLabel');
set(gca, 'FontSize', 14);

figure
plot(la_new/1e-6,imag(graphene_permittivity_collect(1,:)),'linewidth',2)
xlabel('\lambda(\mum)')
ylabel('im(\epsilon)')
legend('No Excitation')
hYLabel = get(gca,'YLabel');
set(gca, 'FontSize', 14);

%%
re_sigma_wl_dependent_at_min_alpha=real(graphene_conductivity_collect(carrier_densi
ty_max_pos,:));
im_sigma_wl_dependent_at_min_alpha=imag(graphene_conductivity_collect(carrier_de
nsity_max_pos,:));

figure
plot(la_new/1e-6,re_sigma_wl_dependent_at_min_alpha/1e-6,'linewidth',2)
xlabel('\lambda(\mum)')
ylabel('Re(\sigma) (uS)')
legend('Maximum Excitation')
hYLabel = get(gca,'YLabel');
set(gca, 'FontSize', 14);

figure
plot(la_new/1e-6,im_sigma_wl_dependent_at_min_alpha/1e-6,'linewidth',2)
xlabel('\lambda(\mum)')
ylabel('Im(\sigma) (uS)')
legend('Maximum Excitation')
```

```
hYLabel = get(gca,'YLabel');
set(gca, 'FontSize', 14);

figure
plot(la_new/1e-6,real(graphene_index_collect(carrier_density_max_pos,:)),'linewidth',2)
xlabel('\lambda(\mum)')
ylabel('n')
legend('Maximum Excitation')
hYLabel = get(gca,'YLabel');
set(gca, 'FontSize', 14);

figure
plot(la_new/1e-
6,imag(graphene_index_collect(carrier_density_max_pos,:)),'linewidth',2)
xlabel('\lambda(\mum)')
ylabel('k')
legend('Maximum Excitation')
hYLabel = get(gca,'YLabel');
set(gca, 'FontSize', 14);
%%
figure
subplot(2,1,1)
plot(la_new/1e-6,real(graphene_permittivity_collect(1,:)),'linewidth',2)
hold on
plot(la_new/1e-
6,real(graphene_permittivity_collect(carrier_density_max_pos,:)),'linewidth',2)
xlabel('\lambda(\mum)')
ylabel('re(\epsilon)')
xlim([0.9 1.3])
legend('With No Excitation','With Laser Excitation')
hYLabel = get(gca,'YLabel');
set(gca, 'FontName', 'Times New Roman')
set(gca, 'FontSize', 40);
% figure
subplot(2,1,2)
plot(la_new/1e-6,imag(graphene_permittivity_collect(1,:)),'linewidth',2)
set(gca, 'FontSize', 14);
hold on
plot(la_new/1e-
6,imag(graphene_permittivity_collect(carrier_density_max_pos,:)),'linewidth',2)
xlabel('\lambda(\mum)')
ylabel('im(\epsilon)')
xlim([0.9 1.3])
hYLabel = get(gca,'YLabel');
set(gca, 'FontName', 'Times New Roman')
set(gca, 'FontSize', 40);
```

```
set(gcf,'color','w');
%%
figure
plot(la_new/1e-6,real(graphene_conductivity_collect(1,:))/1e-6,'linewidth',10)
hold on
plot(la_new/1e-6,real(graphene_conductivity_collect(carrier_density_max_pos,:))/1e-
6,'linewidth',10)
xlabel('\lambda(\mum)')
ylabel('Re(\sigma_S(\muS))')
xlim([0.9 1.3])
ylim([-5 90])
legend('No Excitation','With Excitation')
legend boxoff
hYLabel = get(gca,'YLabel');
set(gca,'XTick',[.9:0.1:1.3]);
set(gca, 'FontName', 'Times New Roman')
set(gca, 'FontSize', 60);
set(gca, 'linewidth', 5);

figure
plot(la_new/1e-6,imag(graphene_conductivity_collect(1,:))/1e-6,'linewidth',10)
hold on
plot(la_new/1e-6,imag(graphene_conductivity_collect(carrier_density_max_pos,:))/1e-
6,'linewidth',10)
xlabel('\lambda(\mum)')
ylabel('Im(\sigma_S(\muS))')
xlim([0.9 1.3])
ylim([-82 20])
legend('No Excitation','With Excitation')
legend boxoff
hYLabel = get(gca,'YLabel');
set(gca, 'FontName', 'Times New Roman')
set(gca,'XTick',[.9:0.1:1.3]);
set(gca,'YTick',[-75:25:25]);
set(gca, 'FontSize', 60);
set(gcf,'color','w');
set(gca, 'linewidth', 5);
%%
figure
plot(time,real(graphene_conductivity_collect(:,1))/1e-6,'linewidth',10)
hold on
plot(time,real(graphene_conductivity_collect(:,central_frequency_position))/1e-
6,'linewidth',10)
xlabel('Time (fs)')
ylabel('Re(\sigma_S(\muS))')
xlim([-300 400])
```

```
ylim([-5 70])
legend('No Excitation','With Excitation','FontSize',40)
legend boxoff
hYLabel = get(gca,'YLabel');
set(gca,'XTick',[-300:150:600]);
set(gca, 'FontName', 'Times New Roman')
set(gca, 'FontSize', 60);
set(gca, 'linewidth', 5);
figure
plot(time,imag(graphene_conductivity_collect(:,1))/1e-6,'linewidth',10)
hold on
plot(time,imag(graphene_conductivity_collect(:,central_frequency_position+9))/1e-
6,'linewidth',10)
xlabel('Time (fs)')
ylabel('Im(\sigma_S(\muS))')
xlim([-300 400])
legend('No Excitation','With Excitation','FontSize',40)
legend boxoff
hYLabel = get(gca,'YLabel');
set(gca, 'FontName', 'Times New Roman')
set(gca,'XTick',[-300:150:400]);
set(gca,'YTick',[-75:25:25]);
set(gca, 'FontSize', 60);
set(gcf,'color','w');
set(gca, 'linewidth', 5);
%% For Paper Only Graphene
figure
plot(time,real(graphene_conductivity_collect(:,central_frequency_position))/1e-
6,'linewidth',10)
xlabel('Time (fs)')
ylabel('Re(\sigma_S(\muS))')
xlim([-300 300])
ylim([56 62])
hYLabel = get(gca,'YLabel');
set(gca,'XTick',[-300:150:600]);
set(gca, 'FontName', 'Times New Roman')
set(gca, 'FontSize', 60);
set(gca, 'linewidth', 5);

figure
plot(time,imag(graphene_conductivity_collect(:,central_frequency_position))/1e-
6,'linewidth',10)
xlabel('Time (fs)')
ylabel('Im(\sigma_S(\muS))')
xlim([-300 300])
ylim([-5 1])
```

```
hYLabel = get(gca,'YLabel');
set(gca, 'FontName', 'Times New Roman')
set(gca,'XTick',[-300:150:400]);
set(gca,'YTick',[-75:2:25]);
set(gca, 'FontSize', 60);
set(gcf,'color','w');
set(gca, 'linewidth', 5);

%% For Paper GPMSA
figure
plot(time,real(graphene_conductivity_collect(:,central_frequency_position))/1e-
6,'linewidth',10)
xlabel('Time (fs)')
ylabel('Re(\sigma_S(\muS))')
xlim([-300 300])
ylim([-20 62])
hYLabel = get(gca,'YLabel');
set(gca,'XTick',[-300:150:600]);
set(gca, 'FontName', 'Times New Roman')
set(gca, 'FontSize', 60);
set(gca, 'linewidth', 5);

figure
plot(time,imag(graphene_conductivity_collect(:,central_frequency_position))/1e-
6,'linewidth',10)
xlabel('Time (fs)')
ylabel('Im(\sigma_S(\muS))')
xlim([-300 300])
ylim([-35 1])
hYLabel = get(gca,'YLabel');
set(gca, 'FontName', 'Times New Roman')
set(gca,'XTick',[-300:150:400]);
set(gca,'YTick',[-40:8:2]);
set(gca, 'FontSize', 60);
set(gcf,'color','w');
set(gca, 'linewidth', 5);

%%
figure
plot(la_new/1e-6,(absorption_collect(1,:))*100,'linewidth',10)
hold on
plot(la_new/1e-6,(absorption_collect(carrier_density_max_pos,:))*100,'linewidth',10)
xlabel('\lambda(\mum)')
ylabel('A(%)')
xlim([1 1.2])
legend('No Excitation','With Excitation')
```

```
legend boxoff
hYLabel = get(gca,'YLabel');
set(gca, 'FontName', 'Times New Roman')
set(gca, 'FontSize', 45);
set(gca,'XTick',[1:0.02:1.1]);
set(gca, 'linewidth', 5);

%%
figure
plot(time,(absorption_collect(:,1))*100,'linewidth',10)
hold on
plot(time,(absorption_collect(:,central_frequency_position))*100,'linewidth',10)
xlabel('time(fs)')
ylabel('A(%)')
xlim([-400 600])
legend('No Excitation','With Excitation')
legend boxoff
hYLabel = get(gca,'YLabel');
set(gca, 'FontName', 'Times New Roman')
set(gca, 'FontSize', 45);
set(gca, 'linewidth', 5);

figure
surf(la_new/1e-6,time,real(graphene_conductivity_collect)/1e-6,'Edgecolor','none')
colormap(parula(512))
xlabel('\lambda(\mum)')
ylabel('time(fs)')
zlabel('Re(\sigma_S(\muS))'); %in Conduction Band
hzLabel = get(gca,'ZLabel');
hyLabel = get(gca,'yLabel');
hxLabel = get(gca,'xLabel');
ylim([-200 310])
xlim([1 1.07])
zlim([-23 93])
grid off
set(hyLabel,'rotation',-30,'VerticalAlignment','middle')
set(hxLabel,'rotation',22,'VerticalAlignment','middle')
set(gca, 'FontSize', 45);
set(gca,'XTick',[1:0.02:1.07]);
set(gca,'YTick',[-300:150:300]);
set(gca,'ZTick',[0:25:100]);
set(gca, 'FontName', 'Times New Roman')
set(gca, 'linewidth',5)
set(gcf,'color','w');

figure
```

```
surf(la_new/1e-6,time,imag(graphene_conductivity_collect)/1e-6,'Edgecolor','none')
colormap(parula(512))
xlabel('\lambda(\mum)')
ylabel('time(fs)')
zlabel('Im(\sigma_S(\muS))'); %in Conduction Band
hzLabel = get(gca,'ZLabel');
hyLabel = get(gca,'yLabel');
hxLabel = get(gca,'xLabel');
ylim([-200 310])
xlim([.98 1.07])
zlim([-110 25])
grid off
set(hyLabel,'rotation',-30,'VerticalAlignment','middle')
set(hxLabel,'rotation',15,'VerticalAlignment','middle')
set(gca, 'FontSize', 45);
set(gca,'XTick',[.98:0.02:1.07]);
set(gca,'YTick',[-300:150:300]);
set(gca,'ZTick',[-125:50:25]);
set(gca, 'FontName', 'Times New Roman')
set(gca, 'linewidth',5)
set(gcf,'color','w');
end
    save_material_data_v2
end
end
end
end
end
end
end

  %%%%%%%%%%% Functions Used in this Code %%%%%%%%%%%%%%%

%%Density of states
function D = DOS(photon_energy)
h_cut=6.62607004e-34/2/pi;
fermi_velocity=1e6; %in m/s
D= 2*photon_energy/(pi*h_cut^2*fermi_velocity^2); %unit: J^-1 m^-2
end
%%VB and CB occupation probability dynamics
function dfdt = my_dfdt_v2(t,y,tspan,photon_energy,main_laser_BW_f,alpha_fine,pulse_intensity,tau1)
dfdt=zeros(2,1);
pulse_intensity=interp1(tspan,pulse_intensity,t);
spectral_width=6.626e-34*main_laser_BW_f; %unit J
```

```
dfdt(1)=(-y(1)+
y(2)).*(pi*alpha_fine*pulse_intensity)/DOS(photon_energy/2)/photon_energy/spectral_
width+(1-y(1))/tau1;
dfdt(2)=(-y(2)+
y(1)).*(pi*alpha_fine*pulse_intensity)/DOS(photon_energy/2)/photon_energy/spectral_
width-y(2)/tau1;
end
%%%Get carrier dynamics
function
[k_collect,alpha_collect,Absorption_collect,nn_complex,CB_carrier_density_collect,f_E
_t_VB_Collect,f_E_t_CB_Collect]=get_carrier_dynamics(f_all,laser_spectrum,main_las
er_BW_f,pulse_width,pulse_rep_rate,beam_diameter,tau1,time,number_of_time,sech2_p
ulse,near_field_enhancement)
k_collect=zeros(number_of_time,length(f_all));
alpha_collect=zeros(number_of_time,length(f_all));
CB_carrier_density_collect=zeros(number_of_time,length(f_all));
f_E_t_VB_Collect=zeros(number_of_time,length(f_all));
f_E_t_CB_Collect=zeros(number_of_time,length(f_all));
nn_complex=zeros(number_of_time,length(f_all));
Absorption_collect=zeros(number_of_time,length(f_all));
only_envelope=1;
beam_radius=beam_diameter/2; %Radius of beam in m
beam_area=pi*beam_radius^2;
%Laser charecteristics
light_velocity=3e8;
h=6.62607004e-34; %planks constant in m2kg/s or Js
h_cut=h/2/pi;
%%Carrier dynamics in graphene
fermi_velocity=1e6; %in m/s
ec=1.6e-19; %electron charge
kB=1.38e-23; %in JK-1
T=300; %in K
epsilon_0=8.854e-12; %in Fm^-1
alpha_fine=(1/4/pi/epsilon_0)*(ec^2/h_cut/light_velocity);%=0.0073   =0.023/pi     %fine
structure constant of graphene
parfor c1=1:length(f_all)
%Incident gaussian beam
freq=f_all(c1);%wavelength in um
incident_avg_power=laser_spectrum(c1);
frequency=freq; %frequency in Hz
omega=2*pi*frequency;
photon_energy=h_cut*omega;
incident_peak_power=incident_avg_power./(pulse_width*pulse_rep_rate); % Peak Power
in W
tau_pulse=pulse_width; % Gaussian (or Sech2) Pulse width in fs
if only_envelope==1
```

```
    pulse_power=incident_peak_power*exp(-2.76*(time*1e-15).^2/tau_pulse^2); %in W
    if sech2_pulse==1
        tau_pulse=pulse_width/1.76;
        pulse_power=incident_peak_power*(sech(time*1e-15/tau_pulse)).^2; %in W
    end
else
    pulse_power=incident_peak_power*exp(-2.76*(time*1e-
15).^2/tau_pulse^2).*cos(omega.*time*1e-15); %in W
end
pulse_intensity=2*pulse_power./beam_area; %in W/m2
%% VB and CB occupation probability dynamics
%initial condition
dfdt_VB_init=1;
dfdt_CB_init=0;
tspan = time*1e-15; %in s
y0 = [dfdt_VB_init dfdt_CB_init];
options=odeset('RelTol',1e-6);
[~,y]                                                =                                                ode45(@(t,y)
my_dfdt_v2(t,y,tspan,photon_energy,main_laser_BW_f,alpha_fine,near_field_enhancem
ent*pulse_intensity,tau1), tspan, y0,options);
    f_E_t_VB=y(:,1)';
    f_E_t_CB=y(:,2)';
number_of_excited_carriers_in_CB_at_each_time=(f_E_t_CB.*DOS(photon_energy/2)*
photon_energy);
Ab=pi*alpha_fine.*(f_E_t_VB-f_E_t_CB);
A=-1/d.*log(1-Ab);%absorption coefficient (m^-1)
k0=2*pi*freq;
deff=2*d;
complex_ref_indx=1/(k0*deff)*(imag(log(1-Ab))-sqrt(-1)*real(log(1-Ab)));

%%imaginary part of index
k=light_velocity*(A)/2/omega; %Its unitless
nn=(complex_ref_indx);
f_E_t_VB_Collect(:,c1)=f_E_t_VB';
f_E_t_CB_Collect(:,c1)=f_E_t_CB';
k_collect(:,c1)=k';
nn_complex(:,c1)=nn';
alpha_collect(:,c1)=A';
Absorption_collect(:,c1)=Ab';
CB_carrier_density_collect(:,c1)=number_of_excited_carriers_in_CB_at_each_time';
end
end

function [sigma]=RPA_sigma(Ef,kbt,ec,w,tau,h_bar,inter,intra)
Ef=Ef*ec;
```

```
sig_intra = (1i.*ec.^2.*kbt./(pi.*h_bar.^2.*(w+1i.*2*(tau)))).*(Ef/kbt+2*log(exp(-Ef./kbt)+1));
sig_inter = (ec^2/4/h_bar)*(0.5+1./pi*(atan((h_bar.*w-2*Ef)/2/kbt))-(1i/2/pi)*log((h_bar.*w+2*Ef).^2./((h_bar.*w-2*Ef).^2+4*(kbt).^2)));
sigma = sig_intra*intra+ sig_inter*inter;
end
```

Save Material Data:

```
% Code to save the data
% Specify location to save data
loc=strcat('C:\Users\merafiqu\Desktop\ZUbair\Gr sig v8 Enhacement-modified\lamda-',num2str(main_laser_wavelength/1e-6),'um\',num2str(near_field_enhancement));
new_directory_name=strcat('CWL=',num2str(main_laser_wavelength*1e6,"%.3f"),'um','_BW=',num2str(main_laser_BW*1e6),'um','_P=',num2str(incident_avg_power*1e3),'mW','_PW=',num2str(pulse_width*1e15),'fs','_RR=',num2str(pulse_rep_rate/1e6),'MHz','_D=',num2str(beam_diameter*1e6),'µ','_t1=',num2str(tau1*1e15),'fs','_T=',num2str(T),'k');

if sech2_pulse==1
new_directory_name=strcat('CenterWL=',num2str(main_laser_wavelength*1e6,"%.3f"),'um','_BW=',num2str(main_laser_BW*1e6),'um','_P=',num2str(incident_avg_power*1e3),'mW','_PW=',num2str(pulse_width*1e15),'fs','_RepR=',num2str(pulse_rep_rate/1e6),'MHz','_Dia=',num2str(beam_diameter*1e6),'um','_tau1=',num2str(tau1*1e15),'fs','_Temp=',num2str(T),'k_Sech2');
end

mkdir(loc,new_directory_name);
loc=strcat(loc,'\',new_directory_name);

temp_pos=[-350 -280 -200 -160 -135 -120 -107 -95 -84 -75 -67 -60 -53 -48 -43 -38 -34 -30 -26 -22 -18 -15 -10 -5 0 5 10 15 18 22 26 30 34 39 43 48 55 63 70 80 92 110 130 150 185 220 250 300 400 520];
time_save_pos=temp_pos + (find(time>0,1)-1); %carrier_density_max_pos=399
if time_save_pos(1)<0
   time_save_pos=time_save_pos-time_save_pos(1)+1;
end
if time_save_pos(end)>length(time)
   time_save_pos=time_save_pos-(time_save_pos(end)-length(time));
end
center_wavelength_check=3e8/f_all(central_frequency_position)/1e-6 %in um
%%
figure
plot(time,CB_carrier_density_collect(:,central_frequency_position)/1e4,'linewidth',5)
hold on
plot(time(time_save_pos),CB_carrier_density_collect(time_save_pos,central_frequency_position)/1e4,'.','MarkerSize',50)
xlabel('Time(fs)')
```

```
ylabel('N (cm^-^2)')
set(gca, 'FontSize', 40);
set(gca,'XTick',[-350:150:700]);
set(gca, 'FontName', 'Times New Roman')
if save_mat_data==1
saveas(gcf,(strcat(loc,'\Conduction band carrier density sampling.png')))
end
%%
which_one=25;
time(temp_pos(which_one) + (find(time>0,1)-1));
%%
if save_mat_data==1
wavelength_save=1:size(graphene_conductivity_collect,2);
t_file_name=strcat(loc,'\time.txt');
fid = fopen(t_file_name, 'w');
fprintf(fid, '    time(fs)\n');
t_data_to_save=time(time_save_pos)';
save(t_file_name,'t_data_to_save','-ascii','-append')
fclose(fid);
t1=dlmread(strcat(loc,'\time.txt'),'',1,0);
for i=1:length(time_save_pos)
file_name=strcat(loc,'\',num2str(i),'sigma_at_time=',num2str(t1(i),'%.4f'),'fs','with
incident power=',num2str(incident_avg_power*1e3),'mW','.txt');
re_sigma_wl_dependent=real(graphene_conductivity_collect(time_save_pos(i),waveleng
th_save))/d;
im_sigma_wl_dependent=imag(graphene_conductivity_collect(time_save_pos(i),wavele
ngth_save))/d;
f_all_save=f_all(wavelength_save);
%%
fid = fopen(file_name, 'w');
fprintf(fid, '    Frequency       Re(Sigma)       Im(Sigma)\n');
data_to_save=[f_all_save' re_sigma_wl_dependent' im_sigma_wl_dependent'];
save(file_name,'data_to_save','-ascii','-append')
fclose(fid);
end
end
```

APPENDIX D

LUMERICAL FDTD CODE FOR TIME DEPENDENT SELF-CONSISTANT FULL WAVE SIMULATION

Main Code:

```
## Need to have the structure in a separate file, open the structure file and then run the code

clear;
power_all=[5, 10, 15, 20, 25, 40, 50, 75, 100, 150, 200, 250, 300 ,500, 800 ,1000, 2500,3000,4000,5000];
tau_all=[10:10:100];
lamda_all=[1.035];
enhance_all=[1,1,1,1]; #Initial E-field enhancement in graphene @begining of pulse

len_PC=length(power_all);
len_tau=length(tau_all);
len_lam=length(lamda_all);

loc_save_r='K:\Zubair\Project 1 Strong coupling\FDTD Simulation files\NIR_Design\Pump-probe reflection\R_NIM_G30_En_modified-latest\Self_consistant\';

for (lam_c=1; lam_c<=len_lam; lam_c=lam_c+1){
for (t_c=1; t_c<= len_tau; t_c=t_c+1){
for (pc=1; pc<= len_PC; pc=pc+1){
for(i=1; i<= 50; i=i+1){
#i=25; #for simulation at the pulse peak
self_c=1;
for (0; self_c>0;0){
switchtolayout;
?self_c; #Self-consistant loop number

c_n1='CWL='+num2str(lamda_all(lam_c),"%.3f")+'um_BW=0.017um_P='+num2str(power_all(pc))+'mW_PW=100fs_RR=100MHz_D=150um_t1='+num2str(tau_all(t_c))+'fs_T=300k';

##Graphene property in the middle of nano gap- R1
loc1='C:\Users\merafiqu\Desktop\ZUbair\Gr sig v8 Enhacement-modified\lamda-'+num2str(lamda_all(lam_c))+'um\'+num2str(enhance_all(1))+'\'+c_n1+'\';

## R2 location
loc2='C:\Users\merafiqu\Desktop\ZUbair\Gr sig v8 Enhacement-modified\lamda-'+num2str(lamda_all(lam_c))+'um\'+num2str(enhance_all(2))+'\'+c_n1+'\';

## U or D location
loc3='C:\Users\merafiqu\Desktop\ZUbair\Gr sig v8 Enhacement-modified\lamda-'+num2str(lamda_all(lam_c))+'um\'+num2str(enhance_all(3))+'\'+c_n1+'\';

##Sourrounding Graphene property location
```

```
loc4='C:\Users\merafiqu\Desktop\ZUbair\Gr sig v8 Enhacement-modified\lamda-
'+num2str(lamda_all(lam_c))+'um\'+num2str(enhance_all(4))+'\'+c_n1+'\';

time_data1=readdata(loc1+"time.txt");
time_data2=readdata(loc2+"time.txt");
time_data3=readdata(loc3+"time.txt");
time_data4=readdata(loc4+"time.txt");
num_of_mat_file=length(time_data1);

material1='C(grphene-cond) R1';
material2='C(grphene-cond) R2';
material3='C(grphene-cond) U_D';
material4='C(grphene-cond)_surrounding';

select('Graphene R1');
set("material",material1);
select('Graphene R2 R');
set("material",material2);
select('Graphene R2 L');
set("material",material2);
select('Graphene U1');
set("material",material3);
select('Graphene D1');
set("material",material3);

select('Graphene on Au_R');
set("material",material4);
select('Graphene on Au_L');
set("material",material4);

select('Graphene left');
set("material",material4);
select('Graphene Right');
set("material",material4);
select('Graphene Up');
set("material",material4);
select('Graphene Down');
set("material",material4);

row_n=getglobalmonitor("frequency points");
out_R=zeros(row_n,num_of_mat_file);
ExR_all=zeros(row_n,num_of_mat_file);
ExI_all=zeros(row_n,num_of_mat_file);

switchtolayout;
```

```
sig_data1=readdata(loc1+num2str(i)+"sigma_at_time="+num2str(time_data1(i),"%.4f")+
"fswith incident power="+num2str(power_all(pc))+"mW.txt");
sig_data2=readdata(loc2+num2str(i)+"sigma_at_time="+num2str(time_data2(i),"%.4f")+
"fswith incident power="+num2str(power_all(pc))+"mW.txt");
sig_data3=readdata(loc3+num2str(i)+"sigma_at_time="+num2str(time_data3(i),"%.4f")+
"fswith incident power="+num2str(power_all(pc))+"mW.txt");
sig_data4=readdata(loc4+num2str(i)+"sigma_at_time="+num2str(time_data4(i),"%.4f")+
"fswith incident power="+num2str(power_all(pc))+"mW.txt");
thickness=0.3e-9;

setmaterial(material1,"layer thickness enabled",true);
setmaterial(material1,"layer thickness",thickness);
setmaterial(material1,"tolerance",0.00001);
setmaterial(material2,"layer thickness enabled",true);
setmaterial(material2,"layer thickness",thickness);
setmaterial(material2,"tolerance",0.00001);
setmaterial(material3,"layer thickness enabled",true);
setmaterial(material3,"layer thickness",thickness);
setmaterial(material3,"tolerance",0.00001);
setmaterial(material4,"layer thickness enabled",true);
setmaterial(material4,"layer thickness",thickness);
setmaterial(material4,"tolerance",0.00001);
## material 1
re_sigma1=sig_data1(:,2)*.3e-9/thickness;
im_sigma1=sig_data1(:,3)*.3e-9/thickness;
f1=sig_data1(:,1); #frequency
len_f1=length(f1);
# Force real(sigma) to be close to zero if its zero or negative
for (f_c=1; f_c<= len_f1; f_c=f_c+1)
{
if (re_sigma1(f_c,1)<=0)
{
re_sigma1(f_c,1)= 1e-4;
}
}
sigma1=re_sigma1+1i*im_sigma1;
setmaterial(material1,"max coefficients",12); # set the number of coefficients
setmaterial(material1,"make fit passive",1);
setmaterial(material1,"sampled data",[f1,sigma1]);
## material 2
re_sigma2=sig_data2(:,2)*.3e-9/thickness;
im_sigma2=sig_data2(:,3)*.3e-9/thickness;
f2=sig_data2(:,1); #frequency
len_f2=length(f2);
# Force real(sigma) to be close to zero if its zero or negative
for (f_c=1; f_c<= len_f2; f_c=f_c+1)
```

```
{
if (re_sigma2(f_c,1)<=0)
{
re_sigma2(f_c,1)= 1e-4;
}
}
sigma2=re_sigma2+1i*im_sigma2;
setmaterial(material2,"max coefficients",12); # set the number of coefficients
setmaterial(material2,"make fit passive",1);
setmaterial(material2,"sampled data",[f2,sigma2]);
## material 3
re_sigma3=sig_data3(:,2)*.3e-9/thickness;
im_sigma3=sig_data3(:,3)*.3e-9/thickness;
f3=sig_data3(:,1); #frequency
len_f3=length(f3);
# Force real(sigma) to be close to zero if its zero or negative
for (f_c=1; f_c<= len_f3; f_c=f_c+1)
{
if (re_sigma3(f_c,1)<=0)
{
re_sigma3(f_c,1)= 1e-4;
}
}
sigma3=re_sigma3+1i*im_sigma3;
setmaterial(material3,"max coefficients",12); # set the number of coefficients
setmaterial(material3,"make fit passive",1);
setmaterial(material3,"sampled data",[f3,sigma3]);
## material 4
re_sigma4=sig_data4(:,2)*.3e-9/thickness;
im_sigma4=sig_data4(:,3)*.3e-9/thickness;
f4=sig_data4(:,1); #frequency
len_f4=length(f4);
# Force real(sigma) to be close to zero if its zero or negative
for (f_c=1; f_c<= len_f4; f_c=f_c+1)
{
if (re_sigma4(f_c,1)<=0)
{
re_sigma4(f_c,1)= 1e-4;
}
}
sigma4=re_sigma4+1i*im_sigma4;

setmaterial(material4,"max coefficients",12); # set the number of coefficients
setmaterial(material4,"make fit passive",1);
setmaterial(material4,"sampled data",[f4,sigma4]);
run;
```

```
out_R(:,i)=transmission("R");
out_R1=transmission("R");
out_f1=getresult("R","f");
out_lamda1=3e8/out_f1/1e-6;
out_data1=[out_lamda1,out_R1];
m="XY"; # monitor name
f_E=getdata(m,"f");
lamda_E=3e8/f_E/1e-6;
laser_lam_pos=find(lamda_E<lamda_all(lam_c)+.001,1);
Ex=getdata(m,"Ex");
Ey=getdata(m,"Ey");
x=getdata(m,"x");
y=getdata(m,"y");
Ex=pinch(Ex,3);#remove z-axis
Ey=pinch(Ey,3);#remove z axis
Ex=pinch(Ex,3,laser_lam_pos);
Ey=pinch(Ey,3,laser_lam_pos);
### Calculate the average |E|^2 over R1 region
R1_x_l=find(x/1e-6>-0.0074,1);
R1_x_m=find(x/1e-6>0.007,1);
R1_y_l=find(y/1e-6>-0.054,1);
R1_y_m=find(y/1e-6>0.052,1);
abs_E_2_R1=abs(Ex(R1_x_l:R1_x_m,R1_y_l:R1_y_m))^2+abs(Ey(R1_x_l:R1_x_m,R1
_y_l:R1_y_m))^2;
div_fact=size(abs_E_2_R1,1)*size(abs_E_2_R1,2);
abs_E_2_R1_T=sum(abs_E_2_R1);
abs_E_2_R1_avg=abs_E_2_R1_T/div_fact;
?abs_E_2_R1_avg;
### Calculate the average |E|^2 over R2 region
R2_x_l=find(x/1e-6>0.007,1);
R2_x_m=find(x/1e-6>0.016,1);
R2_y_l=R1_y_l;
R2_y_m=R1_y_m;
abs_E_2_R2=abs(Ex(R2_x_l:R2_x_m,R2_y_l:R2_y_m))^2+abs(Ey(R2_x_l:R2_x_m,R2
_y_l:R2_y_m))^2;
div_fact=size(abs_E_2_R2,1)*size(abs_E_2_R2,2);
abs_E_2_R2_T=sum(abs_E_2_R2);
abs_E_2_R2_avg=abs_E_2_R2_T/div_fact;
?abs_E_2_R2_avg;
### Calculate the average |E|^2 over R3 region
R3_x_l=find(x/1e-6>-0.062,1);
R3_x_m=find(x/1e-6>0.06,1);
R3_y_l=R1_y_m;
R3_y_m=find(y/1e-6>0.082,1);
abs_E_2_R3=abs(Ex(R3_x_l:R3_x_m,R3_y_l:R3_y_m))^2+abs(Ey(R3_x_l:R3_x_m,R3
_y_l:R3_y_m))^2;
```

```
div_fact=size(abs_E_2_R3,1)*size(abs_E_2_R3,2);
abs_E_2_R3_T=sum(abs_E_2_R3);
abs_E_2_R3_avg=abs_E_2_R3_T/div_fact;
?abs_E_2_R3_avg;
abs_E_2=[abs_E_2_R1_avg, abs_E_2_R2_avg, abs_E_2_R3_avg];
enhance_all_error=abs(round(abs_E_2)-enhance_all(:,1:3));
if (enhance_all_error(1)>1 | enhance_all_error(2)>1 | enhance_all_error(3)>1){
enhance_all(:,1:3)=round(abs_E_2);
self_c=self_c+1;
} else {
self_c=0;
}
if (self_c>3){ self_c=0;}
}
write(loc_save_r+'abs_E_2_avg_R1_R2_R3-'+num2str(power_all(pc))+'mW_tau1-
'+num2str(tau_all(t_c))+'fs_lam-
'+num2str(lamda_all(lam_c))+'um_'+num2str(i)+'.txt',num2str(abs_E_2));
write(loc_save_r+'R_'+num2str(power_all(pc))+'mW_tau1-
'+num2str(tau_all(t_c))+'fs_lam-
'+num2str(lamda_all(lam_c))+'um_'+num2str(i)+'.txt',num2str(out_data1));
write(loc_save_r+'R_'+num2str(i)+'.txt',num2str(out_data1));
}
write(loc_save_r+'time.txt',num2str(time_data1));
}
}
}
```

APPENDIX E

MATLAB CODE FOR NUMERICAL FITTING ALGORITHM TO FIND SUN POSITION FROM LIGHT POLARIZATION

```
clc
clear all
close all

%%Write all the folder names in a file and provide the file location with file name
main_folder_location='E:\Research\Projects\Underwater Geolocation\Underwater
Measurement_TBP 042122\old coordinates';
folder_location_file=strcat(main_folder_location,'\folder locations.txt')
folder_location=importdata(folder_location_file);
under_water_mapping=1;
do_off_center_cam_corr=0;
fit_with_skylight_refraction_model=1;
fit_with_PSD_model=0; % The new model that we developed
azimuth_offset_correction=0;
remove_overexposed=0;
remove_overexposer_manually=0;
if remove_overexposer_manually==1
over_exposed_data_range=dlmread(strcat(main_folder_location,'\over_exposed_data_ran
ge_Oct26.txt'),'',1,0);
end
how_many_fitting=length(folder_location)
full_data_set_fitting=0;
multiple_zenith_angles=0;
fit_at_one_zenith=0;
is_it_reduced_data_point_fitting=0;
zenith_data_is_ready=0;
sampling_method1=1;
      sample_every=1;
      sample_randomly=1;
      random_sample_size=10.948e3;
DoCP_lim_all=2;

% Zenith angle range of the data that we want to fit
zenith_low=0;
zenith_high=48;
select_data_within_range=0;
zen_cover=3;
az_cover=1;
search_in_range=5;  %Specify the range of degree to scan around the guesses
how_many_phi=50;
how_many_theta=20;
AoP_error_limit=150;
plot_it=1; %plot the fitted AoP on top of measurment AoP
find_location=0;
time_dependent=0;
```

```
number_of_iteration_for_location_prediction=2;
no_idea_of_location=1;
save_individual_fitting=1; %set it to 1 if you want to save the data for each of the fittings
save_overall_fitting=0; %set it to 1 if you want to save the data for all the fittings
save_figure=0;  %set it to 1 if you want to save the figures for each of the fittings
save_overall_fitting_in_new_dir=1; %if you want to save the fitting data in a new folder
then it should be 1;
for docp_cc=1:length(DoCP_lim_all)
DoCP_lim=DoCP_lim_all(docp_cc)
new_directory_name = strcat('Fitting result and data-
Zenith',num2str(zenith_low),'to',num2str(zenith_high),'-unwanted transition data
removed')
if fit_with_skylight_refraction_model==1
   new_directory_name=strcat(new_directory_name,'_Sky_ref_model')
elseif fit_with_PSD_model==1
   new_directory_name=strcat(new_directory_name,'_PSD_model')
end
which_location_of_measurement=5; % 1->SDF; 2->Old Main; 3-> Dr. Yao's Swimming
pool; 4-> SDFC swiming pool; 5-> Tempe Beach Park
%%
for global_count=1:how_many_fitting %how_many_fitting
   close all
file_location = folder_location{global_count};
data_file_name = '\Zubair_processed_data-Avg (Zen-0to90)-over1zen
&1az_dlt_unwanted_transition_data.txt';
time_file_name='\Zubair_processed_time-Avg (Zen-0to90)-over1zen
&1az_dlt_unwanted_transition_data.txt';
s_md = strcat(file_location,data_file_name); %location+filename for the measurment data
s_td=strcat(file_location,time_file_name);  %location+filename for the measurment time
data_set_name=strsplit(file_location,'\');
data_set_name=data_set_name{length(data_set_name)-1}
figure_title=strsplit(data_set_name,'_');
measurment_number=1
figure_title='Full map data fit'
        pos_meas=dlmread(s_md,'',1,0);
        meas_time=dlmread(s_td,'',1,0);
pos=pos_meas;
pos = pos(any(pos,2),:); %%Removes any row with all zero
%Do camera off-center correction
if do_off_center_cam_corr==1
   pos=make_off_center_camera_correction(pos);
end
% If there is any negative zenith angles then covert them to positive zenith angles
for i=1:length(pos(:,2))
   if pos(i,2)<0
      pos(i,2)=abs(pos(i,2));
```

```
    end
end
%% Full data plot
    ploar_contour_plot(pos(:,1),pos(:,2),pos(:,4),10)
%%
[pos_plot,meas_time_plot]=select_data_within_limit_greater_zen(pos,meas_time,0);
%Lower Zenith angle
[pos_plot,meas_time_plot]=select_data_within_limit_less_zen(pos_plot,meas_time_plot,
48.8); %Upper Zenith angle
        figure
        scatter(pos_plot(:,1),pos_plot(:,4))
        xlabel('Azimuth')
        ylabel('AoP')
        set(gca, 'FontSize', 18);
        set(gcf,'color','w');
        count_in_range=1;
        data_pos_in_range=[];
        for i=1:length(pos_plot)
            if pos_plot(i,4)>174 && pos_plot(i,4)<175
                data_pos_in_range(count_in_range)=i;
                count_in_range=count_in_range+1;
            end
        end
        pos_plot(data_pos_in_range,:)=[];
        clear data_pos_in_range

        figure
        scatter(pos_plot(:,1),pos_plot(:,4))
        xlabel('Azimuth')
        ylabel('AoP')
        set(gca, 'FontSize', 18);
        set(gcf,'color','w');
%%
% %%Select data within a Zenith angle limit and Azimuth angle limit
[pos,meas_time]=select_data_within_limit_greater_zen(pos,meas_time,zenith_low);
%Lower Zenith angle
[pos,meas_time]=select_data_within_limit_less_zen(pos,meas_time,zenith_high);
%Upper Zenith angle
    ploar_contour_plot(pos(:,1),pos(:,2),pos(:,4),10)
    ploar_contour_plot(pos(:,1),pos(:,2),pos(:,9),10)
    ploar_contour_plot(pos(:,1),pos(:,2),pos(:,7),10)
    cp_ll=-1; %DoCP lower limit (ll)
    cp_ml=1; % DoCP max limit (ml)
    cp_step=.25; %DoCP colorbar tick step
    caxis([cp_ll, cp_ml]); %change colorbar limit
    cbh=colorbar();
```

```
    set(cbh,'YTick',[cp_ll:cp_step:cp_ml]); % Chnage color bar ti
    count_cp_in_range=1;
    data_cp_in_range=[];
    for i=1:length(pos)
        if abs(pos(i,7))>DoCP_lim
            data_cp_in_range(count_cp_in_range)=i;
            count_cp_in_range=count_cp_in_range+1;
        end
    end
    pos(data_cp_in_range,:)=[];

    figure
    scatter(pos(:,1),pos(:,4))
    xlabel('Azimuth')
    ylabel('AoP')
    set(gca, 'FontSize', 18);
    set(gcf,'color','w');
    ploar_contour_plot(pos(:,1),pos(:,2),pos(:,4),10)
    aop_ll=0; % AoP lower limit (ll)
    aop_ml=180; % AoP max limit (ml)
    aop_step=45; %  AoP colorbar tick step
    caxis([aop_ll, aop_ml]); %Change colorbar limit
    cbh=colorbar();
    set(cbh,'YTick',[aop_ll:aop_step:aop_ml]); % Chnage color bar ticks
%%
max_zenith=max(pos(:,2));
min_zenith=min(pos(:,2));
zen_range=strcat(num2str(round(min_zenith)),'to',num2str(round(max_zenith)))
    if remove_overexposed==1
        if remove_overexposer_manually==1
        row_num=global_count;
        how_many_exposer_issue=over_exposed_data_range(row_num,2);
        t_inc=0;
            for c_i=1:how_many_exposer_issue
                start=over_exposed_data_range(row_num,3+t_inc)-
abs(over_exposed_data_range(row_num,4+t_inc));
finish=over_exposed_data_range(row_num,3+t_inc)+over_exposed_data_range(row_nu
m,5+t_inc);
                t_inc=3*c_i;
                [pos]=remove_overexposed_data(start,finish,pos);
            end
        end
    end

if azimuth_offset_correction==1
    pos(:,1)=pos(:,1)+Azimuth_offset(global_count,4);
```

```
    pos_pos=find(pos(:,1)<0);
    pos(pos_pos,1)=pos(pos_pos,1)+360;
end

if select_data_within_range==1
    [pos]=select_data(pos,meas_time,zen_cover/2,az_cover/2);
end

if sampling_method1==1
    if sample_randomly ~=1 %for sampling every other or every 50th points or something
like that
    flag1=1;
    ii=1;
    ii2=1;
    meas_time_temp=meas_time;
    clear meas_time
    while flag1==1
        if ii+sample_every-1>length(pos(:,1))
            break;
        end
        az_meas_at_zen(ii2,1)=pos(ii,1);
        zen_meas_at_zen(ii2,1)=pos(ii,2);
        AoP_meas_at_zen(ii2,1)=pos(ii,4);
        DoLP_meas_at_zen(ii2,1)=pos(ii,3);
        meas_time(ii2,:)=meas_time_temp(ii,:);
        ii=ii+sample_every-1;
        ii2=ii2+1;
    end
    clear pos;
    pos(:,1)=az_meas_at_zen;
    pos(:,2)=zen_meas_at_zen;
    pos(:,3)=DoLP_meas_at_zen;
    pos(:,4)=AoP_meas_at_zen;
    figure
    scatter(pos(:,1),pos(:,4))

    else %For sampling randomly
        rng('default');
        select_rows=sort(randperm(size(pos,1),size(pos,1)));
        pos_temp=pos;
        clear pos;
        pos(:,1)=pos_temp(select_rows,1);
        pos(:,2)=pos_temp(select_rows,2);
        pos(:,3)=pos_temp(select_rows,3);
        pos(:,4)=pos_temp(select_rows,4);
```

```
        meas_time_temp=meas_time;
        clear meas_time
        meas_time(:,:)=meas_time_temp(select_rows,:);
    end
end

one_run=1;%keep it 1 please
number_of_run=1;%keep it 1 please
mid_position_of_zenith=ceil((find(pos(:,1)~=pos(1,1),1)-1)/2)
other_position_of_zenith1=mid_position_of_zenith-ceil(mid_position_of_zenith/2);
other_position_of_zenith2=mid_position_of_zenith+ceil(mid_position_of_zenith/2);
which_zenith = pos(mid_position_of_zenith,2) %Uncomment and update the row number
if you are using one zenith from full data set
number_of_zenith=(find(pos(:,1)~=pos(1,1),1)-1);
err_mean=abs(mean(pos(2:number_of_zenith-1,2)-pos(3:number_of_zenith,2)))/5;
zenith_res=err_mean;
no_idea=1;
if fit_at_one_zenith==1 || no_idea==1
    azimuth_guess=200;   %update after runing with no_idea=1
    zenith_guess=30;   %update after runing with no_idea=1
    if no_idea==1
        DoLP_meas=pos(:,3);
        Azimuth_meas_full=pos(:,1);
        Zenith_meas_full=pos(:,2);
        azimuth_start=0;
        azimuth_end=360;
        zenith_start=0;
        zenith_end=90;
[sun_zen_az,goodness_of_fit]=DoLP_fit2(Azimuth_meas_full,Zenith_meas_full,DoLP_
meas,azimuth_guess,zenith_guess,azimuth_start,azimuth_end,zenith_start,zenith_end);
        sun_zen_az=coeffvalues(sun_zen_az);
        sun_azimuth_DoLP=sun_zen_az(1)
        sun_zenith_DoLP=sun_zen_az(2)
        azimuth_guess=sun_azimuth_DoLP
        zenith_guess=sun_zenith_DoLP;
        if under_water_mapping==1
            zenith_guess=asind(1.33*sind(sun_zenith_DoLP));
        end
    end
elseif fit_at_one_zenith==0 && no_idea==0
    azimuth_guess=from_fitting_at_one_zenith(global_count,5);   %Obtain the guess from
fitting at one zenith
    zenith_guess=from_fitting_at_one_zenith(global_count,6);    %Obtain the guess from
fitting at one zenith
    full_data_set_fitting=0;
end
```

```
if select_data_within_range==1
   AoP_error_limit=150;
end
updated_AoP_error_limt_by=5;
if no_idea==1 && multiple_zenith_angles==0 && select_data_within_range==0 &&
sampling_method1==0
   fit_at_one_zenith=1;
end
   GMT=-7;
   DLS=0; %Day Light Saving
   % %     Date
   year=meas_time(1,1);
   month=meas_time(1,2);
   day=meas_time(1,3);
   % %     Time of day
   hour=meas_time(:,4);
   minute=meas_time(:,5);
   second=meas_time(:,6);
   %%Actual location: 1. SDF Field 2. Old Main
   loc_collection_file=readmatrix('lat_lon_of_different_locations.txt');
   latitude_actual=loc_collection_file(which_location_of_measurement,2);
   longitude_actual=loc_collection_file(which_location_of_measurement,3);
tic
%%
if fit_at_one_zenith==1
   if zenith_data_is_ready==0
      theta_zen_p=which_zenith;
      k_k=find(abs(pos(:,2)-
theta_zen_p)<zenith_res);%find(abs(pos(:,2)==theta_zen_p));%
      az_meas_at_zen=pos(k_k,1);
      AoP_meas_at_zen=pos(k_k,4);
   else
      theta_zen_p=which_zenith;
      az_meas_at_zen=pos(:,1);
      AoP_meas_at_zen=pos(:,2);
      if remove_overexposed==1
         [AoP_meas_at_zen,outlierIndices,thresholdLow,thresholdHigh,center] = ...
         filloutliers(AoP_meas_at_zen,'center','movmedian',100,'ThresholdFactor',2.5);
      end
   end
elseif multiple_zenith_angles==1
[new_pos,mult_zenith,mult_meas_time,mult_mean_time,mult_dir_loc]=get_muliple_zen
ith_maping(folder_location{global_count},data_file_name,time_file_name);
   pos=new_pos;
   year=mult_meas_time(1,1);
   month=mult_meas_time(2,2);
```

```
    day=mult_meas_time(3,3);
    %Time of day
    hour=mult_meas_time(:,4);
    minute=mult_meas_time(:,5);
    second=mult_meas_time(:,6);
    mult_time=hour+(minute/60)+(second/60/60);
    pos(:,4)=mult_time;
    pos=sortrows(pos,4); %Rearrange w.r.t measurment time
    az_meas_at_zen=pos(:,1);
    zen_meas=pos(:,2);
    AoP_meas_at_zen=pos(:,3);
    mult_time=pos(:,4);
        if remove_overexposed==1
            [pos(:,3),outlierIndices,thresholdLow,thresholdHigh,center] = ...
            filloutliers(pos(:,3),'center','movmedian',100,'ThresholdFactor',2.5);
            AoP_meas_at_zen=pos(:,3);
        end
    which_zenith_mult=mult_zenith;
    mid_pos_mult_zen=size(which_zenith_mult,1)*size(which_zenith_mult,2)/2;
    which_zenith = pos(mid_pos_mult_zen,2) %Uncomment and update the row number if
you are using one zenith from full data set
else
    az_meas_at_zen=pos(:,1);
    AoP_meas_at_zen=pos(:,4);
    if remove_overexposed==1
        if select_data_within_range==1
            [pos(:,4),outlierIndices,thresholdLow,thresholdHigh,center] = ...
            filloutliers(pos(:,4),'center','movmedian',100,'ThresholdFactor',2.5);
        else
            [pos(:,4),outlierIndices,thresholdLow,thresholdHigh,center] = ...
            filloutliers(pos(:,4),'center','movmedian',4000,'ThresholdFactor',0.7);
        end
            ploar_contour_plot(pos(:,1),pos(:,2),pos(:,4),10);
    end
end
%%AoP fitting
if no_idea==1
    fiting_loop_run=6; % conducted a test : Mostly good fitting is found within 5th iteration.
else
    fiting_loop_run=1;
end
AoP_error_limit_after_fit_all=[]; %Set the acceptable aop error limit for the next fitting
%%
    search_in_range=20;
for try_c=1:length(AoP_error_limit_after_fit_all)+1
```

```
for i=1:fiting_loop_run
   if i > 1
      AoP_error_limit=100;
      azimuth_guess=sun_azimuth(i-1);
      zenith_guess=sun_zenith(i-1);
      if select_data_within_range==1
         AoP_error_limit=150;
      end
      no_idea=0;
      if i==2
         search_in_range=10;
      elseif i==3
         search_in_range=5;
      elseif i>3
         search_in_range=2.5/(i-3)^2; %/(i-2);
      end
   end
if fit_with_skylight_refraction_model==1
   %%%% to compare with PSD fitting %%%%%%%%
      which_zenith_pass_down=(zenith_low+zenith_high)/2; % fit for data within zenith
24 to 48
      res_pass_down=abs(which_zenith_pass_down-zenith_low);
      AZ_rng_start1=0;
      AZ_rng_end1=160;
      AZ_rng_start2=0;
      AZ_rng_end2=160;
      %%Zenith Range for fitting
      Zen_rng_start=which_zenith_pass_down-res_pass_down;
      Zen_rng_end=which_zenith_pass_down+res_pass_down;
      %%Measured data for fitting
      [pos_tem]=select_data_within_limit_greater_zen(pos,[],Zen_rng_start);      %Lower
Zenith angle
      [pos_tem]=select_data_within_limit_less_zen(pos_tem,[],Zen_rng_end);     %Upper
Zenith angle
      [pos1]=select_data_within_limit_greater_az(pos_tem,[],AZ_rng_start1);     %Lower
Zenith angle
      [pos1]=select_data_within_limit_less_az(pos1,[],AZ_rng_end1);   %Upper   Zenith
angle
      [pos2]=select_data_within_limit_greater_az(pos_tem,[],AZ_rng_start2);     %Lower
Zenith angle
      [pos2]=select_data_within_limit_less_az(pos2,[],AZ_rng_end2);   %Upper   Zenith
angle
      pos=[];
      pos=[pos1;pos2];
      pos=unique(pos,'rows');
```

```
        az_meas_at_zen=pos(:,1);
        AoP_meas_at_zen=pos(:,4);
fitting_conditions=[one_run,number_of_run,fit_at_one_zenith,is_it_reduced_data_point_
fitting,no_idea,azimuth_guess,zenith_guess,search_in_range,how_many_phi,how_many_
theta,AoP_error_limit,updated_AoP_error_limt_by,plot_it,multiple_zenith_angles,select
_data_within_range,sampling_method1,under_water_mapping];
    [sun_azimuth(i),sun_zenith(i),AoP_fit(:,i),AoP_measurment_fitting_std(i)]          =
fit_AoP_data(which_zenith,az_meas_at_zen,AoP_meas_at_zen,pos,fitting_conditions);
    sun_azimuth(i)
    sun_zenith(i)
    standard_deviation_measurment_vs_fitting=AoP_measurment_fitting_std(i)
elseif fit_with_PSD_model==1
    sun_pos_step=0.5;
    sun_ZEN_pass_down=zenith_guess-
search_in_range/2:sun_pos_step:zenith_guess+search_in_range/2;
    sun_AZ_pass_down=azimuth_guess-
search_in_range/2:sun_pos_step:azimuth_guess+search_in_range/2;
    off_set_pass_down=[-15.05]; %different for different measurment
    which_zenith_pass_down=(zenith_low+zenith_high)/2; % fit for data within zenith 24
to 48
    res_pass_down=abs(which_zenith_pass_down-zenith_low);
    AZ_rng_start1_pass_down=0;
    AZ_rng_end1_pass_down=150;
    AZ_rng_start2_pass_down=0;
    AZ_rng_end2_pass_down=150;
    MFP_pass_down=0.3695; %in m
    attenuation_coefficient_pass_down=2.7126; % unit m^-1 % obtained from 03-03-22 lab
measurement
    vertical_measurement_depth_pass_down=11;  % in inch %different for different
measurement
    % 2021: Oct 11: 11; inch Oct 19: 6 inch; Dec 20: 10 inch
    depth_pass_down=0.7;
    sky_scat_ratio_pass_down=1; % different for different measuremnt
fitting_conditions=[zenith_guess,azimuth_guess,off_set_pass_down,which_zenith_pass_
down,res_pass_down,AZ_rng_start1_pass_down,AZ_rng_end1_pass_down,AZ_rng_star
t2_pass_down,AZ_rng_end2_pass_down,MFP_pass_down,attenuation_coefficient_pass_
down,vertical_measurement_depth_pass_down,depth_pass_down,sky_scat_ratio_pass_d
own,search_in_range,sun_pos_step];
[sun_azimuth(i),sun_zenith(i),AoP_fit(:,i),AoP_measurment_fitting_std(i),pos_meas_use
d_in_fit] = fit_AoP_data_PSD_model(pos,fitting_conditions);
    sun_azimuth(i)
    sun_zenith(i)
    standard_deviation_measurment_vs_fitting=AoP_measurment_fitting_std(i)
    pos=[];
    pos=pos_meas_used_in_fit;
```

```
    az_meas_at_zen=pos(:,1);
    AoP_meas_at_zen=pos(:,4);
end
end
min_pos_std_AoP_fit=find(AoP_measurment_fitting_std==min(AoP_measurment_fittin
g_std),1);
    min_pos_1st(global_count)=min_pos_std_AoP_fit
    sun_azimuth=sun_azimuth(min_pos_std_AoP_fit);
    sun_zenith=sun_zenith(min_pos_std_AoP_fit);
standard_deviation_measurment_vs_fitting=AoP_measurment_fitting_std(min_pos_std_
AoP_fit);
    disp('%%%%%%%%%%%%%%')
    disp(strcat('End of fitting-', num2str(try_c),'!'))
    disp(strcat('Sun Azimuth:',num2str(sun_azimuth)))
    disp(strcat('Sun Zenith:',num2str(sun_zenith)))
    disp(strcat('Std:',num2str(standard_deviation_measurment_vs_fitting)))
    disp('%%%%%%%%%%%%%%')
    if try_c<=length(AoP_error_limit_after_fit_all)
        del_AoP_meas_pos=pos(:,4)-AoP_fit(:,min_pos_std_AoP_fit);
        %%
            figure
            scatter(pos(:,1),del_AoP_meas_pos)
            xlabel('Azimuth')
            ylabel('AoP error after fiting')
            set(gca, 'FontSize', 25);
            set(gcf,'color','w');
        %%
        AoP_error_limit_after_fit=AoP_error_limit_after_fit_all(try_c)
allowed_AoP_Error_data_position=find((abs(del_AoP_meas_pos)<AoP_error_limit_afte
r_fit)==1);
        %%
        figure
scatter(pos(allowed_AoP_Error_data_position,1),del_AoP_meas_pos(allowed_AoP_Erro
r_data_position))
        xlabel('Azimuth')
        ylabel('AoP error after fiting')
        set(gca, 'FontSize', 25);
        set(gcf,'color','w');
        allowed_AoP_Error_data=del_AoP_meas_pos(allowed_AoP_Error_data_position);
        pos=pos(allowed_AoP_Error_data_position,:);
        ploar_contour_plot(pos(:,1),pos(:,2),pos(:,4),10)
        figure
        scatter(pos(:,1),pos(:,4))
        xlabel('Azimuth')
        ylabel('AoP')
        legend(strcat('|AoP Error limit|-',num2str(AoP_error_limit_after_fit)))
```

```
        set(gca, 'FontSize', 25);
        set(gcf,'color','w');

        clear sun_azimuth
        clear sun_zenith
        clear AoP_fit
        clear AoP_measurment_fitting_std
    end
    if try_c==length(AoP_error_limit_after_fit_all)+1
        disp('%%%%%%%%%%%%%%')
        disp('End of fitting!')
        disp(strcat('Sun Azimuth:',num2str(sun_azimuth)))
        disp(strcat('Sun Zenith:',num2str(sun_zenith)))
        disp(strcat('Std:',num2str(standard_deviation_measurment_vs_fitting)))
        disp('%%%%%%%%%%%%%%')
    end
end
%%
if full_data_set_fitting==1
    fit_at_one_zenith=0;
    az_meas_at_zen=pos(:,1);
    AoP_meas_at_zen=pos(:,4);
    azimuth_guess=sun_azimuth;
    zenith_guess=sun_zenith;
    search_in_range=1;
    how_many_phi=50;
    how_many_theta=20;
fitting_conditions=[one_run,number_of_run,fit_at_one_zenith,is_it_reduced_data_point_
fitting,no_idea,azimuth_guess,zenith_guess,search_in_range,how_many_phi,how_many_
theta,AoP_error_limit,updated_AoP_error_limt_by,plot_it,multiple_zenith_angles,select
_data_within_range,sampling_method1,under_water_mapping];
    [sun_azimuth,sun_zenith,AoP_fit,AoP_measurment_fitting_std]                    =
fit_AoP_data(which_zenith,az_meas_at_zen,AoP_meas_at_zen,pos,fitting_conditions);
    sun_azimuth
    sun_zenith
    standard_deviation_measurment_vs_fitting=AoP_measurment_fitting_std
end
if plot_it==1
    if fit_at_one_zenith==1 || multiple_zenith_angles==1
        AoP_meas=AoP_meas_at_zen;
        Azimuth_meas=az_meas_at_zen;
    else
        AoP_meas=pos(:,4);
        Azimuth_meas=pos(:,1);
    end
```

```
    del_AoP_meas_pos=AoP_meas-AoP_fit(:,min_pos_std_AoP_fit);
allowed_AoP_Error_data_position=find((abs(del_AoP_meas_pos)<AoP_error_limit)==1
);
    allowed_AoP_Error_data=del_AoP_meas_pos(allowed_AoP_Error_data_position);
    azimuth_at_allowed_AoP_data=Azimuth_meas(allowed_AoP_Error_data_position);

    figure
    scatter(Azimuth_meas,(del_AoP_meas_pos))
    hold on
    plot(azimuth_at_allowed_AoP_data,(allowed_AoP_Error_data))
    xlabel('Azimuth')
    ylabel('AoP Error')
    legend({'Over all error','Used for calculating STD'},'Location','best')
    set(gca, 'FontSize', 14);
    set(gcf,'color','w');
    if fit_at_one_zenith==1
        title(strcat(figure_title,' :  ',['Zenith=',num2str(which_zenith)]))
        fig_name=strcat('\AoP Error-',['Zenith=',num2str(which_zenith)]); %Can't use ':' in
file name
    elseif multiple_zenith_angles==1
        title(strcat(figure_title,'    :       ',['Zenith=',num2str(which_zenith_mult(1))],'    to    ',['
',num2str(which_zenith_mult(end))]))
        fig_name=strcat('\AoP                Error-Multi                Zenith                fitting-
',['Zenith=',num2str(which_zenith_mult(1))],'  to',['  ',num2str(which_zenith_mult(end))]);
%Can't use ':' in  file name
    elseif select_data_within_range==1
        title(strcat(figure_title,'-Az                        cover=',num2str(az_cover),'\circ','-Zen
cover=',num2str(zen_cover),'\circ'))
        fig_name='\AoP Error-selected data';
    elseif sampling_method1==1
        if sample_randomly==1
            title(strcat(figure_title,'-Sampled Data'))
            fig_name=strcat('\AoP Error-sampled data','-',zen_range);
        else
            title(strcat(figure_title,'-Sampled Data'))
            fig_name='\AoP Error-sampled data';
        end
    else
        title(strcat(figure_title))
        fig_name='\AoP Error-Full Dataset';
    end
    if azimuth_offset_correction==1
        fig_name=strcat(fig_name,'-Azimuth offset corrected');
    end
    if remove_overexposed==1
        fig_name=strcat(fig_name,'-overexpose removed');
```

```
    end
    fig_name=strcat(fig_name,'.png');
    fig_location=file_location;
    if multiple_zenith_angles==1
        fig_location=mult_dir_loc;
    end
    if save_figure==1
        if multiple_zenith_angles==1
            saveas(gcf,cell2mat(strcat(fig_location,fig_name)))
        else
            saveas(gcf,(strcat(fig_location,fig_name)))
        end
    end


    if full_data_set_fitting==1
        AoP_sim_at_zen=AoP_fit;
    else
        AoP_sim_at_zen=AoP_fit(:,min_pos_std_AoP_fit);
    end

    figure
    s=scatter(Azimuth_meas,AoP_meas,30,'filled');
    hold on
    scatter(Azimuth_meas,AoP_sim_at_zen,'LineWidth',2);
    xlabel('Azimuth')
    ylabel('AoP')
    legend({'measuremnt','fitting'},'Location','northwest')
    set(gca, 'FontSize', 14);
    set(gcf,'color','w');
    s.MarkerFaceAlpha = 0.1;
    if fit_at_one_zenith==1
        title(strcat(figure_title,' :  ',['Zenith=',num2str(which_zenith)]));
        fig_name=strcat('\AoP measurment and fitting Overlap-
',['Zenith=',num2str(which_zenith)]);
    elseif multiple_zenith_angles==1
        title(strcat(figure_title,' :  ',['Zenith=',num2str(which_zenith_mult(1))],' to ',['
',num2str(which_zenith_mult(end))],['(',num2str(which_zenith),')']));
        fig_name=strcat('\AoP measurment and multi-Zenith fitting Overlap-
',['Zenith=',num2str(which_zenith_mult(1))],' to ',[' ',num2str(which_zenith_mult(end))]);
    elseif select_data_within_range==1
        title(strcat(figure_title,'-Az cover=',num2str(az_cover),'\circ','-Zen
cover=',num2str(zen_cover),'\circ'))
        fig_name='\AoP measurment and fitting Overlap-selected data';
    elseif sampling_method1==1
```

```
        if sample_randomly==1
            title(strcat(figure_title,'-Sampled Data'))
            fig_name=strcat('\AoP measurment and fitting Overlap-sampled data','-
',zen_range);
        else
            title(strcat(figure_title,'-Sampled Data'))
            fig_name='\AoP measurment and fitting Overlap-sampled data';
        end
    else
        title(strcat(figure_title,' :  ',['Zenith=',num2str(which_zenith)]));
        fig_name='\AoP measurment and fitting Overlap-FUll Dataset';
    end
    if azimuth_offset_correction==1
        fig_name=strcat(fig_name,'-Azimuth offset corrected');
    end
    if remove_overexposed==1
        fig_name=strcat(fig_name,'-overexpose removed');
    end
    fig_name=strcat(fig_name,'.png');
    fig_location=file_location;
    if multiple_zenith_angles==1
        fig_location=mult_dir_loc;
    end
    if save_figure==1
        if multiple_zenith_angles==1
            saveas(gcf,cell2mat(strcat(fig_location,fig_name)))
        else
            saveas(gcf,(strcat(fig_location,fig_name)))
        end
    end
end

time_taken_to_fit_AoP=toc;
%%
    time=mean(hour+(minute/60)+(second/60/60)); % in hour (11:30am is 11.5hr)
    if multiple_zenith_angles==1
        time=mean(mult_mean_time);
    end
    time_of_measurment=time;
    [actual_zenith,actual_Azimuth]                                                    =
sun_position(month,day,year,time,latitude_actual,longitude_actual,GMT,DLS);      %time
dependent sun position
AoP_fitting_information(global_count,:)=[measurment_number,time_of_measurment,act
ual_Azimuth,actual_zenith,sun_azimuth,sun_zenith,standard_deviation_measurment_vs_
fitting,time_taken_to_fit_AoP];
if save_individual_fitting==1
```

```
    if fit_at_one_zenith==1
        file_name=strcat('\AoP fitting Information','-',['Zenith=',num2str(which_zenith)]);
    elseif multiple_zenith_angles==1
        file_name=strcat('\AoP Multi Zenith fitting Information','-
',['Zenith=',num2str(which_zenith_mult(1))],' to ',[' ',num2str(which_zenith_mult(end))]);
        file_location=mult_dir_loc;
    elseif select_data_within_range==1
        file_name=strcat('\AoP Selected Data fitting Information','-','Az
cover=',num2str(az_cover),'-Zen cover=',num2str(zen_cover));
    elseif sampling_method1==1
        file_name=strcat('\AoP Sampled Data fitting Information-',zen_range);
    else
        file_name=strcat('\AoP fitting Information-Full Dataset');
    end
    if azimuth_offset_correction==1
        file_name=strcat(file_name,'-Azimuth offset corrected');
    end
    if remove_overexposed==1
        file_name=strcat(file_name,'-overexpose removed');
    end
    file_name=strcat(file_name,'.txt');
    if multiple_zenith_angles==1
        f_name=cell2mat(strcat(file_location,file_name));
    else
        f_name=(strcat(file_location,file_name));
    end
    fid = fopen(f_name, 'w');
    fprintf(fid,
'measurment_number,time_of_measurment,Actual_Azimuth,Actual_zenith,fitted_azimut
h,fitted_zenith,standard_deviation_measurment_vs_fitting,time_taken_to_fit_AoP\n');
    AoP_fitting_information_current=AoP_fitting_information(global_count,:);
    save(f_name,'AoP_fitting_information_current','-ascii','-append')
    fclose(fid);
        %save fitted data
    f_name=(strcat(file_location,'\AoP_meas and fitted data-',zen_range,'.txt'));
    AoP_meas_fitting_data_save=[Azimuth_meas pos(:,2) AoP_meas_at_zen
AoP_sim_at_zen];
    save(f_name,'AoP_meas_fitting_data_save','-ascii','-append')
end
%%
if find_location==1
    time=mean(hour+(minute/60)+(second/60/60)); % in hour (11:30am is 11.5hr)
    if multiple_zenith_angles==1
        time=mean(mult_mean_time);
    end
```

```
    sun_az=sun_azimuth;
    sun_zen=sun_zenith;
    [latitude_predict, longitude_predict]=
location_predict(month,day,year,time,GMT,DLS,sun_az,sun_zen,number_of_iteration_f
or_location_prediction,no_idea_of_location);
    latitude_predict;
    longitude_predict;
location_prediction_error=deg2km(distance(latitude_actual,longitude_actual,latitude_pre
dict,longitude_predict));
    time_taken_to_determine_location=toc;
location_determination_information(global_count,:)=[measurment_number,time_of_mea
surment,latitude_actual,longitude_actual,latitude_predict,longitude_predict,location_pred
iction_error,time_taken_to_determine_location];
    if save_individual_fitting==1
        if fit_at_one_zenith==1
            file_name=strcat('\Location Determination Information','-
',['Zenith=',num2str(which_zenith)]);
        elseif multiple_zenith_angles==1
            file_name=(strcat('\Location Determination Information-Multi Zenith','-
',['Zenith=',num2str(which_zenith_mult(1))],' to ',[' ',num2str(which_zenith_mult(end))]));
            file_location=mult_dir_loc;
        elseif select_data_within_range==1
            file_name=strcat('\Location Determination Information','-','Az
cover=',num2str(az_cover),'-Zen cover=',num2str(zen_cover));
        elseif sampling_method1==1
            file_name=strcat('\Location Determination Information','-','Sampled Dataset');
        else
            file_name=strcat('\Location Determination Information-Full Dataset');
        end
        if azimuth_offset_correction==1
            file_name=strcat(file_name,'-Azimuth offset corrected');
        end
        if remove_overexposed==1
            file_name=strcat(file_name,'-overexpose removed');
        end
        file_name=strcat(file_name,'.txt');
        if multiple_zenith_angles==1
            f_name=cell2mat(strcat(file_location,file_name));
        else
            f_name=(strcat(file_location,file_name));
        end
        fid = fopen(f_name, 'w');
        fprintf(fid,
'measurment_number,time_of_measurment,latitude_actual,longitude_actual,latitude_pred
ict,longitude_predict,location_prediction_error,time_taken_to_determine_location\n');
```

```
location_determination_information_current=location_determination_information(global
_count,:);
        save(f_name,'location_determination_information_current','-ascii','-append')
        fclose(fid);
    end
    if time_dependent==1
        time=(hour+(minute/60)+(second/60/60));
        if multiple_zenith_angles==1
            time=mult_time;
        end
        %In video mode, sometimes several number of images may have
        %the same time. We need to consider those images as one signle image
        k1=1;
        step=0;
        temp_t=0;
        for i=1:length(time)
            if isempty(find(time(temp_t+1:end,1)~=time(temp_t+1,1),1)-1)~=1
            step(k1)=(find(time(temp_t+1:end,1)~=time(temp_t+1,1),1)-1)+temp_t;
            else
            step(k1)=temp_t+1;
            break;
            end
        temp_t=step(k1);
        k1=k1+1;
        end
        number_of_images=length(step); %consider only these 'step' in calculating the
number of images
        time=time(step);
        [actual_zenith_timed,actual_Azimuth_timed]                                    =
sun_position(month,day,year,time,latitude_actual,longitude_actual,GMT,DLS);     %time
dependent sun position
        [theta_timed,phi_timed]                                                       =
sun_position(month,day,year,time,latitude_predict,longitude_predict,GMT,DLS);  %time
dependent sun position
        timed_range=2;
        how_many_timed_theta_phi=20;
        theta_timed_min=theta_timed-timed_range/2;
        phi_timed_min=phi_timed-timed_range/2;
        for i=1:how_many_timed_theta_phi+1
theta_timed_table(:,i)=theta_timed_min+(timed_range)/how_many_timed_theta_phi*(i-
1);
phi_timed_table(:,i)=phi_timed_min+(timed_range)/how_many_timed_theta_phi*(i-1);
        end
        for i=1:how_many_timed_theta_phi+1
            theta=theta_timed_table(:,i);
            phi=phi_timed_table(:,i);
```

```
        if fit_at_one_zenith==1
            measurement_data=[az_meas_at_zen,AoP_meas_at_zen];
            AoP_meas=AoP_meas_at_zen;
            Azimuth_meas=az_meas_at_zen;
        elseif multiple_zenith_angles==1
            measurement_data=pos;
            AoP_meas=pos(:,3);
            Azimuth_meas=pos(:,1);
        else
            measurement_data=pos;
            AoP_meas=pos(:,4);
            Azimuth_meas=pos(:,1);
        end
    [fitting_data,AoP_fit_time(:,i)]=
time_dependent_fitting(theta,phi,measurement_data,multiple_zenith_angles,fit_at_one_z
enith,which_zenith,number_of_images,under_water_mapping);
    del_AoP_meas_timed_fit=AoP_meas-AoP_fit_time(:,i);
allowed_AoP_Error_data_position_timed=find((abs(del_AoP_meas_timed_fit)<AoP_err
or_limit)==1);
allowed_AoP_Error_data_timed=del_AoP_meas_timed_fit(allowed_AoP_Error_data_po
sition_timed);
meas_pos_std_timed(:,i)=sqrt(sum((abs(allowed_AoP_Error_data_timed)).^2,'omitnan')./
length(del_AoP_meas_timed_fit));
    end
    min_position=find(meas_pos_std_timed==min(meas_pos_std_timed));
    sun_azimuth_timed=(phi_timed_table(:,min_position));
    sun_zenith_timed=(theta_timed_table(:,min_position));
standard_deviation_measurment_vs_time_fitting=meas_pos_std_timed(:,min_position);
    if plot_it==1
        del_AoP_meas_timed_fit=AoP_meas-AoP_fit_time(:,min_position);
allowed_AoP_Error_data_position_timed=find((abs(del_AoP_meas_timed_fit)<AoP_err
or_limit)==1);
allowed_AoP_Error_data_timed=del_AoP_meas_timed_fit(allowed_AoP_Error_data_po
sition_timed);
        allowed_AoP_data_timed=AoP_meas(allowed_AoP_Error_data_position_timed);
azimuth_at_allowed_AoP_data_timed=Azimuth_meas(allowed_AoP_Error_data_positio
n_timed);
        figure
        scatter(measurement_data(:,1),del_AoP_meas_timed_fit)
        hold on
        plot(azimuth_at_allowed_AoP_data_timed,(allowed_AoP_Error_data_timed))
        xlabel('Azimuth')
        ylabel('AoP Error')
        legend({'Over all error','Used for calculating STD'},'Location','best')
        set(gca, 'FontSize', 14);
```

```
        set(gcf,'color','w');
        if fit_at_one_zenith==1
            title(strcat(figure_title,'      :            ',['Zenith=',num2str(which_zenith)],'-Time
Dependent'))
            fig_name=strcat('\AoP                    Error-Time                    Dependent-
',['Zenith=',num2str(which_zenith)]);
        elseif multiple_zenith_angles==1
            title(strcat(figure_title,' :    ',['Zenith=',num2str(which_zenith_mult(1))],' to ',['
',num2str(which_zenith_mult(end))]))
            fig_name=strcat('\AoP    Error-Time    Dependent-Multi    Zenith    fitting-
',['Zenith=',num2str(which_zenith_mult(1))],' to',['  ',num2str(which_zenith_mult(end))]);
%Can't use ':' in  file name
        elseif select_data_within_range==1
            title(strcat(figure_title,'-Az                    cover=',num2str(az_cover),'\circ','-Ze
cover=',num2str(zen_cover),'\circ'))
            fig_name='\AoP Error-selected data';
        elseif sampling_method1==1
            title(strcat(figure_title,'-Sampled Data'))
            fig_name='\AoP Error-Sampled data';
        else
            title(strcat(figure_title,' :  ','-Time Dependent'))
            fig_name='\AoP Error-Time Dependent-Full Dataset';
        end
        if azimuth_offset_correction==1
            fig_name=strcat(fig_name,'-Azimuth offset corrected');
        end
        if remove_overexposed==1
            fig_name=strcat(fig_name,'-overexpose removed');
        end
        fig_name=strcat(fig_name,'.png');
        fig_location=file_location;
        if save_figure==1
            if multiple_zenith_angles==1
                saveas(gcf,cell2mat(strcat(fig_location,fig_name)))
            else
                saveas(gcf,strcat(fig_location,fig_name))
            end
        end

        if fit_at_one_zenith==1
            AoP_timed_fit_at_zen=AoP_fit_time(:,min_position);
        else
            theta_zen_p=which_zenith;
            k_k=find(abs(pos(:,2)-theta_zen_p)<zenith_res);
            az_meas_at_zen=pos(k_k,1);
            AoP_timed_fit_at_zen=AoP_fit_time(k_k,min_position);
```

```
            if multiple_zenith_angles==1
                AoP_meas_at_zen=pos(k_k,3);
            else
                AoP_meas_at_zen=pos(k_k,4);
            end
        end

        figure
        s=scatter(az_meas_at_zen,AoP_meas_at_zen,30,'filled');
        hold on
            plot(az_meas_at_zen,AoP_timed_fit_at_zen,'LineWidth',2);
        xlabel('Azimuth')
        ylabel('AoP')
        legend({'measuremnt','Timed fitting'},'Location','northwest')
        set(gca, 'FontSize', 14);
        set(gcf,'color','w');
        s.MarkerFaceAlpha = 0.1;
        if fit_at_one_zenith==1
            title(strcat(figure_title,'      :            ',['Zenith=',num2str(which_zenith)],'-Time
Dependent'));
            fig_name=strcat('\AoP measurment and fitting Overlap-Time Dependent-
',['Zenith=',num2str(which_zenith)]);
        elseif multiple_zenith_angles==1
            title(strcat(figure_title,' :   ',['Zenith=',num2str(which_zenith_mult(1))],' to ',['
',num2str(which_zenith_mult(end))],['(',num2str(which_zenith),')']));
            fig_name=strcat('\AoP measurment and multi-Zenith fitting Overlap-Time
Dependent-          ',['Zenith=',num2str(which_zenith_mult(1))],'          to          ',['
',num2str(which_zenith_mult(end))]);
        elseif select_data_within_range==1
            title(strcat(figure_title,'-Az                cover=',num2str(az_cover),'\circ','-Zen
cover=',num2str(zen_cover),'\circ'))
            fig_name='\AoP measurment and fitting Overlap-selected data';
        elseif sampling_method1==1
            title(strcat(figure_title,'-Sampled Data'))
            fig_name='\AoP measurment and fitting Overlap-Sampled Data';
        else
            title(strcat(figure_title,'      :            ',['Zenith=',num2str(which_zenith)],'-Time
Dependent'));
            fig_name='\AoP measurment and fitting Overlap-Time Dependent-Full Dataset';
        end
        if azimuth_offset_correction==1
            fig_name=strcat(fig_name,'-Azimuth offset corrected');
        end
        if remove_overexposed==1
            fig_name=strcat(fig_name,'-overexpose removed');
        end
```

```
            fig_name=strcat(fig_name,'.png');
            fig_location=file_location;
            if save_figure==1
                if multiple_zenith_angles==1
                    saveas(gcf,cell2mat(strcat(fig_location,fig_name)))
                else
                    saveas(gcf,strcat(fig_location,fig_name))
                end
            end
        end
        time_taken_to_fit_AoP_timed=toc;
        measurment_number_timed(1:length(sun_zenith_timed),1)=measurment_number;
standard_deviation_measurment_vs_time_fitting(1:length(sun_zenith_timed),1)=standar
d_deviation_measurment_vs_time_fitting;
time_taken_to_fit_AoP_timed(1:length(sun_zenith_timed),1)=time_taken_to_fit_AoP_ti
med;
        time_of_measurment_timed=time;
AoP_fitting_information_timed{global_count,:}=[measurment_number_timed,time_of_
measurment_timed,actual_Azimuth_timed',actual_zenith_timed,sun_azimuth_timed,sun_
zenith_timed,standard_deviation_measurment_vs_time_fitting,time_taken_to_fit_AoP_ti
med];
        if save_individual_fitting==1
            if fit_at_one_zenith==1
                file_name=strcat('\AoP fitting Information-Time Dependent','-
',['Zenith=',num2str(which_zenith)]);
            elseif multiple_zenith_angles==1
                file_name=strcat('\AoP Multi Zenith fitting Information-Time Dependent','-
',['Zenith=',num2str(which_zenith_mult(1))],' to ',[' ',num2str(which_zenith_mult(end))]);
                file_location=mult_dir_loc;
            elseif select_data_within_range==1
                file_name=strcat('\AoP fitting Information-Selected Data-Time Dependent','-
','Az cover=',num2str(az_cover),'-Zen cover=',num2str(zen_cover));
            elseif sampling_method1==1
                file_name=strcat('\AoP fitting Information-Sampled Data-Time Dependent');
            else
                file_name=strcat('\AoP fitting Information-Time Dependent-Full Dataset');
            end
            if azimuth_offset_correction==1
                file_name=strcat(file_name,'-Azimuth offset corrected');
            end
            if remove_overexposed==1
                file_name=strcat(file_name,'-overexpose removed');
            end
            file_name=strcat(file_name,'.txt');
            if multiple_zenith_angles==1
                f_name=cell2mat(strcat(file_location,file_name));
```

```
        else
            f_name=(strcat(file_location,file_name));
        end
        fid = fopen(f_name, 'w');
        fprintf(fid,
'measurment_number,time_of_measurment,actual_Azimuth,actual_zenith,fitted_azimuth
_timed,fitted_zenith_timed,standard_deviation_measurment_vs_time_fitting,time_taken_
to_fit_AoP_timed\n');

AoP_fitting_information_timed_current=AoP_fitting_information_timed{global_count,:
};
        save(f_name,'AoP_fitting_information_timed_current','-ascii','-append')
        fclose(fid);
    end
    time_select=randi([1 length(time)],1,5);
    time_new=time(time_select);
    sun_az=sun_azimuth_timed(time_select);
    sun_zen=sun_zenith_timed(time_select);
    latitude_predict_new(1:length(time_new),1)=zeros(length(time_new),1);
    longitude_predict_new(1:length(time_new),1)=zeros(length(time_new),1);
    location_prediction_error_new(1:length(time_new),1)=zeros(length(time_new),1);
    for loc_timed_count=1:length(time_new)
        [latitude_predict_new(loc_timed_count,1),
longitude_predict_new(loc_timed_count,1)]=
location_predict(month,day,year,time_new(loc_timed_count),GMT,DLS,sun_az(loc_tim
ed_count),sun_zen(loc_timed_count),number_of_iteration_for_location_prediction,no_id
ea_of_location);
location_prediction_error_new(loc_timed_count,1)=deg2km(distance(latitude_actual,lon
gitude_actual,latitude_predict_new(loc_timed_count,1),longitude_predict_new(loc_time
d_count,1)));
    end
    time_taken_to_determine_location_timed=toc;
    latitude_actual_timed(1:length(time_new),1)=latitude_actual;
    longitude_actual_timed(1:length(time_new),1)=longitude_actual;
time_taken_to_determine_location_timed(1:length(time_new),1)=time_taken_to_determi
ne_location_timed;
location_determination_information_timed{global_count,:}=[measurment_number_time
d(time_select),time_new,latitude_actual_timed,longitude_actual_timed,latitude_predict_
new,longitude_predict_new,location_prediction_error_new,time_taken_to_determine_loc
ation_timed];
    if save_individual_fitting==1
        if fit_at_one_zenith==1
            file_name=strcat('\Location Determination Information-Time Dependent','-
',['Zenith=',num2str(which_zenith)]);
        elseif multiple_zenith_angles==1
```

```
            file_name=(strcat('\Location Determination Information-Multi Zenith-Time
Dependent','-',['Zenith=',num2str(which_zenith_mult(1))],' to ',['
',num2str(which_zenith_mult(end))]));
            file_location=mult_dir_loc;
        elseif select_data_within_range==1
            file_name=strcat('\Location Determination Information-Selected Data-Time
Dependent','-','Az cover=',num2str(az_cover),'-Zen cover=',num2str(zen_cover));
        elseif sampling_method1==1
            file_name=strcat('\Location Determination Information-Sampled Data-Time
Dependent');
        else
            file_name=strcat('\Location Determination Information-Time Dependent-Full
Dataset');
        end
        if azimuth_offset_correction==1
            file_name=strcat(file_name,'-Azimuth offset corrected');
        end
        if remove_overexposed==1
            file_name=strcat(file_name,'-overexpose removed');
        end
        file_name=strcat(file_name,'.txt');
        if multiple_zenith_angles==1
            f_name=cell2mat(strcat(file_location,file_name));
        else
            f_name=(strcat(file_location,file_name));
        end
        fid = fopen(f_name, 'w');
        fprintf(fid,
'measurment_number,time_of_measurment,latitude_actual,longitude_actual,latitude_pred
ict_new,longitude_predict_new,location_prediction_error_new,time_taken_to_determine
_location_timed\n');
location_determination_information_timed_current=location_determination_information
_timed{global_count,:};
        save(f_name,'location_determination_information_timed_current','-ascii','-
append')
        fclose(fid);
      end
   end
end
if time_dependent==0
   time_taken_to_fit_AoP_timed=0;
   time_taken_to_determine_location_timed=0;
end

if find_location==0
   time_taken_to_determine_location=0;
```

```
end
overall_time_to_finish=toc;
time_required(global_count,:)=[measurment_number,time_taken_to_fit_AoP,time_taken
_to_determine_location,time_taken_to_fit_AoP_timed(1),time_taken_to_determine_locat
ion_timed(1),overall_time_to_finish];
if save_individual_fitting==1
   if fit_at_one_zenith==1
      file_name=strcat('\Time Required to finish','-',['Zenith=',num2str(which_zenith)]);
   elseif multiple_zenith_angles==1
      file_name=(strcat('\Time Required to finish-Multi Zenith','-
',['Zenith=',num2str(which_zenith_mult(1))],' to ',[' ',num2str(which_zenith_mult(end))]));
      file_location=mult_dir_loc;
   elseif select_data_within_range==1
      file_name=strcat('\Time Required to finish-Selected Data','-','Az
cover=',num2str(az_cover),'-Zen cover=',num2str(zen_cover));
   elseif sampling_method1==1
      file_name=strcat('\Time Required to finish-Sampled Data');
   else
      file_name=strcat('\Time Required to finish-Full Dataset');
   end
   if azimuth_offset_correction==1
      file_name=strcat(file_name,'-Azimuth offset corrected');
   end
   if remove_overexposed==1
      file_name=strcat(file_name,'-overexpose removed');
   end
   file_name=strcat(file_name,'.txt');
   if multiple_zenith_angles==1
      f_name=cell2mat(strcat(file_location,file_name));
   else
      f_name=(strcat(file_location,file_name));
   end
   fid = fopen(f_name, 'w');
   fprintf(fid,
'measurment_number,time_taken_to_fit_AoP,time_taken_to_determine_location,time_ta
ken_to_fit_AoP_timed,time_taken_to_determine_location_timed,overall_time_to_finish\
n');
   time_required_current=time_required(global_count,:);
   save(f_name,'time_required_current','-ascii','-append')
   fclose(fid);
end
   clear theta_timed_table;
   clear phi_timed_table;
   clear AoP_fit;
   clear AoP_fit_time;
   clear measurment_number_timed;
```

```
    clear standard_deviation_measurment_vs_time_fitting;
    clear time_taken_to_fit_AoP_timed;
    clear time_of_measurment_timed;
    clear pos_meas
    clear pos
    clear DoLP_meas
    clear AoP_meas
    clear az_meas_at_zen
    clear zen_meas_at_zen
    clear AoP_meas_at_zen
    clear DoLP_meas_at_zen
    clear Q_meas_at_zen
    clear U_meas_at_zen
end
%%
AoP_fitting_information
if find_location==1
    location_determination_information
end
if time_dependent==1
    AoP_fitting_information_timed
    location_determination_information_timed
end
time_required
if save_overall_fitting==1
    if save_overall_fitting_in_new_dir==1
        mkdir(main_folder_location,new_directory_name);
        main_folder_location_new=strcat(main_folder_location,'\',new_directory_name);
    end
    %save fitted data
    f_name=(strcat(main_folder_location_new,'\AoP_meas and fitted data-
',zen_range,'.txt'));
    AoP_meas_fitting_data_save=[Azimuth_meas pos(:,2) AoP_meas_at_zen
AoP_sim_at_zen];
    save(f_name,'AoP_meas_fitting_data_save','-ascii','-append')
    if fit_at_one_zenith==1
        file_name='\All AoP fitting information';
    elseif multiple_zenith_angles==1
        file_name=(strcat('\All AoP fitting information-Multi Zenith','-
',['Zenith=',num2str(which_zenith_mult(1))],' to ',[' ',num2str(which_zenith_mult(end))]));
    elseif select_data_within_range==1
        file_name=strcat('\All AoP fitting information-Selected Data','-','Az
cover=',num2str(az_cover),'-Zen cover=',num2str(zen_cover));
    elseif sampling_method1==1
        file_name=strcat('\All AoP fitting information-Sampled Data');
    else
```

```
        file_name='\All AoP fitting information-Full Dataset';
    end
    if azimuth_offset_correction==1
        file_name=strcat(file_name,'-Azimuth offset corrected');
    end
    if remove_overexposed==1
        file_name=strcat(file_name,'-Overexpose Removed');
    end
    file_name=strcat(file_name,'.txt');
    fid = fopen((strcat(main_folder_location_new,file_name)), 'w');
    fprintf(fid,
'measurment_number,time_of_measurment,Actual_Azimuth,Actual_zenith,fitted_azimut
h,fitted_zenith,standard_deviation_measurment_vs_fitting,time_taken_to_fit_AoP\n');
    save((strcat(main_folder_location_new,file_name)),'AoP_fitting_information','-ascii','-
append')
    fclose(fid);
    if find_location==1
        if fit_at_one_zenith==1
            file_name=('\All Location Determination Information');
        elseif multiple_zenith_angles==1
            file_name=(strcat('\All Location Determination Information-Multi Zenith','-
',['Zenith=',num2str(which_zenith_mult(1))],' to ',[' ',num2str(which_zenith_mult(end))]));
        elseif select_data_within_range==1
            file_name=strcat('\All Location Determination Information-Selected Data','-','Az
cover=',num2str(az_cover), '-Zen cover=',num2str(zen_cover));
        elseif sampling_method1==1
            file_name=strcat('\All Location Determination Information-Sampled Data');
        else
            file_name=('\All Location Determination Information-Full Dataset');
        end
        if azimuth_offset_correction==1
            file_name=strcat(file_name,'-Azimuth offset corrected');
        end
        if remove_overexposed==1
            file_name=strcat(file_name,'-Overexpose Removed');
        end
        file_name=strcat(file_name,'.txt');
        fid = fopen((strcat(main_folder_location_new,file_name)), 'w');
        fprintf(fid,
'measurment_number,time_of_measurment,latitude_actual,longitude_actual,latitude_pred
ict,longitude_predict,location_prediction_error,time_taken_to_determine_location\n');
save((strcat(main_folder_location_new,file_name)),'location_determination_information',
'-ascii','-append')
        fclose(fid);
    end
```

```
    if time_dependent==1
        if fit_at_one_zenith==1
            file_name=('\All AoP fitting information Timed');
        elseif multiple_zenith_angles==1
            file_name=(strcat('\All AoP fitting information Timed-Multi Zenith','-
',['Zenith=',num2str(which_zenith_mult(1))],' to ',[' ',num2str(which_zenith_mult(end))]));
        elseif select_data_within_range==1
            file_name=strcat('\All AoP fitting information Timed-Selected Data','-','Az
cover=',num2str(az_cover),'-Zen cover=',num2str(zen_cover));
        elseif sampling_method1==1
            file_name=strcat('\All AoP fitting information Timed-Sampled Data');
        else
            file_name=('\All AoP fitting information Timed-Full Dataset');
        end
        if azimuth_offset_correction==1
            file_name=strcat(file_name,'-Azimuth offset corrected');
        end
        if remove_overexposed==1
            file_name=strcat(file_name,'-Overexpose Removed');
        end
        file_name=strcat(file_name,'.txt');
        fid = fopen(strcat(main_folder_location_new,file_name), 'w');
        fprintf(fid,
'measurment_number,time_of_measurment,actual_Azimuth,actual_zenith,fitted_azimuth
_timed,fitted_zenith_timed,standard_deviation_measurment_vs_time_fitting,time_taken_
to_fit_AoP_timed\n');
        AoP_fitting_information_timed=cell2mat(AoP_fitting_information_timed);
save(strcat(main_folder_location_new,file_name),'AoP_fitting_information_timed','-
ascii','-append')
        fclose(fid);
        if fit_at_one_zenith==1
            file_name=('\All Location Determination Information Timed');
        elseif multiple_zenith_angles==1
            file_name=(strcat('\All Location Determination Information Timed-Multi Zenith','-
',['Zenith=',num2str(which_zenith_mult(1))],' to ',[' ',num2str(which_zenith_mult(end))]));
        elseif select_data_within_range==1
            file_name=strcat('\All Location Determination Information Timed-Selected
Data','-','Az cover=',num2str(az_cover) ,'-Zen cover=',num2str(zen_cover));
        elseif sampling_method1==1
            file_name=strcat('\All Location Determination Information Timed-Sampled Data');
        else
            file_name=('\All Location Determination Information Timed-Full Dataset');
        end
        if azimuth_offset_correction==1
            file_name=strcat(file_name,'-Azimuth offset corrected');
        end
```

```
        if remove_overexposed==1
           file_name=strcat(file_name,'-Overexpose Removed');
        end
        file_name=strcat(file_name,'.txt');
        fid = fopen(strcat(main_folder_location_new,file_name), 'w');
        fprintf(fid,
'measurment_number,time_of_measurment,latitude_actual,longitude_actual,latitude_pred
ict_new,longitude_predict_new,location_prediction_error_new,time_taken_to_determine
_location_timed\n');
location_determination_information_timed=cell2mat(location_determination_informatio
n_timed);
save(strcat(main_folder_location_new,file_name),'location_determination_information_ti
med','-ascii','-append')
        fclose(fid);
    end
end
    clear theta_timed_table;
    clear phi_timed_table;
    clear AoP_fit;
    clear AoP_fit_time;
    clear measurment_number_timed;
    clear standard_deviation_measurment_vs_time_fitting;
    clear time_taken_to_fit_AoP_timed;
    clear time_of_measurment_timed;
    clear pos_meas
    clear pos
    clear DoLP_meas
    clear AoP_meas
    clear az_meas_at_zen
    clear zen_meas_at_zen
    clear AoP_meas_at_zen
    clear DoLP_meas_at_zen
    clear Q_meas_at_zen
    clear U_meas_at_zen
    clear AoP_meas_fitting_data_save
    clear new_directory_name
    clear main_folder_location_new
    clear AoP_fitting_information
end
%%
                    %%%%%%%Start of functions%%%%%%%%%%%%%

function [sun_azimuth,sun_zenith, AoP_fit, meas_pos_std] =
fit_AoP_data(which_zenith,az_meas_at_zen,AoP_meas_at_zen,pos,fitting_conditions)
one_run=fitting_conditions(1);
number_of_run=fitting_conditions(2);
```

```
fit_at_one_zenith=fitting_conditions(3);
is_it_reduced_data_point_fitting=fitting_conditions(4);
no_idea=fitting_conditions(5);
aimuth_guess=fitting_conditions(6);
zenith_guess=fitting_conditions(7);
search_in_range=fitting_conditions(8);
how_many_phi=fitting_conditions(9);
how_many_theta=fitting_conditions(10);
AoP_error_limit=fitting_conditions(11);
updated_AoP_error_limt_by=fitting_conditions(12);
plot_it=fitting_conditions(13);
multiple_zenith_angles=fitting_conditions(14);
select_data_within_range=fitting_conditions(15);
sampling_method1=fitting_conditions(16);
under_water_mapping=fitting_conditions(17);

for loop=1:number_of_run
initial_point=1;
if fit_at_one_zenith==1
%%For fitting at one zenith angle
   theta_zen_p=which_zenith;
   end_pint=length(az_meas_at_zen);
   Zenith_meas=theta_zen_p;
   if loop==1
      Azimuth_meas=az_meas_at_zen(initial_point:end_pint,1);
      AoP_meas=AoP_meas_at_zen;
   else
      %updates with new data obtained from last loop
      AoP_meas=allowed_AoP_data;
      AoP_error_limit=updated_AoP_error_limt_by;
      Azimuth_meas=azimuth_at_allowed_AoP_data;
   end
elseif multiple_zenith_angles==1
   end_pint=length(pos);
   Azimuth_meas=pos(initial_point:end_pint,1);
   Zenith_meas=pos(initial_point:end_pint,2);
   AoP_meas=pos(initial_point:end_pint,3);
else
   %%For full data set fitting or smapling randomly
   end_pint=length(pos);
   Azimuth_meas=pos(initial_point:end_pint,1);
   Zenith_meas=pos(initial_point:end_pint,2);
   AoP_meas=pos(initial_point:end_pint,4);
end
cam_az=Azimuth_meas;
cam_zen=Zenith_meas;
```

```
lamda=480e-9; %%Wavelength
Temp=5782; %Sun's blackbody temperature in K on TOA (top of Atmosphere)
h=6.6261e-34; %plank's constant in Js
c=3e8; %light velocity in ms-1;
kb=1.3806e-23; %boltzmann's contant in JK-1
%%% Solar irradiance measured at earth surface
Rsun=6.95e5; %suns radius in km
Rearth=1.496e8; % radius of earth's orbit in km
I=(Rsun/Rearth)^2*2*pi*h*c^2/lamda^5*(1/(exp(h*c/(lamda*kb*Temp))-1))*lamda;   %
in Wm-2
% %particle detail
r=10e-9; %Particle radius
mr=1.53; %Real part of complex refractive index m = mr + i*mi
mi=0.007;%Imaginary part of complex refractive index m = mr + i*mi
m=mr+1i*mi;%Complex refractive index m = mr + i*mi
% For unpolarized light I=I; For LP|| I,Q=I; For LP_per I=I,Q=-I;
I=I; Q=0; U=0;V=0;
S=[I;  Q;  U;  V];
%%
len_min=0;
flag=1;
min_flag=0;
k=1;
if  fit_at_one_zenith==1  ||  multiple_zenith_angles==1  ||  select_data_within_range==1  ||
sampling_method1==1
   if no_idea==1
      phi_i=aimuth_guess-10;
      phi_f=aimuth_guess+10;
      theta_i=0;
      theta_f=90;
      how_many_phi=50;
      how_many_theta=50;
   else
      phi_i=aimuth_guess-search_in_range/2;
      phi_f=aimuth_guess+search_in_range/2;
      theta_i=zenith_guess-search_in_range/2;
      theta_f=zenith_guess+search_in_range/2;
   end
else
   phi_i=aimuth_guess-search_in_range/2;
   phi_f=aimuth_guess+search_in_range/2;
   theta_i=zenith_guess-search_in_range/2;
   theta_f=zenith_guess+search_in_range/2;
end
range=5;
factor=1;
```

```
while (flag==1)
   phi_range=linspace(phi_i,phi_f,how_many_phi);
   theta_range=linspace(theta_i,theta_f,how_many_theta);
for i=1:length(phi_range)
   if fit_at_one_zenith==1 && is_it_reduced_data_point_fitting==1
      for                                                    j=1:length(theta_range)
[DoLP_Pol,AoP_Pol,Q_Pol,U_Pol]=Rayleigh_Scat_det_fit_with_meas(r,m,lamda,S,thet
a_range(j),phi_range(i),cam_az,cam_zen,fit_at_one_zenith,under_water_mapping);
         AoP_S(:,j)=AoP_Pol;
         del_AoP_meas_pos=AoP_meas-AoP_Pol;
allowed_AoP_Error_data_position=find((abs(del_AoP_meas_pos)<AoP_error_limit)==1
);
allowed_AoP_Error_data=del_AoP_meas_pos(allowed_AoP_Error_data_position);
meas_pos_std=sqrt(sum((abs(del_AoP_meas_pos(allowed_AoP_Error_data_position))).^
2,'omitnan')./length(del_AoP_meas_pos));
         min_AoP_std(j)=meas_pos_std;
      end
   else
     parfor j=1:length(theta_range)
[DoLP_Pol,AoP_Pol,Q_Pol,U_Pol]=Rayleigh_Scat_det_fit_with_meas(r,m,lamda,S,thet
a_range(j),phi_range(i),cam_az,cam_zen,fit_at_one_zenith,under_water_mapping);
         AoP_S(:,j)=AoP_Pol;
         del_AoP_meas_pos=AoP_meas-AoP_Pol;
allowed_AoP_Error_data_position=find((abs(del_AoP_meas_pos)<AoP_error_limit)==1
);
allowed_AoP_Error_data=del_AoP_meas_pos(allowed_AoP_Error_data_position);
meas_pos_std=sqrt(sum((abs(del_AoP_meas_pos(allowed_AoP_Error_data_position))).^
2,'omitnan')./length(del_AoP_meas_pos));
  min_AoP_std(j)=meas_pos_std;
      end
   end
   min_AoP_std_all(1+len_min:len_min+length(min_AoP_std))=min_AoP_std;
   phi_all(1+len_min:len_min+length(min_AoP_std))=phi_range(i);
theta_all(1+len_min:len_min+length(min_AoP_std))=theta_range(1:length(min_AoP_std
));
   len_min=length(min_AoP_std_all);
end
if flag==1
   [min_v,min_pos]=min(min_AoP_std_all);
   phi_guess(k)=phi_all(min_pos);
   theta_guess(k)=theta_all(min_pos);
   if (min_v < min_flag) && abs(min_v-min_flag)<=5e-2
      flag=0;
      phi_f=phi_guess(k);
      theta_f=theta_guess(k);
   else
```

```
        phi_i=phi_guess(k)-range;
        phi_f=phi_guess(k)+range;
        theta_i=theta_guess(k)-range;
        theta_f=theta_guess(k)+range;
        range=range/factor;
        factor=factor*2;
        min_flag=min_v;
        len_min=0;
        clear min_AoP_std_all
        clear phi_all
        clear theta_all
    end
end
if one_run==1
    flag=0;
    phi_f=phi_guess(1);
    theta_f=theta_guess(1);
end
        k=k+1;
end
sun_azimuth(loop)=phi_f;
sun_zenith(loop)=theta_f;
[DoLP_Pol,AoP_Pol,Q_Pol,U_Pol]=Rayleigh_Scat_det_fit_with_meas(r,m,lamda,S,sun
_zenith(loop),sun_azimuth(loop),cam_az,cam_zen,fit_at_one_zenith,under_water_mappi
ng);
AoP_fit=AoP_Pol;
del_AoP_meas_pos=AoP_meas-AoP_Pol;
allowed_AoP_Error_data_position=find((abs(del_AoP_meas_pos)<AoP_error_limit)==1
);
allowed_AoP_Error_data=del_AoP_meas_pos(allowed_AoP_Error_data_position);
meas_pos_std=sqrt(sum((abs(del_AoP_meas_pos(allowed_AoP_Error_data_position))).^
2,'omitnan')./length(del_AoP_meas_pos(allowed_AoP_Error_data_position)));
end
end

%%
function [best_sun_AZ,best_sun_ZEN,best_fit_data,best_rmse, pos_meas_used_in_fit] =
fit_AoP_data_PSD_model(meas_data, fitting_condition)
zenith_guess=fitting_condition(1)
azimuth_guess=fitting_condition(2)
off_set_pass_down=fitting_condition(3)
which_zenith_pass_down=fitting_condition(4)
res_pass_down=fitting_condition(5)
AZ_rng_start1_pass_down=fitting_condition(6)
AZ_rng_end1_pass_down=fitting_condition(7)
AZ_rng_start2_pass_down=fitting_condition(8)
```

```
AZ_rng_end2_pass_down=fitting_condition(9)
MFP_pass_down=fitting_condition(10)
attenuation_coefficient_pass_down=fitting_condition(11)
vertical_measurement_depth_pass_down=fitting_condition(12)
depth_pass_down=fitting_condition(13)
sky_scat_ratio_pass_down=fitting_condition(14)
search_in_range=fitting_condition(15)
sun_pos_step=fitting_condition(16)
sun_ZEN_pass_down=zenith_guess-
search_in_range/2:sun_pos_step:zenith_guess+search_in_range/2;
sun_AZ_pass_down=azimuth_guess-
search_in_range/2:sun_pos_step:azimuth_guess+search_in_range/2;
[best_sun_AZ,best_sun_ZEN,best_fit_data,best_rmse,
pos_meas_used_in_fit]=PSD_model_Sim(meas_data,sun_ZEN_pass_down,sun_AZ_pas
s_down,off_set_pass_down,which_zenith_pass_down,res_pass_down,AZ_rng_start1_pas
s_down,AZ_rng_end1_pass_down,AZ_rng_start2_pass_down,AZ_rng_end2_pass_down
,MFP_pass_down,attenuation_coefficient_pass_down,vertical_measurement_depth_pass
_down,depth_pass_down,sky_scat_ratio_pass_down);
end

function                                                    [latitude_predict,
longitude_predict]=location_predict(month,day,year,time,GMT,DLS,sun_az,sun_zen,nu
mber_of_iteration_for_location_prediction,no_idea_of_location)
for run_count=1:number_of_iteration_for_location_prediction
   if run_count==1 && number_of_iteration_for_location_prediction==2
      no_idea_of_location=1;
   end
   if run_count==2
      no_idea_of_location=0;
   end
%%%Estimate the position on earth
if no_idea_of_location==1
   latitude_low_limit=-90;
   latitude_high_limit=90;
   if GMT<0
      longitude_low_limit=15*GMT+15;
      longitude_high_limit=15*GMT-15;
   else
      longitude_low_limit=15*GMT-15;
      longitude_high_limit=15*GMT+15;
   end
else
   lat_initial_guess=latitude_predict;
   lon_initial_guess=longitude_predict;
   if run_count==2
      range_half=10;
```

```
    elseif run_count>2
        range_half=2/(run_count-2)^2;
    end
    if lat_initial_guess<0
        latitude_low_limit=lat_initial_guess+range_half;
        latitude_high_limit=lat_initial_guess-range_half;
    elseif lat_initial_guess>0
        latitude_low_limit=lat_initial_guess-range_half;
        latitude_high_limit=lat_initial_guess+range_half;
    end
    if lon_initial_guess<0
        longitude_low_limit=lon_initial_guess+range_half;
        longitude_high_limit=lon_initial_guess-range_half;
    elseif lon_initial_guess>0
        longitude_low_limit=lon_initial_guess-range_half;
        longitude_high_limit=lon_initial_guess+range_half;
    end
end
latitude_table=linspace(latitude_low_limit,latitude_high_limit,1e4);
longitude_table=linspace(longitude_low_limit,longitude_high_limit,1e3);
if leapyear(year)
    day_month=[31 29 31 30 31 30 31 31 30 31 30 31];
else
    day_month=[31 28 31 30 31 30 31 31 30 31 30 31];
end
%calculate day number of the year (n)
if(month==1)
    n=day;
elseif(month==2)
    n=day+31;
elseif(month>=3)
    n=day+sum(day_month(1:month-1))+leapyear(year);
end
%Declination angle (independent of position on earth; dependent on n)
delta=23.45.*sind(360./365.*(284+n));
%%%Equation of time
B=360/365*(n-81);
EOT=0.165*sind(2*B)-0.126*cosd(B)-0.025*sind(B);
Lstd=15*GMT; %For mountain Standard Time in Arizona
CT=time; %Clock time
for lat_count=1:length(latitude_table)
    latitude=latitude_table(lat_count);
    for lon_count=1:length(longitude_table)
        longitude=longitude_table(lon_count);
```

```
LST=CT-(1/15)*(Lstd-longitude)+EOT-DLS;
%%%Solar hour angle
omega=15.*(LST-12);
theta=acosd(cosd(latitude).*cosd(omega).*cosd(delta)+sind(latitude).*sind(delta));
for i=1:length(time)
if theta(i)>90 %i.e sun is below the horizon (equator)
   theta(i)=90-theta(i);
else
   theta(i)=theta(i);
end
if theta(i)>0 % i.e its sun in above horizon (equator)
   phi(i)=acosd((cosd(delta).*sind(latitude).*cosd(omega(i))-
sind(delta).*cosd(latitude))./cosd(90-theta(i)));
else
   phi(i)=acosd((cosd(delta).*sind(latitude).*cosd(omega(i))-
sind(delta).*cosd(latitude))./cosd(theta(i)));
end
end
% %Proper sign convention:
for i=1:length(time)
if omega(i)<0
   phi(i)=phi(i)*-1; %(bcz: phi: the sun's azimuth angle is negative east of south and
positive west of south )
end
end
% If phi is calculated from due north
phi_N_CW=phi+180; %if calculated From due north CW
phi=phi_N_CW;
% As sun can go below the horrizon this part is just to make the zenith positive or (0 to
180)
for i=1:length(time)
   if theta(i)<0
   theta(i)=90-theta(i);
   end
end
theta_table(lat_count,lon_count)=theta;
phi_table(lat_count,lon_count)=phi;
   end
end
dist_between_combinations=sqrt((theta_table-sun_zen).^2+(phi_table-sun_az).^2);
min_dist=min(min(dist_between_combinations));
[row_number,column_number]=find(dist_between_combinations==min_dist);
theta_predict=theta_table(row_number,column_number);
phi_predict=phi_table(row_number,column_number);
latitude_predict=latitude_table(row_number);
longitude_predict=longitude_table(column_number);
```

```
end
end

%%
function                                                    [fitting_data,AoP_fit_time]=
time_dependent_fitting(theta,phi,pos,multiple_zenith_angles,fit_at_one_zenith,which_ze
nith,number_of_images,under_water_mapping)
   if fit_at_one_zenith==1
      Zenith_meas=which_zenith;
      Azimuth_meas=pos(:,1);
   elseif multiple_zenith_angles==1
      Azimuth_meas=pos(:,1);
      Zenith_meas=pos(:,2);
   else
      Azimuth_meas=pos(:,1);
      Zenith_meas=pos(:,2);
   end
   cam_az_Pol=Azimuth_meas;
   cam_zen_Pol=Zenith_meas;
length_sim=0;
number_of_theta_s_phi_s_at_a_time=ceil(length(Azimuth_meas)/number_of_images);
for i =1:number_of_images
   if i==number_of_images
      cam_az=pos(1+length_sim:length(Azimuth_meas),2);
   else
      cam_az=pos(1+length_sim:length_sim+number_of_theta_s_phi_s_at_a_time,1);
   end
   if fit_at_one_zenith==0
      if i==number_of_images
         cam_zen=pos(1+length_sim:length(Azimuth_meas),2);
      else
         cam_zen=pos(1+length_sim:length_sim+number_of_theta_s_phi_s_at_a_time,2);
      end
   else
      cam_zen=cam_zen_Pol;
   end
   theta_dyn=theta(i);
   phi_dyn=phi(i);
   %%
   lamda=480e-9; %%Wavelength
   Temp=5782; %Sun's blackbody temperature in K on TOA (top of Atmosphere)
   h=6.6261e-34; %plank's constant in Js
   c=3e8; %light velocity in ms-1;
   kb=1.3806e-23; %boltzmann's contant in JK-1
   Rsun=6.95e5; %suns radius in km
   Rearth=1.496e8; % radius of earth's orbit in km
```

```
    I=(Rsun/Rearth)^2*2*pi*h*c^2/lamda^5*(1/(exp(h*c/(lamda*kb*Temp))-1))*lamda;
% in Wm-2
    r=100e-9; %Particle radius
    mr=1.53; %Real part of complex refractive index m = mr + i*mi
    mi=0.007;%Imaginary part of complex refractive index m = mr + i*mi
    m=mr+1i*mi;%Complex refractive index m = mr + i*mi
    I=I; Q=0; U=0;V=0;
    S=[I;  Q;  U;  V];
    %%
[DoLP_Pol,AoP_Pol,Q_Pol,U_Pol]=Rayleigh_Scat_det_fit_with_meas(r,m,lamda,S,thet
a_dyn,phi_dyn,cam_az,cam_zen,fit_at_one_zenith,under_water_mapping);
    if i==number_of_images
sim.pos(1+length_sim:length(Azimuth_meas),1:4)=[DoLP_Pol,AoP_Pol,Q_Pol,U_Pol];
    else
sim.pos(1+length_sim:length_sim+number_of_theta_s_phi_s_at_a_time,1:4)=[DoLP_Po
l,AoP_Pol,Q_Pol,U_Pol];
    end
    length_sim=length(sim.pos);
end
if fit_at_one_zenith==1
    sim.pos(:,5)=sim.pos(:,4);
    sim.pos(:,4)=sim.pos(:,3);
    sim.pos(:,3)=sim.pos(:,2);
    sim.pos(:,2)=sim.pos(:,1);
    sim.pos(:,1)=cam_az_Pol;
    AoP_fit_time=sim.pos(:,3);
else
    sim.pos(:,6)=sim.pos(:,4);
    sim.pos(:,5)=sim.pos(:,3);
    sim.pos(:,4)=sim.pos(:,2);
    sim.pos(:,3)=sim.pos(:,1);
    sim.pos(:,2)=cam_zen_Pol;
    sim.pos(:,1)=cam_az_Pol;
    AoP_fit_time=sim.pos(:,4);
end
    fitting_data=sim.pos;
end

function
[new_pos,which_zenith,new_meas_time,time,dir_loc]=get_muliple_zenith_maping(folde
r_location,data_file_name,time_file_name)
a=strsplit(folder_location,'\');
dir_loc=a(1);
for i=2:length(a)-1
    dir_loc=strcat(dir_loc,'\',a(i));
end
```

```
A=dir(dir_loc{1});
k=1;
for i=3:size(A,1)
   if A(i).isdir==1
   all_folder_names{k}=A(i).name;
   k=k+1;
   end
end
new_pos_len=0;
new_meas_time_len=0;
for i=1:length(all_folder_names)
new_file_location=strcat(dir_loc,'\',all_folder_names{i});
s_md = strcat(new_file_location,data_file_name); %location+filename for the measurment
data
s_td=strcat(new_file_location,time_file_name);  %location+filename for the measurment
time
%%
pos_meas_temp=dlmread(s_md{1});
meas_time_temp=dlmread(s_td{1});
%%%Averaging over certain data points
total_num_of_zenith=find(pos_meas_temp(1,1)~=pos_meas_temp(:,1),1)-1;
avg_over_how_many_data=total_num_of_zenith/10;
new_len1=ceil(length(pos_meas_temp)/avg_over_how_many_data);
for cc1=1:new_len1
   init=1+avg_over_how_many_data*(cc1-1);
   fin=avg_over_how_many_data*(cc1-1)+avg_over_how_many_data;
   if cc1==new_len1
      fin=length(pos_temp);
   end
   pos_temp(cc1,:)=mean(pos_meas_temp(init:fin,:));
end
mid_position_of_zenith=ceil((find(pos_temp(:,1)~=pos_temp(1,1),1)-1)/2);
other_position_of_zenith1=mid_position_of_zenith-ceil(mid_position_of_zenith/2);
other_position_of_zenith2=mid_position_of_zenith-ceil(mid_position_of_zenith/3);
other_position_of_zenith3=mid_position_of_zenith+ceil(mid_position_of_zenith/3);
other_position_of_zenith4=mid_position_of_zenith+ceil(mid_position_of_zenith/2);
which_zenith(i,:)                =                [pos_temp(other_position_of_zenith1,2)
pos_temp(other_position_of_zenith2,2)                pos_temp(mid_position_of_zenith,2)
pos_temp(other_position_of_zenith3,2)           pos_temp(other_position_of_zenith4,2)]';
%Uncomment and update the row number if you are using one zenith from full data set
number_of_zenith=(find(pos_temp(:,1)~=pos_temp(1,1),1)-1);
err_mean=abs(mean(pos_temp(2:number_of_zenith-1,2)-
pos_temp(3:number_of_zenith,2)))/6;
zenith_res=err_mean;
%Time of day
hour=meas_time_temp(:,4);
```

```
minute=meas_time_temp(:,5);
second=meas_time_temp(:,6);
time1=(hour+(minute/60)+(second/60/60));
%In video mode, sometimes several number of images may have
%the same time. We need to consider those images as one single image
k1=1;
step=0;
temp_t=0;
for ic=1:length(time1)
   if isempty(find(time1(temp_t+1:end,1)~=time1(temp_t+1,1),1)-1)~=1
      step(k1)=(find(time1(temp_t+1:end,1)~=time1(temp_t+1,1),1)-1)+temp_t;
   else
      step(k1)=temp_t+1;
   break;
   end
   temp_t=step(k1);
   k1=k1+1;
end
number_of_images(i)=length(step); %consider only these 'step' in calculating the number
of images
time(i)=mean((hour(step)+(minute(step)/60)+(second(step)/60/60)));
new_meas_time_temp{:,i}=meas_time_temp(step,:);      %%Accumulate      all      the
measurement times for all the data sets
temp_metric(i)=length(pos_temp)/number_of_images(i);
%%%Accumulate the data for each measurement (choose the center of the zenith angle for
each measurement)
for cc2=1:length(which_zenith(i,:))
   theta_zen_p(i)=which_zenith(i,cc2);
   k_k=find(abs(pos_temp(:,2)-
theta_zen_p(i))<zenith_res);%find(abs(pos(:,2)==theta_zen_p));%
   az_meas_at_zen{cc2,i}=pos_temp(k_k,1);
   AoP_meas_at_zen{cc2,i}=pos_temp(k_k,4);
   temp_j=0;
   met1=ceil(length(AoP_meas_at_zen{cc2,i})/number_of_images(i));
   for cc3=1:number_of_images(i)
      count_t1=1+(cc3-1)*met1:cc3*met1;
      if cc3==number_of_images(i)
         count_t1=1+(cc3-1)*met1:length(AoP_meas_at_zen{cc2,i});
      end
      for cc4=1:size(new_meas_time_temp{:,i},2)
new_meas_time_temp2{cc2,i}(count_t1,cc4)=new_meas_time_temp{:,i}(cc3,cc4);
      end
   end
end
   new_pos_len=new_pos_len+length(AoP_meas_at_zen{cc2,i});
   new_meas_time_len=new_meas_time_len+size(meas_time_temp,1);
```

```
end
new_pos=zeros(new_pos_len,3);
new_meas_time=zeros(new_meas_time_len,size(meas_time_temp,2));
for i=1:length(all_folder_names)
   for m=1:size(which_zenith,2)
      new_pos(m+(i-1)*size(which_zenith,2),1)=az_meas_at_zen{m,i}(i,1);
      new_pos(m+(i-1)*size(which_zenith,2),2)=which_zenith(i,m);
      new_pos(m+(i-1)*size(which_zenith,2),3)=AoP_meas_at_zen{m,i}(i,1);
      new_meas_time(m+(i-1)*size(which_zenith,2),:)=new_meas_time_temp2{m,i}(i,:);
      len1=length(AoP_meas_at_zen{m,i});
      for j=2:len1
         new_pos(m+(i-1)*size(which_zenith,2)+(j-
1)*length(all_folder_names)*size(which_zenith,2),1)=az_meas_at_zen{m,i}(j,1);
         new_pos(m+(i-1)*size(which_zenith,2)+(j-
1)*length(all_folder_names)*size(which_zenith,2),2)=which_zenith(i,m);
         new_pos(m+(i-1)*size(which_zenith,2)+(j-
1)*length(all_folder_names)*size(which_zenith,2),3)=AoP_meas_at_zen{m,i}(j,1);
         new_meas_time(m+(i-1)*size(which_zenith,2)+(j-
1)*length(all_folder_names)*size(which_zenith,2),:)=new_meas_time_temp2{m,i}(j,:);
      end
   end
end
new_pos = new_pos(any(new_pos,2),:);
new_meas_time = new_meas_time(any(new_meas_time,2),:);
%
mult_zenith=which_zenith;
end

function[pos]=remove_overexposed_data(start,finish,pos)
   camera_azimuth=pos(:,1);
   %handel the fact that the start and end point can't be larger than
   %360 and less than 0
   if start>360
      start=start-360;
   elseif start<0
      start=start+360;
   end
   if finish>360
      finish=finish-360;
   elseif finish<0
      finish=finish+360;
   end
   aa=find(abs(camera_azimuth-
start)<(camera_azimuth(find(camera_azimuth~=camera_azimuth(1),1))-
camera_azimuth(1))*1);
```

```
   bb=find(abs(camera_azimuth-
finish)<(camera_azimuth(find(camera_azimuth~=camera_azimuth(1),1))-
camera_azimuth(1))*1);
   if aa(1)>bb(1)
      pos=pos(bb(end)+1:aa(1)-1,:);
   elseif aa(1)<bb(1)
      temp=zeros(size(pos));
      temp(1:aa(1)-1,:)=pos(1:aa(1)-1,:);
      temp(bb(end)+1:end,:)=pos(bb(end)+1:end,:);
      temp=temp(any(temp,2),:);
      pos=temp;
   end
end

function [pos2]=select_data(pos,meas_time,zen_cover,az_cover)
%find the zenith angles
first_az_pos=find(pos(:,1)==pos(1,1)); %find the positions of the first azimuth angle; This
is equal to the first zenith angle coverage by the camera
m=ceil(length(first_az_pos)/2); %middle point
l_c=length(first_az_pos); %the period of occurance of the zenith angles
del_zen=pos(first_az_pos(end),2)-pos(first_az_pos(1),2); %difference between the first
and last zenith angle
cover=(ceil(l_c/del_zen*zen_cover/10)*10)/2; %find the nearest 10th place % how many
data points should be covered from the mid point (both + and -)
po1=zeros(size(pos));
for i=1:length(pos(:,1))/l_c
po1(m-cover+l_c*(i-1):m+cover+l_c*(i-1),:)=pos(m-cover+l_c*(i-1):m+cover+l_c*(i-
1),:);
end
po1=po1(any(po1,2),:);
az_angle_coverge_for_each_image=8
second_az=(find(pos(:,1)~=pos(1,1),1)); %position of the second azimuth
err_mean=pos(second_az,1)-pos(1,1);
az_res=0.02;
com1=po1(1,1)+az_angle_coverge_for_each_image;
if com1>360
   com1=com1-360;
end
aa=abs(po1(:,1)-com1);
[m1 ind]=min(aa);
a=ind;
b(1,1)=1;
for i=1:length(meas_time(:,1))
b(i+1,1)=a(end)*i;
if b(i+1,1)>length(po1(:,1))
b(i+1,1)=0;
```

```
break;
end
end
b=b(any(b,2),:);
first_az_pos=b(1):b(2);
m=ceil(length(first_az_pos)/2);
l_c=length(first_az_pos); %periode of the azimuth angles
del_az=po1(first_az_pos(end),1)-po1(first_az_pos(1),1);
cover=(ceil(l_c/del_az*az_cover/10)*10)/2; %find the nearest 10th place % how many data points should be covered from the mid point (both + and -)
po2=zeros(b(end),size(po1,2));
for i=1:length(po2)/l_c
po2(m-cover+l_c*(i-1):m+cover+l_c*(i-1),:)=po1(m-cover+l_c*(i-1):m+cover+l_c*(i-1),:);
end
po2=po2(any(po2,2),:);
pos2=po2;
end
%%
function [new]=get_maping_within_limit(data,limit)
   k_k=find(abs(data(:,3))<limit);
   new(:,1)=data(k_k,1); %Azimuth
   new(:,2)=data(k_k,2); %Zenith
   new(:,3)=data(k_k,3); %DoCP/AoP/DoLP
end

function [maping_new]=make_off_center_camera_correction(maping_new)
%% Zenith and Azimuth Correction
AB=0.83;
AC=6.625;
theta=maping_new(:,2)';
phi=maping_new(:,1)';
nair=1;
nwater=1.33;
theta_t_D = asind(AB/AC)
theta_i_D = asind(nair * sind (theta_t_D)/nwater)
for i =1:length(theta)
      theta_temp=theta(i);
      theta_new(i)= acosd(cosd(theta_temp) / sqrt((tand (theta_t_D - theta_i_D))^2 + (sind(theta_temp))^2+ (cosd(theta_temp))^2));
      del_phi(i) = asind((tand(theta_t_D - theta_i_D))/ sqrt(( tand (theta_t_D - theta_i_D))^2+(sind(theta_temp))^2));
end
del_theta=theta_new-theta;
maping_new(:,2)=theta_new';
```

```
for i=1:length(maping_new(:,1))
   %Bring all the azimuth angles between 0 to 360
   if maping_new(i,1)<0
      maping_new(i,1)=maping_new(i,1)+360;
   elseif maping_new(i,1)>360
      maping_new(i,1)=maping_new(i,1)-360;
   end
   %%Either Clockwise or Counter-Clockwise
   if maping_new(i,1)<=180 && maping_new(i,1)>=0
      maping_new(i,1)=phi(i)-del_phi(i);
   else
      maping_new(i,1)=phi(i)+del_phi(i);
   end
   %Bring all the azimuth angles between 0 to 360
   if maping_new(i,1)<0
      maping_new(i,1)=maping_new(i,1)+360;
   elseif maping_new(i,1)>360
      maping_new(i,1)=maping_new(i,1)-360;
   end
end
end

function
[ouput_data]=average_over_whole_data_deltheta_delphi(input_data,del_theta,del_phi)
%% Averaging throughout the whole data with certain del_theta and del_phi
phi_all=0:del_phi:360;
theta_all=0:del_theta:90;
for i=1:length(phi_all)
   for j=1:length(theta_all)
      phi_Mat(i,j)=phi_all(i);
      theta_Mat(i,j)=theta_all(j);
   end
end
%% Make the column matrix
phi_Mat=phi_Mat';
theta_Mat=theta_Mat';
phi_col_Mat=phi_Mat(:);
theta_col_Mat=theta_Mat(:);
%% Make the 1D matrix for fast operation
phi_col_Mat(1:end-length(theta_all),2)=phi_col_Mat(length(theta_all)+1:end,1);
theta_col_Mat(1:end-1,2)=theta_col_Mat(2:end,1);
theta_90_pos=find(theta_col_Mat(:,1)==90);
theta_col_Mat(theta_90_pos,:)=[];
phi_col_Mat(theta_90_pos,:)=[];
phi_360_pos=find(phi_col_Mat(:,1)==360);
phi_col_Mat(phi_360_pos,:)=[];
```

```
theta_col_Mat(phi_360_pos,:)=[];
%% Seperate the data within a range and do average of those data
num_of_column=size(input_data,2);
ouput_data=zeros(length(phi_col_Mat),num_of_column);
parfor i=1:length(ouput_data)
do_average=select_data_within_limit_greater_az(input_data,phi_col_Mat(i,1));
do_average=select_data_within_limit_less_az(do_average,phi_col_Mat(i,2));
do_average=select_data_within_limit_greater_zen(do_average,theta_col_Mat(i,1));
do_average=select_data_within_limit_less_zen(do_average,theta_col_Mat(i,2));
if ~isempty(do_average)
   ouput_data(i,:)=mean(do_average,1); %average over each column
end
end
%% Remove any row with all zero
ouput_data = ouput_data(any(ouput_data,2),:); %%Removes any row with all zero
end

function [best_sun_AZ, best_sun_ZEN, best_fit_AoP_data, best_rmse,
pos_meas]=PSD_model_Sim(meas_data,sun_ZEN_pass_down,sun_AZ_pass_down,off_
set_pass_down,which_zenith_pass_down,res_pass_down,AZ_rng_start1_pass_down,AZ
_rng_end1_pass_down,AZ_rng_start2_pass_down,AZ_rng_end2_pass_down,MFP_pass
_down,attenuation_coefficient_pass_down,vertical_measurement_depth_pass_down,dept
h_pass_down,sky_scat_ratio_pass_down)
sun_ZEN_all=sun_ZEN_pass_down;
sun_AZ_all=sun_AZ_pass_down;
off_set=off_set_pass_down;
which_zenith=which_zenith_pass_down;
res=res_pass_down;
AZ_rng_start1=AZ_rng_start1_pass_down;
AZ_rng_end1=AZ_rng_end1_pass_down;
AZ_rng_start2=AZ_rng_start2_pass_down;
AZ_rng_end2=AZ_rng_end2_pass_down;
MFP=MFP_pass_down; %in m
attenuation_coefficient=attenuation_coefficient_pass_down; % unit m^-1 %
vertical_measurement_depth=vertical_measurement_depth_pass_down; %inch
vertical_measurement_depth=convlength(vertical_measurement_depth,'in','m'); %in m
sky_light=0.3;
sun_light=1-sky_light;
depth=depth_pass_down;
skylight_scat_percent_all=sky_scat_ratio_pass_down;
total_scat_num=1; %number of mie scattering for sun light inside water
%%
AZ_meas=meas_data(:,1);
ZEN_meas=meas_data(:,2);
%% Zenith Range for fitting
Zen_rng_start=which_zenith-res_pass_down;
```

```
Zen_rng_end=which_zenith+res_pass_down;
%% Measured data for fitting
[pos_tem]=select_data_within_limit_greater_zen_psd(meas_data,Zen_rng_start);
%Lower Zenith angle
[pos_tem]=select_data_within_limit_less_zen_psd(pos_tem,Zen_rng_end);          %Upper
Zenith angle
[pos1]=select_data_within_limit_greater_az_psd(pos_tem,AZ_rng_start1);         %Lower
Zenith angle
[pos1]=select_data_within_limit_less_az_psd(pos1,AZ_rng_end1); %Upper Zenith angle
[pos2]=select_data_within_limit_greater_az_psd(pos_tem,AZ_rng_start2);         %Lower
Zenith angle
[pos2]=select_data_within_limit_less_az_psd(pos2,AZ_rng_end2); %Upper Zenith angle
pos_meas=[pos1;pos2];
pos_meas=unique(pos_meas,'rows');
meas_data_for_fit=pos_meas;
clear pos_tem;
AoP_meas=meas_data_for_fit(:,4);
%%
for i_sun_AZ=1:length(sun_AZ_pass_down)
   for i_sun_ZEN=1:length(sun_ZEN_pass_down)
      sun_AZ_all(i_sun_AZ,i_sun_ZEN)=sun_AZ_pass_down(i_sun_AZ);
      sun_ZEN_all(i_sun_AZ,i_sun_ZEN)=sun_ZEN_pass_down(i_sun_AZ);
   end
end
sun_AZ_all=sun_AZ_all';
sun_AZ_all=sun_AZ_all(:);
sun_ZEN_all=sun_ZEN_all';
sun_ZEN_all=sun_ZEN_all(:);
%%
for i_sun_pos=1:length(sun_AZ_all)
sun_AZ=sun_AZ_all(i_sun_pos,1);
sun_ZEN=sun_ZEN_all(i_sun_pos,1);
%% Skylight Ref
Skylight_refract=
find_sim_data_in_between_zenith_angles_v3_2(0,1,0,sun_ZEN,total_scat_num);
%Skylight polarization going (refracted) into water
Intensity_distribution_Skylight_refract=Skylight_refract(:,3)/max(Skylight_refract(:,3));
distance_at_diff_zenith=(vertical_measurement_depth./cosd(Skylight_refract(:,2)));
different_intensity_ratio_at_diff_zenith_skylight_refract=exp(-
distance_at_diff_zenith*attenuation_coefficient);
%% Sunlight Scat
Sunlight_scat=cell(total_scat_num,1);
for i=1:total_scat_num
   Sunlight_scat{i,1}=
find_sim_data_in_between_zenith_angles_v3_2(1,0,0,sun_ZEN,i);
   Sunlight_scat{i,1}=sortrows(Sunlight_scat{i,1},1);
```

```
Intensity_distribution_sun_light_scat(:,i)=Sunlight_scat{i,1}(:,3)/max(Sunlight_scat{i,1}
(:,3));
end
Skylight_scat=
find_sim_data_in_between_zenith_angles_v3_2(0,0,1,sun_ZEN,total_scat_num);
Intensity_distribution_Skylight_scat=Skylight_scat(:,3)/max(Skylight_scat(:,3));
%%
AZ_cam=Skylight_refract(:,1);
ZEN_cam=Skylight_refract(:,2);
%%
skylight_refract_percent=1-skylight_scat_percent_all;
skylight_scat_percent=skylight_scat_percent_all;
sunlight_scat_percent=skylight_scat_percent;
%%
stat_mc_part_I=0;
stat_mc_part_Q=0;
stat_mc_part_U=0;
stat_mc_part_V=0;
% Add all contribution from Sunlight scattering
for i=1:total_scat_num %size(num_of_scat,1)
% Normalize
stat_mc_part_I=stat_mc_part_I+Sunlight_scat{i,1}(:,3)*sunlight_scat_percent.*Intensity
_distribution_sun_light_scat(:,i)./Sunlight_scat{i,1}(:,3);
stat_mc_part_Q=stat_mc_part_Q+Sunlight_scat{i,1}(:,4)*sunlight_scat_percent.*Intensit
y_distribution_sun_light_scat(:,i)./Sunlight_scat{i,1}(:,3);
stat_mc_part_U=stat_mc_part_U+Sunlight_scat{i,1}(:,5)*sunlight_scat_percent.*Intensit
y_distribution_sun_light_scat(:,i)./Sunlight_scat{i,1}(:,3);
stat_mc_part_V=stat_mc_part_V+Sunlight_scat{i,1}(:,6)*sunlight_scat_percent.*Intensit
y_distribution_sun_light_scat(:,i)./Sunlight_scat{i,1}(:,3);
end
%%
% contribution from Sunlight Scat (1), Skylight refraction and Skylight scattering (1)
stat_mc_part_I=sun_light*stat_mc_part_I+sky_light*(skylight_refract_percent*Skylight
_refract(:,3).*Intensity_distribution_Skylight_refract.*different_intensity_ratio_at_diff_z
enith_skylight_refract./Skylight_refract(:,3)+skylight_scat_percent*Skylight_scat(:,3).*I
ntensity_distribution_Skylight_scat./Skylight_scat(:,3));
stat_mc_part_Q=sun_light*stat_mc_part_Q+sky_light*(skylight_refract_percent*Skylig
ht_refract(:,4).*Intensity_distribution_Skylight_refract.*different_intensity_ratio_at_diff
_zenith_skylight_refract./Skylight_refract(:,3)+skylight_scat_percent*Skylight_scat(:,4).
*Intensity_distribution_Skylight_scat./Skylight_scat(:,3));
stat_mc_part_U=sun_light*stat_mc_part_U+sky_light*(skylight_refract_percent*Skylig
ht_refract(:,5).*Intensity_distribution_Skylight_refract.*different_intensity_ratio_at_diff
_zenith_skylight_refract./Skylight_refract(:,3)+skylight_scat_percent*Skylight_scat(:,5).
*Intensity_distribution_Skylight_scat./Skylight_scat(:,3));
stat_mc_part_V=sun_light*stat_mc_part_V+sky_light*(skylight_refract_percent*Skylig
ht_refract(:,6).*Intensity_distribution_Skylight_refract.*different_intensity_ratio_at_diff
```

```
_zenith_skylight_refract./Skylight_refract(:,3)+skylight_scat_percent*Skylight_scat(:,6).
*Intensity_distribution_Skylight_scat./Skylight_scat(:,3));
%%
stat_mc_part_S=[stat_mc_part_I stat_mc_part_Q stat_mc_part_U
stat_mc_part_V]./stat_mc_part_I;
%%
% DoLP
stat_mc_part_dolp=sqrt(stat_mc_part_S(:,2).^2+stat_mc_part_S(:,3).^2)./stat_mc_part_S
(:,1);
% DoCP
stat_mc_part_docp=stat_mc_part_S(:,4)./stat_mc_part_S(:,1);
for i=1:length(AZ_cam)
 stat_mc_part_aop(i,1)=0.5*myatand_0to180(stat_mc_part_S(i,3),stat_mc_part_S(i,2));
end
%%
stat_mc_part=[AZ_cam ZEN_cam stat_mc_part_dolp stat_mc_part_docp
stat_mc_part_aop];
%%
simulation_data_temp=stat_mc_part;
simulation_data=[simulation_data_temp(:,1:2) simulation_data_temp(:,5)];
%%
for i=1:length(simulation_data(:,1))
      simulation_data(i,1)=simulation_data(i,1)-(246-sun_AZ) + off_set; %if larger than
246 then add the difference otherwise minus
      if simulation_data(i,1)>360
         simulation_data(i,1)=simulation_data(i,1)-360;
      end
      if simulation_data(i,1)<0
         simulation_data(i,1)=simulation_data(i,1)+360;
      end
end
simulation_data=sortrows(simulation_data,1);
%% interpolate to get data at the measured AZ and ZEN
   AZ_sim=simulation_data(:,1);
   ZEN_sim=simulation_data(:,2);
   AoP_sim=simulation_data(:,3);
   F_AoP_Sim = scatteredInterpolant(AZ_sim,ZEN_sim,AoP_sim,'nearest','nearest');
   new_AoP_Sim= F_AoP_Sim(AZ_meas,ZEN_meas);
simulation_data=[];
simulation_data=[AZ_meas ZEN_meas new_AoP_Sim];
%% Sim data select
[pos_tem]=select_data_within_limit_greater_zen_psd(simulation_data,Zen_rng_start);
%Lower Zenith angle
[pos_tem]=select_data_within_limit_less_zen_psd(pos_tem,Zen_rng_end);      %Upper
Zenith angle
```

```
[pos1]=select_data_within_limit_greater_az_psd(pos_tem,AZ_rng_start1);          %Lower
Zenith angle
[pos1]=select_data_within_limit_less_az_psd(pos1,AZ_rng_end1); %Upper Zenith angle
[pos2]=select_data_within_limit_greater_az_psd(pos_tem,AZ_rng_start2);          %Lower
Zenith angle
[pos2]=select_data_within_limit_less_az_psd(pos2,AZ_rng_end2); %Upper Zenith angle
pos_sim=[pos1;pos2];
pos_sim=unique(pos_sim,'rows');
simulation_data_for_fit=pos_sim;
%%
  fit_AZ(:,i_sun_pos)=simulation_data_for_fit(:,1);
  fit_ZEN(:,i_sun_pos)=simulation_data_for_fit(:,2);
  AoP_fit(:,i_sun_pos)=simulation_data_for_fit(:,3);
  del_AoP(:,i_sun_pos)=AoP_meas-AoP_fit(:,i_sun_pos);
  allowed_AoP_Error_data_position=find((abs(del_AoP(:,i_sun_pos))<150)==1);        %
Setting a limit for the AoP error to reduce noise
meas_pos_rmse=sqrt(mean((abs(del_AoP(allowed_AoP_Error_data_position,i_sun_pos)
)).^2)); %Root mean square error
  min_AoP_rmse(i_sun_pos)=meas_pos_rmse;
end
[min_pos]=find((min_AoP_rmse==min(min_AoP_rmse)));
best_sun_AZ=mean(sun_AZ_all(min_pos)) % take the mean of the sun AZ if there is
multiple point having smallest offset
best_sun_ZEN=mean(sun_ZEN_all(min_pos)) % take the mean of the sun ZEN if there is
multiple point having smallest offset
best_fit_data=[fit_AZ(:,round(mean(min_pos)))          fit_ZEN(:,round(mean(min_pos)))
AoP_fit(:,round(mean(min_pos)))];
best_fit_AoP_data=AoP_fit(:,round(mean(min_pos)));
best_rmse=min_AoP_rmse(round(mean(min_pos)))
%% AoP line plot fit and meas
which_zenith_for_line_plot=round((min(meas_data_for_fit(:,2))+max(meas_data_for_fit
(:,2)))/2);
res_for_line_plot=1;
[meas_fit_new]=get_maping_at_specific_zenith(meas_data_for_fit,which_zenith_for_lin
e_plot,res_for_line_plot);
[fit_new]=get_maping_at_specific_zenith(best_fit_data,which_zenith_for_line_plot,res_f
or_line_plot);

figure
scatter(meas_fit_new(:,1),meas_fit_new(:,4),'filled')
hold on
scatter(fit_new(:,1),fit_new(:,3),'filled')
% xlim([0 150])
yticks([0:50: 200])
xlabel('Azimuth')
ylabel('AoP')
```

```
set(gca, 'FontSize', 50);
set(gca, 'linewidth', 5);
title(strcat('Zenith-',num2str(which_zenith_for_line_plot)),'FontSize',30)
legend('Measurement','Simulation-Sun(1)+Sky(1)+Skyref','FontSize',25)
   legend box off
set(gcf,'color','w');
end

function [simulated_data_temp] =
find_sim_data_in_between_zenith_angles_v3_2(do_Sunlight_scat,do_Skylight_refract,d
o_Skylight_scat,desired_zenith_angle,sun_scattering_events_pass_down)
location_of_sun_scat_data='C:\Users\meraf\OneDrive\Desktop\Underwater Project
Extension\MC Simulation Data_ Depth dependent\';
location_of_sky_ref_data='C:\Users\meraf\OneDrive\Desktop\Underwater Project
Extension\MC Simulation Data_ Depth dependent\';
location_of_sky_scat_data='C:\Users\meraf\OneDrive\Desktop\Underwater Project
Extension\MC Simulation Data_ Depth dependent\Sklight Single scattering PSD
model\Skylight Scattering\';
which_sun_zenith_angle_you_want_to_have_data=desired_zenith_angle; %
%% Define the intrpolation azimuth and zenith angles
AZ_all=0:0.5:360;
ZEN_all=0:0.5:90;
for i=1:length(AZ_all)
   for j=1:length(ZEN_all)
      AZ_new_temp(i,j)=AZ_all(i);
      ZEN_new_temp(i,j)=ZEN_all(j);
   end
end
AZ_new_temp=AZ_new_temp';
ZEN_new_temp=ZEN_new_temp';
AZ_new=AZ_new_temp(:);
ZEN_new=ZEN_new_temp(:);
%%
which_zenith_angle_you_already_have_data=[floor(which_sun_zenith_angle_you_want
_to_have_data) ceil(which_sun_zenith_angle_you_want_to_have_data)];
%% %% Sunlight Scattering %% %% %%
if do_Sunlight_scat==1
%% Sunlight Scattering : read data for all Sunlight Scattering Simulations we want to
consider
scattering_events=sun_scattering_events_pass_down;
for i=1:length(which_zenith_angle_you_already_have_data)
   f_name=strcat(location_of_sun_scat_data,'S_147.65to477.7nm radius
Particle_',num2str(scattering_events),'scat_non-
normalized_Z',num2str(which_zenith_angle_you_already_have_data(i)),'_A246.txt');
   Sunlight_scat_sim_collection{i,1}=dlmread(f_name,'',1,0);
   Sunlight_scat_sim_collection{i,1}=sortrows(Sunlight_scat_sim_collection{i,1},1);
```

```
end
%% Interpolate all data to have the same azimuth and zenith angles of measured data
for i=1:length(which_zenith_angle_you_already_have_data)
    AZ_Sun_Scat=Sunlight_scat_sim_collection{i,1}(:,1);
    ZEN_Sun_Scat=Sunlight_scat_sim_collection{i,1}(:,2);
    unnorm_I_Sun_Scat=Sunlight_scat_sim_collection{i,1}(:,3);
    unnorm_Q_Sun_Scat=Sunlight_scat_sim_collection{i,1}(:,4);
    unnorm_U_Sun_Scat=Sunlight_scat_sim_collection{i,1}(:,5);
    unnorm_V_Sun_Scat=Sunlight_scat_sim_collection{i,1}(:,6);
    F_unnorm_I_Sun_Scat                                                    =
scatteredInterpolant(AZ_Sun_Scat,ZEN_Sun_Scat,unnorm_I_Sun_Scat,'nearest','nearest'
);
    F_unnorm_Q_Sun_Scat                                                    =
scatteredInterpolant(AZ_Sun_Scat,ZEN_Sun_Scat,unnorm_Q_Sun_Scat,'nearest','nearest
');
    F_unnorm_U_Sun_Scat                                                    =
scatteredInterpolant(AZ_Sun_Scat,ZEN_Sun_Scat,unnorm_U_Sun_Scat,'nearest','nearest
');
    F_unnorm_V_Sun_Scat                                                    =
scatteredInterpolant(AZ_Sun_Scat,ZEN_Sun_Scat,unnorm_V_Sun_Scat,'nearest','nearest
');
    new_unnorm_I_Sun_Scat_at_current_Sun_Zenith(:,i)=
F_unnorm_I_Sun_Scat(AZ_new,ZEN_new);
    new_unnorm_Q_Sun_Scat_at_current_Sun_Zenith(:,i)=
F_unnorm_Q_Sun_Scat(AZ_new,ZEN_new);
    new_unnorm_U_Sun_Scat_at_current_Sun_Zenith(:,i)=
F_unnorm_U_Sun_Scat(AZ_new,ZEN_new);
    new_unnorm_V_Sun_Scat_at_current_Sun_Zenith(:,i)=
F_unnorm_V_Sun_Scat(AZ_new,ZEN_new);
    clear AZ_Sun_Scat ZEN_Sun_Scat unnorm_I_Sun_Scat unnorm_Q_Sun_Scat
unnorm_U_Sun_Scat unnorm_V_Sun_Scat F_unnorm_I_Sun_Scat
F_unnorm_Q_Sun_Scat F_unnorm_U_Sun_Scat F_unnorm_V_Sun_Scat
end
%% linear fitting to get data at desired sun zenith angle
new_unnorm_I_data_at_Desired_sun_ZEN=fit_to_new_sun_zenith(which_zenith_angle
_you_already_have_data,new_unnorm_I_Sun_Scat_at_current_Sun_Zenith,which_sun_z
enith_angle_you_want_to_have_data);
new_unnorm_Q_data_at_Desired_sun_ZEN=fit_to_new_sun_zenith(which_zenith_angle
_you_already_have_data,new_unnorm_Q_Sun_Scat_at_current_Sun_Zenith,which_sun_
zenith_angle_you_want_to_have_data);
new_unnorm_U_data_at_Desired_sun_ZEN=fit_to_new_sun_zenith(which_zenith_angle
_you_already_have_data,new_unnorm_U_Sun_Scat_at_current_Sun_Zenith,which_sun_
zenith_angle_you_want_to_have_data);
new_unnorm_V_data_at_Desired_sun_ZEN=fit_to_new_sun_zenith(which_zenith_angle
_you_already_have_data,new_unnorm_V_Sun_Scat_at_current_Sun_Zenith,which_sun_
zenith_angle_you_want_to_have_data);
```

```
end
%% %% %% %% Skylight Refraction %% %% %% %% %% %% %%
if do_Skylight_refract==1
%% Skylight Refraction: read data for Skylight Refraction Simulations we want to
consider
for i=1:length(which_zenith_angle_you_already_have_data)
   f_name=strcat(location_of_sky_ref_data,'Rayleigh_scat_Und_Watr_Simulation_Sun
Zenith-',num2str(which_zenith_angle_you_already_have_data(i)),'_Azimuth-246.txt');
   Skylight_refraction_sim_collection{i,1}=dlmread(f_name,'',1,0);
Skylight_refraction_sim_collection{i,1}=sortrows(Skylight_refraction_sim_collection{i,
1},1);
end
%% Interpolate all data to have the same azimuth and zenith angles of measured data
for i=1:length(which_zenith_angle_you_already_have_data)
   AZ_Skylight_refract=Skylight_refraction_sim_collection{i,1}(:,1);
   ZEN_Skylight_refract=Skylight_refraction_sim_collection{i,1}(:,2);
   unnorm_I_Skylight_refract=Skylight_refraction_sim_collection{i,1}(:,3);
   unnorm_Q_Skylight_refract=Skylight_refraction_sim_collection{i,1}(:,4);
   unnorm_U_Skylight_refract=Skylight_refraction_sim_collection{i,1}(:,5);
   unnorm_V_Skylight_refract=Skylight_refraction_sim_collection{i,1}(:,6);
   F_unnorm_I_Skylight_refract                                                    =
scatteredInterpolant(AZ_Skylight_refract,ZEN_Skylight_refract,unnorm_I_Skylight_refr
act,'nearest','nearest');
   F_unnorm_Q_Skylight_refract                                                    =
scatteredInterpolant(AZ_Skylight_refract,ZEN_Skylight_refract,unnorm_Q_Skylight_ref
ract,'nearest','nearest');
   F_unnorm_U_Skylight_refract                                                    =
scatteredInterpolant(AZ_Skylight_refract,ZEN_Skylight_refract,unnorm_U_Skylight_ref
ract,'nearest','nearest');
   F_unnorm_V_Skylight_refract                                                    =
scatteredInterpolant(AZ_Skylight_refract,ZEN_Skylight_refract,unnorm_V_Skylight_ref
ract,'nearest','nearest');
   new_unnorm_I_Skylight_refract_at_current_Sun_Zenith(:,i)=
F_unnorm_I_Skylight_refract(AZ_new,ZEN_new);
   new_unnorm_Q_Skylight_refract_at_current_Sun_Zenith(:,i)=
F_unnorm_Q_Skylight_refract(AZ_new,ZEN_new);
   new_unnorm_U_Skylight_refract_at_current_Sun_Zenith(:,i)=
F_unnorm_U_Skylight_refract(AZ_new,ZEN_new);
   new_unnorm_V_Skylight_refract_at_current_Sun_Zenith(:,i)=
F_unnorm_V_Skylight_refract(AZ_new,ZEN_new);
   clear AZ_Skylight_refract ZEN_Skylight_refract unnorm_I_Skylight_refract
unnorm_Q_Skylight_refract unnorm_U_Skylight_refract unnorm_V_Skylight_refract
F_unnorm_I_Skylight_refract                                F_unnorm_Q_Skylight_refract
F_unnorm_U_Skylight_refract F_unnorm_V_Skylight_refract
end
%% linear fitting to get data at desired sun zenith angle
```

```
new_unnorm_I_data_at_Desired_sun_ZEN=fit_to_new_sun_zenith(which_zenith_angle
_you_already_have_data,new_unnorm_I_Skylight_refract_at_current_Sun_Zenith,which
_sun_zenith_angle_you_want_to_have_data);
new_unnorm_Q_data_at_Desired_sun_ZEN=fit_to_new_sun_zenith(which_zenith_angle
_you_already_have_data,new_unnorm_Q_Skylight_refract_at_current_Sun_Zenith,whic
h_sun_zenith_angle_you_want_to_have_data);
new_unnorm_U_data_at_Desired_sun_ZEN=fit_to_new_sun_zenith(which_zenith_angle
_you_already_have_data,new_unnorm_U_Skylight_refract_at_current_Sun_Zenith,whic
h_sun_zenith_angle_you_want_to_have_data);
new_unnorm_V_data_at_Desired_sun_ZEN=fit_to_new_sun_zenith(which_zenith_angle
_you_already_have_data,new_unnorm_V_Skylight_refract_at_current_Sun_Zenith,whic
h_sun_zenith_angle_you_want_to_have_data);
end
%% %% %% %% Sunlight Scattering %% %% %% %% %% %%
if do_Skylight_scat==1
%% Skylight Scattering : read data for all Sunlight Scattering Simulations we want to
consider
for i=1:length(which_zenith_angle_you_already_have_data)
   f_name=strcat(location_of_sky_scat_data,'Skylight_scat_underwater_MFP-
0.36947m_147.65to477.7nm
part_Z',num2str(which_zenith_angle_you_already_have_data(i)),'_A246_ang_res-2.txt');
   Skylight_scat_sim_collection{i,1}=dlmread(f_name,'',1,0);
   Skylight_scat_sim_collection{i,1}=sortrows(Skylight_scat_sim_collection{i,1},1);
end
%% Interpolate all data to have the same azimuth and zenith angles of measured data
for i=1:length(which_zenith_angle_you_already_have_data)
   AZ_Sky_Scat=Skylight_scat_sim_collection{i,1}(:,1);
   ZEN_Sky_Scat=Skylight_scat_sim_collection{i,1}(:,2);
   unnorm_I_Sky_Scat=Skylight_scat_sim_collection{i,1}(:,3);
   unnorm_Q_Sky_Scat=Skylight_scat_sim_collection{i,1}(:,4);
   unnorm_U_Sky_Scat=Skylight_scat_sim_collection{i,1}(:,5);
   unnorm_V_Sky_Scat=Skylight_scat_sim_collection{i,1}(:,6);
   F_unnorm_I_Sky_Scat                                                    =
scatteredInterpolant(AZ_Sky_Scat,ZEN_Sky_Scat,unnorm_I_Sky_Scat,'nearest','nearest'
);
   F_unnorm_Q_Sky_Scat                                                    =
scatteredInterpolant(AZ_Sky_Scat,ZEN_Sky_Scat,unnorm_Q_Sky_Scat,'nearest','nearest
');
   F_unnorm_U_Sky_Scat                                                    =
scatteredInterpolant(AZ_Sky_Scat,ZEN_Sky_Scat,unnorm_U_Sky_Scat,'nearest','nearest
');
   F_unnorm_V_Sky_Scat                                                    =
scatteredInterpolant(AZ_Sky_Scat,ZEN_Sky_Scat,unnorm_V_Sky_Scat,'nearest','nearest
');
   new_unnorm_I_Sky_Scat_at_current_Sun_Zenith(:,i)=
F_unnorm_I_Sky_Scat(AZ_new,ZEN_new);
```

```
   new_unnorm_Q_Sky_Scat_at_current_Sun_Zenith(:,i)=
F_unnorm_Q_Sky_Scat(AZ_new,ZEN_new);
   new_unnorm_U_Sky_Scat_at_current_Sun_Zenith(:,i)=
F_unnorm_U_Sky_Scat(AZ_new,ZEN_new);
   new_unnorm_V_Sky_Scat_at_current_Sun_Zenith(:,i)=
F_unnorm_V_Sky_Scat(AZ_new,ZEN_new);
   clear AZ_Sky_Scat ZEN_Sky_Scat unnorm_I_Sky_Scat unnorm_Q_Sky_Scat
unnorm_U_Sky_Scat unnorm_V_Sky_Scat F_unnorm_I_Sky_Scat
F_unnorm_Q_Sky_Scat F_unnorm_U_Sky_Scat F_unnorm_V_Sky_Scat
end
%% linear fitting to get data at desired sun zenith angle
new_unnorm_I_data_at_Desired_sun_ZEN=fit_to_new_sun_zenith(which_zenith_angle
_you_already_have_data,new_unnorm_I_Sky_Scat_at_current_Sun_Zenith,which_sun_z
enith_angle_you_want_to_have_data);
new_unnorm_Q_data_at_Desired_sun_ZEN=fit_to_new_sun_zenith(which_zenith_angle
_you_already_have_data,new_unnorm_Q_Sky_Scat_at_current_Sun_Zenith,which_sun_
zenith_angle_you_want_to_have_data);
new_unnorm_U_data_at_Desired_sun_ZEN=fit_to_new_sun_zenith(which_zenith_angle
_you_already_have_data,new_unnorm_U_Sky_Scat_at_current_Sun_Zenith,which_sun_
zenith_angle_you_want_to_have_data);
new_unnorm_V_data_at_Desired_sun_ZEN=fit_to_new_sun_zenith(which_zenith_angle
_you_already_have_data,new_unnorm_V_Sky_Scat_at_current_Sun_Zenith,which_sun_
zenith_angle_you_want_to_have_data);
end
simulated_data_temp=[AZ_new ZEN_new new_unnorm_I_data_at_Desired_sun_ZEN
new_unnorm_Q_data_at_Desired_sun_ZEN new_unnorm_U_data_at_Desired_sun_ZEN
new_unnorm_V_data_at_Desired_sun_ZEN];
%%
function [new]=get_maping_at_specific_zenith(data,which_zenith,res)
   k_k=find(abs(data(:,2)-which_zenith)<res);
   new(:,:)=data(k_k,:); %Azimuth
end

function
[new_AoP_data_at_Desired_ZEN]=fit_to_new_sun_zenith(which_zenith_angle_you_alr
eady_have_data,new_AoP_MC,which_sun_zenith_angle_you_want_to_have_data)
aaa = new_AoP_MC(:,2)-new_AoP_MC(:,1);
bbb = new_AoP_MC(:,1)- aaa.*which_zenith_angle_you_already_have_data(1);
new_AoP_data_at_Desired_ZEN=aaa*which_sun_zenith_angle_you_want_to_have_data
+bbb;
end
end
```

APPENDIX F

MATLAB CODE FOR MONTE CARLO SIMULATION OF UNDERWATER LIGHT POLARIZATION

```
clc
clear all
close all
mfp_all=[0.4];
theta_all=[0:1:90];
phi_all=246;
for theta_c=1:length(theta_all)
for imfp=1:length(mfp_all)
tic
save_raw_data=0;
save_result=0;
save_pecent_photons_vs_no_of_scat=1;
location_of_saving_data='C:\Users\meraf\OneDrive\Desktop\Underwater Project Extension\MC Simulation Data_ Depth dependent\Scatter probability\';
scat_num=1;
plot_2d_figures_of_full_map=0;
plot_2d_plot_of_maping_at_one_zenith=0;
plot_polar_plot_at_one_zenith_angle=0;
plot_polar_plot_of_full_map=0;
%% Simulate at this wavelength
lamda=530e-9; %wavelength in nm
n_air=1;
%% Number of photons
N=200e3;
%% Maximum number of scattering
maximum_number_of_scattering=30+1; %last one is discarded as it will contain all the photons that would have gone through more than the indicated maximum scattering events

%% Scatter Particle properties and medium properties
par_size_and_PSD=dlmread('E:\Research\Projects\Underwater Geolocation\Water turbidiy and particle size\Sample Town Lake July 26 2021\particle distribution for simulation-072621.txt',',',1,0);
% %from lab measurement
par_size_and_PSD(:,2)=par_size_and_PSD(:,2)/max(par_size_and_PSD(:,2))*2.0513e12
;
particle_size=par_size_and_PSD(:,1)';
PSD=par_size_and_PSD(:,2)'; %in m-3
PSD_probabilty=PSD/sum(PSD);
% n_medium is the refractive index of that medium
n_medium=1.3333;

%Refractive Index of scatterer
index_of_scatter=1.5721; % Polystyrene sphere
for i=1:length(particle_size)
%Particle size
radius=particle_size(i)/2 %in nm
```

```
a=radius*1e-9; %radius in m
k0=2*pi/lamda;
%Size Parameter
x=k0*a*n_medium; %Size parameter
%Refractive index mismatch ratio
n_r=index_of_scatter/n_medium;
%Geometrical Cross-sectional area of the particles
A=pi*a^2; %m^2
%Concentration or volume density or number density of particles in the medium
p_s=PSD(i); %in m-3
%Scattering efficiency
Res_temp= mie(n_r,x); %Run the mie theory algorithm
Qs=Res_temp(5); %Scattering efficiency
Qa=Res_temp(6); %Absorption efficiency
%Effective scattering cross-section or scattering cross section
sigma_s=Qs*A;
%Effective absorption cross-section or absorption cross section
sigma_a=Qa*A;
%Scattering coefficient
us(i)= p_s*sigma_s;
%Absorption coefficient
ua(i)= p_s*sigma_a;
%Attenuation coefficient
ut(i)=ua(i)+us(i); %attenuation coefficient in m^-1
end
%Total scattering coefficient
uS=sum(us)
%Total attenuation coefficient
uT=sum(ut)
MFP=1/uT % Mean free path of the medium in m
%% Where is the detector?
depth=mfp_all(imfp)*1.00000*MFP; %%in m (meters) % 1 inch = 0.0254 m
initial_d=0;
phi_i=phi_all;
theta_sun=theta_all(theta_c);
   % Sun position in (x,y,z) coordinate
   x_ii=sind(theta_sun)*cosd(phi_i)
   y_ii=sind(theta_sun)*sind(phi_i)
   z_ii=cosd(theta_sun);
%% Transmission/Refraction in air-water interface %%% Beam entering Water
n1=n_air; %Air refractive index
n2=n_medium; %Water refractive index
theta_t=asind(n1*sind(theta_sun)/n2); %Snell's law
t_s=    2*n1*cosd(theta_sun)./(n1*cosd(theta_sun)+n2*cosd(theta_t));    %Perpendicular
component of transmitted light
```

```
t_p=        2*n1*cosd(theta_sun)./(n2*cosd(theta_sun)+n1*cosd(theta_t));        %Parallel component of transmitted light
M_r11=t_s.^2+t_p.^2;
M_r12=t_s.^2-t_p.^2;
M_r21=M_r12;
M_r22=M_r11;
M_r33=2.*t_s.*t_p;
M_r34=0;
M_r43=0;
M_r44=M_r33;
%%%Transmission Mueller matrix
M_r=(1/2)*[M_r11 M_r12 0 0; M_r21 M_r22 0 0; 0 0 M_r33 M_r34; 0 0 M_r43 M_r44];
theta_i=theta_t;
Sii=[1 0 0 0]'; % S_initial=[I Q U V]; Stokes vector for unpolarized light S_i=[1 0 0 0]
Sii=M_r*Sii; %After entering water
DoP_i_i=sqrt(Sii(2)^2+Sii(3)^2+Sii(4)^2)/Sii(1)
dc=1;
discard_count=[]; % count the number of photons that didn't reach the detector or the depth at which the detector is at
esc=1;
escaped_photon_count=[]; % count the number of photons that escaped from water
Reflection_flag=zeros(N,1);
TIR_flag=zeros(N,1);
parfor j=1:N
   %for unit r => r=1
   r1=(depth-initial_d)/cosd(theta_i);
   x_i=r1*sind(theta_i)*cosd(phi_i);
   y_i=r1*sind(theta_i)*sind(phi_i);
   z_i=r1*cosd(theta_i);
   S_i=[1 0 0 0]'; % S_initial=[I Q U V]; Stokes vector for unpolarized light S_i=[1 0 0 0]
   S_i=M_r*S_i; %After entering water
   r=sqrt(x_i^2+y_i^2+z_i^2);
   %initial photon direction cosine
   ux=x_i/r;
   uy=y_i/r;
   uz=z_i/r;

   S_new=[0 0 0 0]';
   xyz_init=[x_i y_i z_i];
   uxyz_init=[ux uy uz];
   ds=1; %counting delta_s storage
   xyzc=1;
for i=1:maximum_number_of_scattering
   %% Select particle according to the probability set by PSD and calculate size parameter
   rand_particle_select=sum(rand >= cumsum([0,PSD_probabilty]));% Selects a number between 1 to n(number of particles) based on the probability defined by  PSD
```

```
    particle_radius=particle_size(rand_particle_select)/2; %in nm
    a=particle_radius*1e-9; %particle radius in m
    %Size Parameter
    x=k0*a*n_medium; %Size parameter
    %Refractive index mismatch ratio
    n_r=index_of_scatter/n_medium;
%%
flag1=0;
flag2=0;
    %% Move
    a1=0;
    b1=1;
    zeta=a1+(b1-a1)*rand(1); %psudo random number within 0 to 1
    delta_s=-log(zeta)/uT; %traversed path or pathlength before being scattered/absorbed
    x_old=[x_i y_i z_i];
    x_new=x_i+ux*delta_s;
    y_new=y_i+uy*delta_s;
    z_new=z_i+uz*delta_s;
    %%%Check if the photon still inside water or not
    Refraction_flag=0;
    if z_new>=depth % Is the photon going out of water?
        %new Move step
        %Distance from previous point to the water-air interface
        delta_s_1=abs((z_i-depth)/uz);
        %Retract the full step it took
        x_new=x_new-ux*delta_s;
        y_new=y_new-uy*delta_s;
        z_new=z_new-uz*delta_s;
        xyz_start=[x_new y_new z_new]; %Start position of photon before it went out of
water
        %new position of photon puts it at the interface of water-air
        x_new=x_new+ux*delta_s_1;
        y_new=y_new+uy*delta_s_1;
        z_new=z_new+uz*delta_s_1;

        xyz_water_air_interf=[x_new y_new z_new];
        %%%%Save the position of photon on the water-air interface ; For the movie.
        %Does the photon refracts or goes through TIR? It depends on the incident angle and
critical angle
        [theta_check,phi_check]=uxuyuz2thetaphi(ux,uy,uz);
        theta_critical=asind(n_air/n_medium);
theta_check;
        if theta_check>=theta_critical %Is the incident angle of photon on water-air interface
larger than critical angle?
            %TIR  %Calculate M_TIR and Multiply with S_i[150]
            n2=n_air;
```

```
        n1=n_medium;
        n=n1/n2;
        del_s=((2*atand(sqrt(n^2*(sind(theta_check))^2-1)/n/cosd(theta_check))));%s
polarization
        del_p=((2*atand(n*sqrt(n^2*(sind(theta_check))^2-1)/cosd(theta_check))));    %p
polarization
        del=del_s-del_p;
        M_TIR=[
              1 0 0 0
              0 1 0 0
              0 0 cosd(del) -sind(del)
              0 0 sind(del) cosd(del)];
        S_i=M_TIR*S_i;
        if TIR_flag(j,1)==0
           TIR_flag(j,1)=1;
        elseif TIR_flag(j,1)~=0
           TIR_flag(j,1)=TIR_flag(j,1)+1;
        end
      elseif theta_check<theta_critical
        %Refraction
        %Calculate M_refrac and Multiply with S_i[151]
        n2=n_air; %Refractive index of the reflective medium
        n1=n_medium; %Refractive index of the incident medium
        theta_inc=theta_check;
        theta_trans=asind(n1*sind(theta_check)/n2); %Snell's law
        rs=(n1*cosd(theta_check)-
n2*cosd(theta_trans))./(n1*cosd(theta_inc)+n2*cosd(theta_trans));
        rp=(n2*cosd(theta_check)-
n1*cosd(theta_trans))./(n2*cosd(theta_inc)+n1*cosd(theta_trans));
        M_ref_11=rs^2+rp^2;
        M_ref_12=rp^2-rs^2;
        M_ref_21=M_ref_12;
        M_ref_22=M_ref_11;
        M_ref_33=2.*rs.*rp;
        M_ref_44=2.*rs.*rp;
        M_reflect=(1/2)*[
                    M_ref_11 M_ref_12 0 0
                    M_ref_21 M_ref_22 0 0
                    0 0 M_ref_33 0
                    0 0 0 M_ref_44];
        S_i=M_reflect*S_i;
        if Reflection_flag(j,1)==0
           Reflection_flag(j,1)=1;
        elseif Reflection_flag(j,1)~=0
           Reflection_flag(j,1)=Reflection_flag(j,1)+1;
        end
```

```
            S_i_1=S_i(1);
            S_i=S_i./S_i(1);
        end
        %Calculate new direction cosines
        %ux and uy stays the same
        uz=-uz; %trajectory is reversed with respect to z-axis
        %Calculate New move step
        delta_s=delta_s-delta_s_1;
        %Calculate New position based on new direction cosine and move step
        x_new=x_new+ux*delta_s;
        y_new=y_new+uy*delta_s;
        z_new=z_new+uz*delta_s;
        xyz_after_refraction=[x_new y_new z_new];
        Refraction_flag=1;
    end
    xyz_new=[x_new y_new z_new];
    r_new=sqrt(x_new^2+y_new^2+z_new^2); %new r
    uxyz_old=[ux uy uz];
    ux=x_new/r_new;
    uy=y_new/r_new;
    uz=z_new/r_new;
    %%If the Photons reaches close to the detector within these multiple scattering events,
It will be collected.
    %%Also, it can be collected right after the "Move". So, I break from
    %%multiple scattering and proceed to next photon at this point
    if z_new<=0 %Is the photon close to photodetector? Detector plane is at Z=0
        flag1=1;
        %Distance from last point to the detector at Z=0
        if Refraction_flag==1
            delta_s_2=abs((xyz_water_air_interf(3))/uxyz_old(3));
        else
            delta_s_2=abs((z_i)/uxyz_old(3));
        end
        %Retract the full step it took
        x_new=x_new-uxyz_old(1)*delta_s;
        y_new=y_new-uxyz_old(2)*delta_s;
        z_new=z_new-uxyz_old(3)*delta_s;
        xyz_bef_detector=[x_new y_new z_new];
        %new position of photon puts it on the detector plane z=0
        x_new=x_new+uxyz_old(1)*delta_s_2;
        y_new=y_new+uxyz_old(2)*delta_s_2;
        z_new=z_new+uxyz_old(3)*delta_s_2;
        xyz_on_detector_plane=[x_new y_new z_new];
        r_new=sqrt(x_new^2+y_new^2+z_new^2); %new r
        ux=uxyz_old(1);
        uy=uxyz_old(2);
```

```
        uz=uxyz_old(3);
        ux_new=ux;
        uy_new=uy;
        uz_new=uz;
        if Reflection_flag(j,1)~=0
            S_new=S_i*S_i_1;
        else
            S_new=S_i;
        end
        S_i=S_i./S_i(1);
        no_of_scat(j)=i-1;   % it has probability to produce non-scattering event
        break; %If this photon has reached the detector (got collected) then we have to launch
another photon
    end
    u_old=[ux uy uz];

    %% Rejection method and choosing alpha and beta
    flag=0;
    while flag==0
        P_rand=0+(1-0)*rand(1);
        a_rand=0+(180-0)*rand(1);
        b_rand=0+(360-0)*rand(1);
        I0=S_i(1);
        Q0=S_i(2);
        U0=S_i(3);
        %%
        scattering_angle=a_rand;
        u=cos(pi/180*scattering_angle);
        % S1 and S2
        result_s1_s2=mie_S12(n_r, x, u);
        S1=result_s1_s2(1);
        S2=result_s1_s2(2);
        % Mie Scattering Matrix
        s11=1/2*(abs(S1)^2+abs(S2)^2);
        s12=1/2*(abs(S2)^2-abs(S1)^2);
        %%
        P_a_b=s11'*I0+s12'.*(Q0*cosd(2*b_rand)+U0*sind(2*b_rand));
        if P_rand<P_a_b
            alpha=a_rand;
            beta=b_rand;
            flag=1;
        end
    end
    M11=s11;
    M12=s12;
    M33=1/2*(S2*conj(S1)+S1*conj(S2));
```

```
    M34=-1/2*(S1*conj(S2)-S2*conj(S1))*1i;
    M=[M11 M12 0 0
       M12 M11 0 0
       0 0 M33 M34
       0 0 -M34 M33];
    %% Photon Scattering and rotation
    rand_cap_phi=0+(1-0)*rand(1);
    capital_phi=360*rand_cap_phi;
    [u_new]=new_direction_cosine(ux,uy,uz,alpha,capital_phi);
    ux_new=u_new(1);
    uy_new=u_new(2);
    uz_new=u_new(3);
    [theta_temp1,phi_temp1]=uxuyuz2thetaphi(ux_new,uy_new,uz_new);
[beta,gamma,R_beta,R_gamma]=get_rotation_matrix(phi_i,theta_i,phi_temp1,theta_tem
p1,alpha);
    S_new=R_gamma*M*R_beta*S_i;
    %update stokes vector
    S_i_for_test=S_new./S_new(1);
    % Check if all of the skoes parameters adds up to less than 1
    s_test(j,:)=sqrt(S_i_for_test(2)^2+S_i_for_test(3)^2+S_i_for_test(4)^2);
    %% Drop
    w=(uS/uT)^i; %weight of the photon after ith scattering; Mainly affects the results when
dealing with absorbing medium or particles
    S_i=S_new;
    S_i=real(S_i)*w; %%%Check
    S_i=S_i./S_i(1); %After each scattering event loop I need to normalize the soteks
parameters otherwise it will become very large and it will make the mie codes very slow.
    ux_old=ux;
    uy_old=uy;
    uz_old=uz;
    %update reference plane: Update the direction cosine and position of photon
    ux=ux_new;
    uy=uy_new;
    uz=uz_new;
    %How do i claculate the position from ux,uy,uz?
    x_i=x_new;
    y_i=y_new;
    z_i=z_new;
end
if i==maximum_number_of_scattering && flag1==0 %scattering reached the maximum
number of scattering and photon didn't reach the detector
    no_of_scat(j)=maximum_number_of_scattering;
end
S_i=S_i./S_i(1); %Normalized Stokes parameters
S_new; %Non-normalized Stokes parameters
S_new_f(j,:)=S_new;
```

```
S_f(j,:)=S_i;
ux_f(j,1)=ux;
uy_f(j,1)=uy;
uz_f(j,1)=uz;
x_f(j,1)=x_i;
y_f(j,1)=y_i;
z_f(j,1)=z_i;
DoLP_T(j,1)=sqrt(S_i(2).^2+S_i(3).^2)./S_i(1);
DoCP_T(j,1)=S_i(4)./S_i(1);
AoP_T(j,1)=0.5*myatand_0to180(S_i(3),S_i(2));
%Calculate the Zenith and Azimuth angle of the photon at the end of it journey
[theta_f(j),phi_f(j)]=uxuyuz2thetaphi(ux,uy,uz);
phi_f(j);
theta_f(j);
if ~mod(j,10000)
   disp(strcat('Simulating photon number##',num2str(j)))
end
end
%%
S_f_T=[x_f y_f z_f S_f]; %Stokes parameters at different position at the detector plane
S_new_f_T=[phi_f(:) theta_f(:) S_new_f]; %Un-normalized Stokes parameters at different
radial position at the detector plane
final_res=[phi_f(:) theta_f(:) S_f DoLP_T DoCP_T AoP_T]; %final results including all
the radial positions, normalized S, DoLP, DoCP and AoP.
%%
if save_raw_data==1
f_name=strcat(location_of_saving_data,'final_result_',num2str(particle_size(1)/2),'to',nu
m2str(particle_size(end)/2),'nm radius Particle_',num2str(N/1e6),'million photons.txt');
   fid = fopen(f_name, 'w');
   results_for_saving1=final_res;
   fprintf(fid, 'AZ ZEN I Q U V DoLP DoCP AoP\n');
   save(f_name,'results_for_saving1','-ascii','-append')
   fclose(fid);
f_name=strcat(location_of_saving_data,'S_',num2str(particle_size(1)/2),'to',num2str(parti
cle_size(end)/2),'nm radius Particle_',num2str(scat_num),'scat_non-normalized.txt');
   fid = fopen(f_name, 'w');
   results_for_saving2=S_new_f_T;
   fprintf(fid, 'AZ ZEN I Q U V\n');
   save(f_name,'results_for_saving2','-ascii','-append')
   fclose(fid);
f_name=strcat(location_of_saving_data,'S_',num2str(particle_size(1)/2),'to',num2str(parti
cle_size(end)/2),'nm radius Particle_',num2str(scat_num),'scat_normalized.txt');
   fid = fopen(f_name, 'w');
   results_for_saving3=S_f;
   fprintf(fid, 'x_f y_f z_f I Q U V\n');
   save(f_name,'results_for_saving3','-ascii','-append')
```

```
    fclose(fid);
f_name=strcat(location_of_saving_data,'no_of_scat_',num2str(particle_size(1)/2),'to',num
2str(particle_size(end)/2),'nm radius Particle_',num2str(scat_num),'.txt');
    fid = fopen(f_name, 'w');
    results_for_saving4=no_of_scat;
    fprintf(fid, 'no of scattering for each photon\n');
    save(f_name,'results_for_saving4','-ascii','-append')
    fclose(fid);
end
%% Seperate the photons w.r.t different scattering events
single_scat_results=final_res(find(no_of_scat==1),:);
two_scat_results=final_res(find(no_of_scat==2),:);
three_scat_results=final_res(find(no_of_scat==3),:);
%%Discard the zenith angles larger than 90.
    count_larger_90=1;
    out_of_water=[];
    for i=1:size(final_res,1)
        if (final_res(i,2)>90)
            out_of_water(count_larger_90)=i;
            count_larger_90=count_larger_90+1;
        end
    end

    if ~isempty(out_of_water)
        S_f_T(out_of_water,:)=[];
        final_res(out_of_water,:)=[];
        S_new_f_T(out_of_water,:)=[];
    end
%%
if save_result==1
f_name=strcat(location_of_saving_data,'S_',num2str(particle_size(1)/2),'to',num2str(parti
cle_size(end)/2),'nm radius Particle_',num2str(scat_num),'scat_non-normalized.txt');
    fid = fopen(f_name, 'w');
    results_for_saving=sortrows(S_new_f_T,1);
    fprintf(fid, 'AZ ZEN I Q U V\n');
    save(f_name,'results_for_saving','-ascii','-append')
    fclose(fid);
end
toc/60
disp('minutes')
if plot_2d_figures_of_full_map==1
    figure
    scatter(final_res(:,1),final_res(:,7))
    ylabel('DoLP')
    xlabel('Azimuth')
    set(gca, 'FontSize', 18);
```

```
    set(gcf,'color','w');
    figure
    scatter(final_res(:,1),final_res(:,8))
    ylabel('DoCP')
    xlabel('Azimuth')
    set(gca, 'FontSize', 18);
    set(gcf,'color','w');
    figure
    scatter(final_res(:,1),final_res(:,9))
    ylabel('AoP')
    xlabel('Azimuth')
    set(gca, 'FontSize', 18);
    set(gcf,'color','w');
    figure
    scatter(final_res(:,1),final_res(:,4))
    ylabel('Q')
    xlabel('Azimuth')
    set(gca, 'FontSize', 18);
    set(gcf,'color','w');

    figure
    scatter(final_res(:,1),final_res(:,5))
    ylabel('U')
    xlabel('Azimuth')
    set(gca, 'FontSize', 18);
    set(gcf,'color','w');
end

if plot_2d_plot_of_maping_at_one_zenith==1
    angle_resolution=1 %in degree
    which_zenith=25; %in degree
    k_k=find(abs(final_res(:,2)-which_zenith)<angle_resolution);
    az_at_zen=final_res(k_k,1);
    AoP_sim_at_zen=final_res(k_k,9);
    DoLP_sim_at_zen=final_res(k_k,7);
    DoCP_sim_at_zen=final_res(k_k,8);

    figure
    scatter(az_at_zen,AoP_sim_at_zen)
    ylabel('AoP')
    xlabel('Azimuth')
    set(gca, 'FontSize', 18);
    set(gcf,'color','w');
    figure
    scatter(az_at_zen,DoLP_sim_at_zen)
    ylabel('DoLP')
```

```
    xlabel('Azimuth')
    set(gca, 'FontSize', 18);
    set(gcf,'color','w');
    figure
    scatter(az_at_zen,DoCP_sim_at_zen)
    ylabel('DoCP')
    xlabel('Azimuth')
    set(gca, 'FontSize', 18);
    set(gcf,'color','w');
end
%% Ploar plot at one Zenith angle
if plot_polar_plot_at_one_zenith_angle==1
    figure
    ps=polarscatter(pi/180*az_at_zen,DoLP_sim_at_zen);
    title('DoLP')
    set(gca, 'FontSize', 18);
    set(gcf,'color','w');
    ps.SizeData = 200;
    ps.MarkerFaceColor = 'red';
    ps.MarkerFaceAlpha = .5;

    figure
    ps=polarscatter(pi/180*az_at_zen,DoCP_sim_at_zen);
    title('DoCP')
    set(gca, 'FontSize', 18);
    set(gcf,'color','w');
    ps.SizeData = 200;
    ps.MarkerFaceColor = 'red';
    ps.MarkerFaceAlpha = .5;
end

%% Polar plot of the full map
if plot_polar_plot_of_full_map==1
ploar_contour_plot(final_res(:,1),final_res(:,2),final_res(:,7))
dolp_ll=0; %DoCP lower limit (ll)
dolp_ml=1; % DoCP max limit (ml)
dolp_step=0.2; %DoCP colorbar tick step
caxis([dolp_ll, dolp_ml]); %change colorbar limit
cbh=colorbar();
set(cbh,'YTick',[dolp_ll:dolp_step:dolp_ml]); % Chnage color bar ticks
ploar_contour_plot(final_res(:,1),final_res(:,2),final_res(:,8))
%%
ploar_contour_plot(final_res(:,1),final_res(:,2),final_res(:,8))
docp_ll=-.3; %DoCP lower limit (ll)
docp_ml=.3; % DoCP max limit (ml)
docp_step=0.15; %DoCP colorbar tick step
```

```
caxis([docp_ll, docp_ml]); %change colorbar limit
cbh=colorbar();
set(cbh,'YTick',[docp_ll:docp_step:docp_ml]); % Chnage color bar ticks
%% Averaging
del_theta=0.5;
del_phi=1;
input_data=[final_res(:,1:2) final_res(:,7:9)];
[ouput_data]=average_over_whole_data_deltheta_delphi(input_data,del_theta,del_phi);

% DoCP
ploar_contour_plot(ouput_data(:,1),ouput_data(:,2),ouput_data(:,4))
title('Avg DoCP')
ploar_contour_plot(ouput_data(:,1),ouput_data(:,2),ouput_data(:,4))
docp_ll=-.1; %DoCP lower limit (ll)
docp_ml=.1; % DoCP max limit (ml)
docp_step=0.05; %DoCP colorbar tick step
caxis([docp_ll, docp_ml]); %change colorbar limit
cbh=colorbar();
set(cbh,'YTick',[docp_ll:docp_step:docp_ml]); % Chnage color bar ticks
title('Avg DoCP')
end

for i=1:maximum_number_of_scattering % not considering the cases where the number of scattering excceds the max limit of scattering
   percent_photons_vs_no_of_scat(i,1)=(i-1); %number of scattering
   percent_photons_vs_no_of_scat(i,2)=length(find(no_of_scat==(i-1)))/N; % percent of photons
end

percent_of_photons_got_scattered=sum(percent_photons_vs_no_of_scat(:,2))

if save_pecent_photons_vs_no_of_scat==1
f_name=strcat(location_of_saving_data,'no of scat., part. radius-',num2str(particle_size(1)/2,3),'to',num2str(particle_size(end)/2,3),'nm_depth-',num2str(depth,3),'m','_max Part. dens.-',num2str(max(PSD)/1e12,3),'x10^12m^-3','_nMFP-',num2str(mfp_all(imfp)),'_SAZ-',num2str(phi_i),'_SZEN-',num2str(theta_sun),'.txt');
   if theta_sun<.1
      f_name=strcat(location_of_saving_data,'no of scat., part. radius-',num2str(particle_size(1)/2,3),'to',num2str(particle_size(end)/2,3),'nm_depth-',num2str(depth,3),'m','_max Part. dens.-',num2str(max(PSD)/1e12,3),'x10^12m^-3','_nMFP-',num2str(mfp_all(imfp)),'_SAZ-',num2str(phi_i),'_SZEN-',num2str(floor(theta_sun)),'.txt');
   elseif theta_sun>89.9
      f_name=strcat(location_of_saving_data,'no of scat., part. radius-',num2str(particle_size(1)/2,3),'to',num2str(particle_size(end)/2,3),'nm_depth-
```

```
',num2str(depth,3),'m','_max Part. dens.-',num2str(max(PSD)/1e12,3),'x10^12m^-
3','_nMFP-',num2str(mfp_all(imfp)),'_SAZ-',num2str(phi_i),'_SZEN-
',num2str(ceil(theta_sun)),'.txt');
    end
    fid = fopen(f_name, 'w');
    fprintf(fid, 'No of scat Portion of photons\n');
    data_to_save=percent_photons_vs_no_of_scat;
    save(f_name,'data_to_save','-ascii','-append')
    fclose(fid);
end
end
end
figure
plot(percent_photons_vs_no_of_scat(:,1),100*percent_photons_vs_no_of_scat(:,2),'o-
','linewidth',2)
xlabel('No of scattering')
ylabel('(%) of photons')
set(gca, 'FontSize', 18);
set(gcf,'color','w');

figure
plot(particle_size,PSD,'o-','linewidth',2)
xlabel('Particle Size (nm)')
ylabel('Concentration(m^-^3)')
set(gca, 'FontSize', 18);
set(gcf,'color','w');

toc/60

function result = mie_S12(m, x, u)[71,159]
%Huffman (1983) BEWI:TDD122
% C. Mätzler, May 2002
nmax=round(2+x+4*x^(1/3));
abcd=mie_abcd(m,x);
an=abcd(1,:);
bn=abcd(2,:);
pt=mie_pt(u,nmax);
pin =pt(1,:);
tin=pt(2,:);
n=(1:nmax);
n2=(2*n+1)./(n.*(n+1));
pin=n2.*pin;
tin=n2.*tin;
S1=(an*pin'+bn*tin');
S2=(an*tin'+bn*pin');
result=[S1;S2];
```

```
end

function result=mie_pt(u,nmax) [71,159]
p(1)=1;
t(1)=u;
p(2)=3*u;
t(2)=3*cos(2*acos(u));
for n1=3:nmax
p1=(2*n1-1)./(n1-1).*p(n1-1).*u;
p2=n1./(n1-1).*p(n1-2);
p(n1)=p1-p2;
t1=n1*u.*p(n1);
t2=(n1+1).*p(n1-1);
t(n1)=t1-t2;
end
result=[p;t];
end

function result = mie(m, x) [71,159]
if x==0 % To avoid a singularity at x=0
result=[real(m) imag(m) 0 0 0 0 0 0 1.5];
elseif x>0 % This is the normal situation
nmax=round(2+x+4*x^(1/3));
n1=nmax-1;
n=(1:nmax);cn=2*n+1; c1n=n.*(n+2)./(n+1); c2n=cn./n./(n+1);
x2=x*x;
f=mie_abcd(m,x);
anp=(real(f(1,:))); anpp=(imag(f(1,:)));
bnp=(real(f(2,:))); bnpp=(imag(f(2,:)));
g1(1:4,nmax)=[0; 0; 0; 0]; % displaced numbers used for
g1(1,1:n1)=anp(2:nmax); % asymmetry parameter, p. 120
g1(2,1:n1)=anpp(2:nmax);
g1(3,1:n1)=bnp(2:nmax);
g1(4,1:n1)=bnpp(2:nmax);
dn=cn.*(anp+bnp);
q=sum(dn);
qext=2*q/x2;
en=cn.*(anp.*anp+anpp.*anpp+bnp.*bnp+bnpp.*bnpp);
q=sum(en);
qsca=2*q/x2;
qabs=qext-qsca;
fn=(f(1,:)-f(2,:)).*cn;
gn=(-1).^n;
f(3,:)=fn.*gn;
q=sum(f(3,:));
qb=q*q'/x2;
```

```
asy1=c1n.*(anp.*g1(1,:)+anpp.*g1(2,:)+bnp.*g1(3,:)+bnpp.*g1(4,:));
asy2=c2n.*(anp.*bnp+anpp.*bnpp);
asy=4/x2*sum(asy1+asy2)/qsca;
qratio=qb/qsca;
result=[real(m) imag(m) x qext qsca qabs qb asy qratio];
end
end

function result = mie_tetascan(m, x, nsteps) [71,159]
nsteps=nsteps;
m1=real(m); m2=imag(m);
nx=(1:nsteps); dteta=pi/(nsteps-1);
teta=(nx-1).*dteta;
for j = 1:nsteps
u=cos(teta(j));
a(:,j)=mie_S12(m,x,u);
SL(j)= real(a(1,j)'*a(1,j));
SR(j)= real(a(2,j)'*a(2,j));
end
y=[teta teta+pi;SL SR(nsteps:-1:1)]';
polar(y(:,1),y(:,2))
title(sprintf('Mie angular scattering: m=%g+%gi, x=%g',m1,m2,x));
xlabel('Scattering Angle')
result=y;
end

function result = mie_abcd(m, x) [71,159]
nmax=round(2+x+4*x^(1/3));
n=(1:nmax); nu = (n+0.5); z=m.*x; m2=m.*m;
sqx= sqrt(0.5*pi./x); sqz= sqrt(0.5*pi./z);
bx = besselj(nu, x).*sqx;
bz = besselj(nu, z).*sqz;
yx = bessely(nu, x).*sqx;
hx = bx+1i*yx;
b1x=[sin(x)/x, bx(1:nmax-1)];
b1z=[sin(z)/z, bz(1:nmax-1)];
y1x=[-cos(x)/x, yx(1:nmax-1)];
h1x= b1x+1i*y1x;
ax = x.*b1x-n.*bx;
az = z.*b1z-n.*bz;
ahx= x.*h1x-n.*hx;
an = (m2.*bz.*ax-bx.*az)./(m2.*bz.*ahx-hx.*az);
bn = (bz.*ax-bx.*az)./(bz.*ahx-hx.*az);
cn = (bx.*ahx-hx.*ax)./(bz.*ahx-hx.*az);
dn = m.*(bx.*ahx-hx.*ax)./(m2.*bz.*ahx-hx.*az);
result=[an; bn; cn; dn];
```

```
end

function result = mie_xscan(m, nsteps, dx) [71,159]
nx=(1:nsteps)';
x=(nx-1)*dx;
for j = 1:nsteps
a(j,:)=Mie(m,x(j));
end
output_parameters='Real(m), Imag(m), x, Qext, Qsca, Qabs, Qb, <costeta>, Qb/Qsca'
m1=real(m);m2=imag(m);
plot(a(:,3),a(:,4:9)) % plotting the results
legend('Qext','Qsca','Qabs','Qb','<costeta>','Qb/Qsca')
title(sprintf('Mie Efficiencies, m=%g+%gi',m1,m2))
xlabel('x')
result=a;
end

function result = mie_Esquare(m, x, nj) [71,159]
nmax=round(2+x+4*x^(1/3));
n=(1:nmax); nu =(n+0.5);
m1=real(m); m2=imag(m);
abcd=mie_abcd(m,x);
cn=abcd(3,:);dn=abcd(4,:);
cn2=abs(cn).^2;
dn2=abs(dn).^2;
dx=x/nj;
for j=1:nj
xj=dx.*j;
z=m.*xj;
sqz= sqrt(0.5*pi./z);
bz = besselj(nu, z).*sqz; % This is jn(z)
bz2=(abs(bz)).^2;
b1z=[sin(z)/z, bz(1:nmax-1)]; % Note that sin(z)/z=j0(z)
az = b1z-n.*bz./z;
az2=(abs(az)).^2;
z2=(abs(z)).^2;
n1 =n.*(n+1);
n2 =2.*(2.*n+1);
mn=real(bz2.*n2);
nn1=az2;
nn2=bz2.*n1./z2;
nn=n2.*real(nn1+nn2);
en(j)=0.25*(cn2*mn'+dn2*nn');
end
xxj=[0:dx:xj]; een=[en(1) en];
plot(xxj,een);
```

```
legend('Radial Dependence of (abs(E))^2')
title(sprintf('Squared Amplitude E Field in a Sphere, m=%g+%gi x=%g',m1,m2,x))
xlabel('r k')
result=een;
end

function result = mie_abs(m, x) [71,159]
nj=5*round(2+x+4*x.^(1/3))+160;
e2=imag(m.*m);
dx=x/nj;
x2=x.*x;
nj1=nj+1;
xj=(0:dx:x);
en=mie_Esquare(m,x,nj);
en1=0.5*en(nj1).*x2; % End-Term correction in integral
enx=en*(xj.*xj)'-en1; % Trapezoidal radial integration
inte=dx.*enx;
Qabs=4.*e2.*inte./x2;
result=Qabs;
end

function [theta,phi] = sun_position(month,day,year,time,phi_lat,phi_lon,GMT,DLS)
if leapyear(year)
   day_month=[31 29 31 30 31 30 31 31 30 31 30 31];
else
   day_month=[31 28 31 30 31 30 31 31 30 31 30 31];
end
%calculate day number of the year (n)
if(month==1)
   n=day;
elseif(month==2)
   n=day+31;
elseif(month>=3)
   n=day+sum(day_month(1:month-1))+leapyear(year);
end
%Declination angle (independent of position on earth; dependent on n)
delta=23.45.*sind(360./365.*(284+n));
theta_S_c=90-delta; %Angle between sun ray and the NS direction
%%%Equation of time
B=360/365*(n-81);
EOT=0.165*sind(2*B)-0.126*cosd(B)-0.025*sind(B);
Lstd=15*GMT; %For mountain Standard Time in Arizona
Lloc=phi_lon; %Longitude of actual location
CT=time; %CLock time
LST=CT-(1/15)*(Lstd-Lloc)+EOT-DLS;
%%%Solar hour angle
```

```
omega=15.*(LST-12);
%Calculated from due south
theta=acosd(cosd(phi_lat).*cosd(omega).*cosd(delta)+sind(phi_lat).*sind(delta));
for i=1:length(time)
if theta(i)>90 %i.e sun is below the horrizon (equator)
   theta(i)=90-theta(i);
else
   theta(i)=theta(i);
end

if theta(i)>0 % i.e its sun in above horrizon (equator)
   phi(i)=acosd((cosd(delta).*sind(phi_lat).*cosd(omega(i))-
sind(delta).*cosd(phi_lat))./cosd(90-theta(i))); %Measured from due south
else
   phi(i)=acosd((cosd(delta).*sind(phi_lat).*cosd(omega(i))-
sind(delta).*cosd(phi_lat))./cosd(theta(i))); %Measured from due south
end
end
for i=1:length(time)
if omega(i)<0
   phi(i)=phi(i)*-1; %(bcz: phi: the sun's azimuth angle is negative east of south and
positive west of south )
end
end
phi;
% If phi is calculated from due north
phi_N_CW=phi+180; %if calculated From due north CW
phi=phi_N_CW;

% As sun can go below the horrizon this part is just to make the zenith positive or (0 to
180)
for i=1:length(time)
   if theta(i)<0
   theta(i)=90-theta(i);
   end
end
end

function [u_new]=new_direction_cosine(ux,uy,uz,alpha,capital_phi)
 u_s= cosd(alpha);
 u11=ux*uz/sqrt(1-uz^2);
 u12=-uy/sqrt(1-uz^2);
 u13=ux;
 u21=uy*uz/sqrt(1-uz^2);
 u22=ux/sqrt(1-uz^2);
 u23=uy;
```

```
  u31=-sqrt(1-uz^2);
  u32=0;
  u33=uz;
  m2=[sqrt(1-u_s^2)*cosd(capital_phi) sqrt(1-u_s^2)*sind(capital_phi) u_s]';
  if uz^2>0.99999
    u_new=sign(uz)*m2;
  else
    u_new=[u11 u12 u13; u21 u22 u23; u31 u32 u33]*m2;
  end
end

function [sigma, sigma_S, L_sigma,L_sigma_S]=get_rotation_matrix(phi,theta,phi_S,theta_S,mu)
phi_n=phi;
if phi>180
    phi=phi-180;
end
del_phi=phi_S-phi;
if ((mu ~= 0 || mu ~= 180) && ((0<=del_phi && del_phi<=180) && del_phi>=0))

    if (theta_S~=0 && theta~=0)
        sigma=acosd((cosd(theta_S)-cosd(theta)*cosd(mu))/(sind(mu)*sind(theta)));
        sigma_S=180+acosd((cosd(theta)-cosd(theta_S)*cosd(mu))/(sind(mu)*sind(theta_S)));
    end
    if theta_S==0 % we don't have the condition when phi_S==0 or 360
        sigma=0;
        sigma_S=acosd(cosd(theta)*cosd(phi_S-phi));
    end
    if theta==0
        sigma=acosd(cosd(theta_S)*cosd(phi_S-phi));
        sigma_S=0;
    end
    if (theta_S==0 && theta==0)
        sigma=(phi_S-phi);
    end
elseif ((mu ~= 0 || mu ~= 180) && ((del_phi>0 &&(180<del_phi && del_phi<360)) || del_phi<0))
    if (theta_S~=0 && theta~=0)
    sigma=180-acosd((cosd(theta_S)-cosd(theta)*cosd(mu))/(sind(mu)*sind(theta)));
    sigma_S=360-acosd((cosd(theta)-cosd(theta_S)*cosd(mu))/(sind(mu)*sind(theta_S)));
    end
    if ((phi_S==0 || phi_S==360) && theta_S==0)
        sigma=0;
        sigma_S=acosd(cosd(theta)*cosd(phi));
    end
```

```
    if ((phi_S~=0 || phi_S~=360) && theta_S==0)
        sigma=0;
        if phi_S>phi
            sigma_S=acosd(cosd(theta)*cosd(phi+(360-phi_S)));
        else
            sigma_S=acosd(cosd(theta)*cosd(phi-phi_S));
        end
    end
    if ((phi_S==0 || phi_S==360) && theta==0)
        sigma=acosd(cosd(theta_S)*cosd(phi));
        sigma_S=0;
    end
    if ((phi_S~=0 || phi_S~=360) && theta==0)
        sigma_S=0;
        if phi_S>phi
            sigma=acosd(cosd(theta_S)*cosd(phi+(360-phi_S)));
        else
            sigma=acosd(cosd(theta_S)*cosd(phi-phi_S));
        end
    end
    if (theta_S==0 && theta==0 && (phi_S==0 || phi_S==360))
        sigma=phi;
    elseif (theta_S==0 && theta==0 && (phi_S~=0 || phi_S~=360))
        if phi_S>phi
            sigma=phi+(360-phi_S);
        else
            sigma=phi-phi_S;
        end
    end
elseif (mu==0 || mu==180)
    sigma=0;
    sigma_S=0;
end
phi=phi_n;

%%% Rotation matrix calculated from sigma and sigma_S: we are looking to
%%% the beam (or we see that beam is coming towards us/detector)
L_sigma_S=[1 0 0 0; 0 cosd(2*(sigma_S)) sind(2*(sigma_S)) 0; 0 -sind(2*(sigma_S))
cosd(2*(sigma_S)) 0; 0 0 0 1];
L_sigma=[1 0 0 0; 0 cosd(2*(sigma)) sind(2*(sigma)) 0; 0 -sind(2*(sigma))
cosd(2*(sigma)) 0; 0 0 0 1];
if phi>180
    L_sigma_S=[1 0 0 0; 0 cosd(2*(180-sigma_S)) sind(2*(180-sigma_S)) 0; 0 -
sind(2*(180-sigma_S)) cosd(2*(180-sigma_S)) 0; 0 0 0 1];
    L_sigma=[1 0 0 0; 0 cosd(2*(180-sigma)) sind(2*(180-sigma)) 0; 0 -sind(2*(180-
sigma)) cosd(2*(180-sigma)) 0; 0 0 0 1];
```

```
end
if phi>180
   sigma=180-sigma;
   sigma_S=180-sigma_S;
else
   sigma=sigma;
   sigma_S=sigma_S;
end
end

function [theta,phi]=uxuyuz2thetaphi(ux,uy,uz)
theta=acosd(uz);
phi_temp=acosd(ux/sqrt(1-uz^2));
if uz^2==1
phi=0+(360-0)*rand(1);
elseif uz^2~=1 && uy>=0
phi=phi_temp;
elseif uz^2~=1 && uy<0
phi=360-phi_temp;
end
end

function [new]=select_data_within_limit_greater_zen(data,which_zenith)
   k_k=find((data(:,2)>which_zenith));
   new(:,:)=data(k_k,:); %Azimuth
end

function [new]=select_data_within_limit_less_zen(data,which_zenith)
   k_k=find((data(:,2)<which_zenith));
   new(:,:)=data(k_k,:); %Azimuth
end

function [new]=select_data_within_limit_greater_az(data,which_az)
   k_k=find((data(:,1)>which_az));
   new(:,:)=data(k_k,:); %Azimuth
end

function [new]=select_data_within_limit_less_az(data,which_az)
   k_k=find((data(:,1)<which_az));
   new(:,:)=data(k_k,:); %Azimuth
end

function
[ouput_data]=average_over_whole_data_deltheta_delphi(input_data,del_theta,del_phi)
%% Averaging throughout the whole data with certain del_theta and del_phi
phi_all=0:del_phi:360;
```

```
theta_all=0:del_theta:90;

for i=1:length(phi_all)
   for j=1:length(theta_all)
      phi_Mat(i,j)=phi_all(i);
      theta_Mat(i,j)=theta_all(j);
   end
end
%% Make the column matrix
phi_Mat=phi_Mat';
theta_Mat=theta_Mat';
phi_col_Mat=phi_Mat(:);
theta_col_Mat=theta_Mat(:);
%% Make the 1D matrix for fast operation
phi_col_Mat(1:end-length(theta_all),2)=phi_col_Mat(length(theta_all)+1:end,1);
theta_col_Mat(1:end-1,2)=theta_col_Mat(2:end,1);

theta_90_pos=find(theta_col_Mat(:,1)==90);
theta_col_Mat(theta_90_pos,:)=[];
phi_col_Mat(theta_90_pos,:)=[];

phi_360_pos=find(phi_col_Mat(:,1)==360);
phi_col_Mat(phi_360_pos,:)=[];
theta_col_Mat(phi_360_pos,:)=[];
%% Separate the data within a range and do average of those data
num_of_column=size(input_data,2);
ouput_data=zeros(length(phi_col_Mat),num_of_column);

parfor i=1:length(ouput_data)
do_average=select_data_within_limit_greater_az(input_data,phi_col_Mat(i,1));
do_average=select_data_within_limit_less_az(do_average,phi_col_Mat(i,2));
do_average=select_data_within_limit_greater_zen(do_average,theta_col_Mat(i,1));
do_average=select_data_within_limit_less_zen(do_average,theta_col_Mat(i,2));
if ~isempty(do_average)
   ouput_data(i,:)=mean(do_average,1); %average over each column
end
end
%% Remove any row with all zero
ouput_data = ouput_data(any(ouput_data,2),:); %%Removes any row with all zero
end

function v=myatand_0to180(y,x)
  v=zeros(size(x));
  v(x>0 & y>0) = atand( y(x>0 & y>=0) ./ x(x>0 & y>=0) );
  v(x>0 & y==0) = 0;
  v(x>0 & y<0) = 2*180+atand( y(x>0 & y<0) ./ x(x>0 & y<0) );
```

```
  v(x<0 & y>0) = 180+atand( y(x<0 & y>0) ./ x(x<0 & y>0) );
  v(x<0 & y==0) = 180;
  v(x<0 & y<0)  = 180+atand( y(x<0 & y<0) ./ x(x<0 & y<0) );
  v(x==0 & y>0) = 180/2;
  v(x==0 & y==0) = NaN;
  v(x==0 & y<0) = 180+180/2;
  v(isnan(x)==1 & isnan(y)==1) = NaN;
end

function []=ploar_contour_plot(phi,theta,Z)
vals=Z;
az=phi;
incl=theta;

figure
ph = polarscatter(az * pi/180, incl, [], vals, 'filled'); %convert to radians
ph. LineWidth=0.000001;
ph.SizeData=10; %50
colormap(jet(512));
h = colorbar();
grid off
set(gca, 'FontSize', 36);
set(gca, 'FontName', 'Times New Roman');
set(gca, 'FontWeight', 'bold');
set(gcf,'color','w');
pax=gca;
pxa.ThetaAxis.Visible = 'off'
pax.RTick=[]; % Theta labels
end

function [input_data]=remove_theta_larger_than_90(input_data)
   count_larger_90=1;
   out_of_water=[];
   for i=1:size(input_data,1)
      if (input_data(i,2)>90)
         out_of_water(count_larger_90)=i;
         count_larger_90=count_larger_90+1;
      end
   end
   if ~isempty(out_of_water)
      input_data(out_of_water,:)=[];
   end
end
```

APPENDIX G

MATLAB CODE FOR MIRROR POLARIZATION MAP SIMULATION CONSIDERING SCATTERING FROM NANOPARTICLES

```
%% Skylight Reflect and Sunlight Scattering from particles on mirror.
clc
clear all
close all

save_result=0;
main_folder_location='E:\Research\Projects\DOE SETO Project\My work-ZER\Mirror Polarization\Measuremnt\Outside Measurment\02-11-22\Mirror Sample 2 with reference\Simulation with PSD';

%% Particle and environmental properties
lamda=530e-9;
n_med=1; % Index of the surrounding
n_glass=1.52;% Index of the mirror glass
k0=2*pi/lamda;
m=1.57+.0000i; %Particle index
probability_cal_crosssection=1;

% % % % Dust sample from Sandia
PDC_1={'MM'};
pdc=1;
P_s=dlmread(strcat('E:\Research\Projects\DOE SETO Project\My work-ZER\Simulation result storage\Different size distribution peak\PSD_',(PDC_1{pdc}),'.txt'));

%%
P_s=sortrows(P_s,1);%Diameter
figure(1)
hold on
plot(P_s(:,1)*12,P_s(:,2),'-','linewidth',5)
xlim([min(P_s(:,1))-100 max(P_s(:,1))])
xlabel('Particle Size (nm)')
ylabel('Number of particles')
set(gca, 'FontSize', 40);
set(gcf,'color','w');
set(gca, 'FontName', 'Times New Roman')
% set(gca, 'FontWeight', 'bold')
set(gca,'linewidth',2);

%% camera viewing direction
% %Full map
% camera_az_mid=180
% camera_zen_mid=45
% camera_acceptance_az=360;
% camera_acceptance_zen=90;

% %At certain direction
```

```
camera_az_mid=330
camera_zen_mid=65
camera_acceptance_az=10;
camera_acceptance_zen=10;

%%
camera_az=[camera_az_mid-camera_acceptance_az/2
camera_az_mid+camera_acceptance_az/2];
camera_zen=[camera_zen_mid-camera_acceptance_zen/2
camera_zen_mid+camera_acceptance_zen/2];

%% Mirror
start_phi=0;
end_phi=360;
start_theta=0;
end_theta=90;

how_many_steps_in_one_degree=1;
angle_res=1/how_many_steps_in_one_degree;

phi_mirror_image=start_phi:angle_res:end_phi;
theta_mirror_image=start_theta:angle_res:end_theta;
%% Source direction & Stokes parameter of Incident light
%%Date
month=3;
day=1;
year=2021;

%Time of day
hour=15;
minute=18;
second=00;
time=hour+(minute/60)+(second/60/60); % in hour (11:30am is 11.5hr)

%%% Church Parking lot
phi_lat=33.419258;
phi_lon=-111.929590;
GMT=-7;
DLS=0;

[theta,phi] = sun_position(month,day,year,time,phi_lat,phi_lon,GMT,DLS)

%%
reference_az_measured=230
actual_sun_az_at_reference=225
```

```
phi1=reference_az_measured-(phi-actual_sun_az_at_reference) % Inital sun az on the
black mat - (current phi - initial phi)

if phi1>360
   phi1=phi1-360;
end
phi1
theta

cam_az_T=phi_mirror_image;
cam_zen_T=theta_mirror_image;

[cam_az,cam_zen,I_Skylight_Pol,Q_Skylight_Pol,U_Skylight_Pol,V_Skylight_Pol,skyli
ght_zen_Pol,skylight_az_Pol,DoLP_Skylight,AoP_Skylight,DoCP_Skylight,DoP_Skylig
ht,un_normalized_I_Skylight_Pol,un_normalized_Q_Skylight_Pol,un_normalized_U_Sk
ylight_Pol,un_normalized_V_Skylight_Pol]=get_skylight_Rayleigh_Scat(theta,phi1,cam
_az_T,cam_zen_T);

S_skylight=[un_normalized_I_Skylight_Pol un_normalized_Q_Skylight_Pol
un_normalized_U_Skylight_Pol un_normalized_V_Skylight_Pol]; %unpolarized
skylight stokes parameter
skylight_angles_pol=[skylight_az_Pol skylight_zen_Pol];

%% Full mapping
figure
ploar_contour_plot(skylight_angles_pol(:,1),skylight_angles_pol(:,2),AoP_Skylight,8)
figure
ploar_contour_plot(skylight_angles_pol(:,1),skylight_angles_pol(:,2),DoLP_Skylight,8)
pax = gca;
pax.ThetaZeroLocation = 'right'%'top';

figure
ploar_contour_plot(skylight_angles_pol(:,1),skylight_angles_pol(:,2),DoCP_Skylight,8)
figure
ploar_contour_plot(skylight_angles_pol(:,1),skylight_angles_pol(:,2),DoP_Skylight,8)

%% Skylight gets reflected from mirror
for skylight_s_c=1:length(skylight_angles_pol)
   % MIrror muller matrix and reflected stokes parameter
   S_reflected_skylight(skylight_s_c,:)=S_skylight(skylight_s_c,:);
   skylight_angles_pol(skylight_s_c,1)=skylight_angles_pol(skylight_s_c,1);
   if skylight_angles_pol(skylight_s_c,1)>360
      skylight_angles_pol(skylight_s_c,1)=skylight_angles_pol(skylight_s_c,1)-360;
   end
end
```

```
mod_skylght=[skylight_angles_pol S_reflected_skylight];
mod_skylght=sortrows(mod_skylght,1);

skylight_angles_pol=mod_skylght(:,1:2);
S_reflected_skylight=mod_skylght(:,3:6);

un_normalized_I_Skylight_Pol_reflected=S_reflected_skylight(:,1);
un_normalized_Q_Skylight_Pol_reflected=S_reflected_skylight(:,2);
un_normalized_U_Skylight_Pol_reflected=S_reflected_skylight(:,3);
un_normalized_V_Skylight_Pol_reflected=S_reflected_skylight(:,4);

%% Skylight Intensity distribution and Skylight Ratio
Intensity_distribution_of_skylight=un_normalized_I_Skylight_Pol_reflected./max(un_no
rmalized_I_Skylight_Pol_reflected);
mult_factor=0.61 % Made up number to reduce the DoLP in Skylight

%Intensity_Distribution
figure
ploar_contour_plot(skylight_az_Pol,skylight_zen_Pol,Intensity_distribution_of_skylight,
8)

%%Multiply the skylight polarization with its intensity distribution
un_normalized_I_Skylight_Pol_reflected=un_normalized_I_Skylight_Pol_reflected.*Inte
nsity_distribution_of_skylight;
un_normalized_Q_Skylight_Pol_reflected=un_normalized_Q_Skylight_Pol_reflected.*I
ntensity_distribution_of_skylight*mult_factor;
un_normalized_U_Skylight_Pol_reflected=un_normalized_U_Skylight_Pol_reflected.*I
ntensity_distribution_of_skylight*mult_factor;
un_normalized_V_Skylight_Pol_reflected=un_normalized_V_Skylight_Pol_reflected.*I
ntensity_distribution_of_skylight*mult_factor;

%% Calculate DoP, DoLP, AoP and DoCP after considering skylight intensity
distribution
% DoP
DoP_Skylight_mod=sqrt(un_normalized_Q_Skylight_Pol_reflected.^2+un_normalized_
U_Skylight_Pol_reflected.^2+un_normalized_V_Skylight_Pol_reflected.^2)./un_normali
zed_I_Skylight_Pol_reflected;
% DoLP
DoLP_Skylight_mod=sqrt(un_normalized_Q_Skylight_Pol_reflected.^2+un_normalized
_U_Skylight_Pol_reflected.^2)./un_normalized_I_Skylight_Pol_reflected;
% DoCP
DoCP_Skylight_mod=un_normalized_V_Skylight_Pol_reflected./un_normalized_I_Skyl
ight_Pol_reflected;
for i=1:length(skylight_angles_pol(:,1))
```

```
AoP_Skylight_mod(i,1)=0.5*myatand_0to180(un_normalized_U_Skylight_Pol_reflected
(i),un_normalized_Q_Skylight_Pol_reflected(i));
end

%%
DoLP_Skylight_mod=DoLP_Skylight_mod;

%% Full maping (skylight observed through the mirror)
figure
ploar_contour_plot(skylight_angles_pol(:,1),skylight_angles_pol(:,2),AoP_Skylight_mod
,8)
figure
ploar_contour_plot(skylight_angles_pol(:,1),skylight_angles_pol(:,2),DoLP_Skylight_m
od,8)
figure
ploar_contour_plot(skylight_angles_pol(:,1),skylight_angles_pol(:,2),DoCP_Skylight_m
od,8)
figure
ploar_contour_plot(skylight_angles_pol(:,1),skylight_angles_pol(:,2),DoP_Skylight_mod
,8)

%% Select data within camera viewing angle coverage
%Skylight data
data1=[skylight_angles_pol(:,1) skylight_angles_pol(:,2) DoLP_Skylight_mod
AoP_Skylight_mod DoCP_Skylight_mod DoP_Skylight_mod
un_normalized_I_Skylight_Pol_reflected un_normalized_Q_Skylight_Pol_reflected
un_normalized_U_Skylight_Pol_reflected un_normalized_V_Skylight_Pol_reflected];

%Select the data within camera viewing angle
[new_data,~]=select_data_within_limit_greater_az(data1,[],camera_az(1));
[new_data,~]=select_data_within_limit_less_az(new_data,[],camera_az(2));
[new_data,~]=select_data_within_limit_greater_zen(new_data,[],camera_zen(1));
[new_data,~]=select_data_within_limit_less_zen(new_data,[],camera_zen(2));

camera_az_skylight_all=new_data(:,1);
camera_zen_skylight_all=new_data(:,2);
camera_dolp_skylight_all=new_data(:,3);
camera_aop_skylight_all=new_data(:,4);
camera_docp_skylight_all=new_data(:,5);
camera_dop_skylight_all=new_data(:,6);
camera_un_normalized_I_Skylight_Pol=new_data(:,7);
camera_un_normalized_Q_Skylight_Pol=new_data(:,8);
camera_un_normalized_U_Skylight_Pol=new_data(:,9);
camera_un_normalized_V_Skylight_Pol=new_data(:,10);
```

```
camera_skylight=[camera_un_normalized_I_Skylight_Pol
camera_un_normalized_Q_Skylight_Pol camera_un_normalized_U_Skylight_Pol
camera_un_normalized_V_Skylight_Pol];

%% Full maping after considering skylight intensity
figure
ploar_contour_plot(skylight_az_Pol,skylight_zen_Pol,DoLP_Skylight_mod,8)

%% Maping within camera viweing angle
figure
ploar_contour_plot(camera_az_skylight_all,camera_zen_skylight_all,camera_dolp_skyli
ght_all,8)

%% DoLP vs mirror Reflection for only skylight
R=[80:1:100]'; %mirror reflection

figure
hold on
plot(R,mean(camera_dolp_skylight_all).*R/100,'-o','linewidth',5)
xlabel('R(%)')
ylabel('DoLP')
set(gca, 'FontSize', 18);
set(gcf,'color','w');

%% Particle Size parameter
x=[k0*n_med*(P_s(:,1)'*1e-9/2)];

%% Single Mie Scattering of unpolarized light from a Mirror surface
% Input light stokes parameter
S_source=[1 0 0 0]; %unpolarized sunlight

phi_source_all=(phi1);
theta_source_all=theta;

%First cell of I1 and I2 corresponds to no scattering event for sunlight.
for source_count=1:length(phi_source_all)

    phi_source=phi_source_all(source_count);
    theta_source=theta_source_all(source_count);
    S=S_source(source_count,:)';
    [I1,Q1,U1,V1] =
get_mirror_stokes_map_size_distribution(phi_mirror_image,theta_mirror_image,phi_sou
rce,theta_source,m,n_med,x,k0,S);

    theta_source_image=180-theta_source;
    phi_source_image=phi_source_all(source_count);
```

```
        [I2,Q2,U2,V2] =
get_mirror_stokes_map_size_distribution(phi_mirror_image,theta_mirror_image,phi_sou
rce_image,theta_source_image,m,n_med,x,k0,S);
end

%%
%%Sunlight : Skylight ratio --->depends on weather condition
sun_light_all=[.85];

for sun_l_count=1:length(sun_light_all)
sun_light=sun_light_all(sun_l_count);
sky_light=1-sun_light_all(sun_l_count);

%% Modify particle concentration
modifier=[0.0225 0.025 0.0272 0.03 0.04 0.045 0.05 0.06 0.07 0.08 0.09 0.1 .12 0.15 .18
0.2 .22 0.25 .28 0.3 0.4 0.5 0.65 1 1.5 2 3 8]/10; %modify the particle concentration

for mod_count=1:length(modifier)
particle_size=P_s(:,1)'; %Diameter of the particle
number_concentration_=P_s(:,2)'/modifier(mod_count);
Totoal_area_covered_by_particles=sum(pi*(particle_size*1e-
9/2).^2.*number_concentration_); %in percentage
total_area_of_image=280e-6*280e-6; %280um x 280um for 10x objective lens (NA=0.3)
percent_area_covered_by_particles=Totoal_area_covered_by_particles/total_area_of_im
age
percent_area_NOT_covered_by_particles=1-percent_area_covered_by_particles
R_collect(mod_count)=percent_area_NOT_covered_by_particles+0.25*percent_area_co
vered_by_particles;

%%
figure
plot(particle_size,number_concentration_,'-','linewidth',5)
xlim([min(particle_size)-100 max(particle_size)])
xlabel('Particle Size (nm)')
ylabel('Number of particles')
set(gca, 'FontSize', 25);
set(gcf,'color','w');
set(gca, 'FontName', 'Times New Roman')
set(gca,'linewidth',2);

%%
for i=1:length(particle_size)
   %Particle size
   radius=particle_size(i)/2; %in nm
   a=radius*1e-9; %radius in m
   %Size Parameter
```

```
    x=k0*a*n_med; %Size parameter
    %Refractive index mismatch ratio
    n_r=m/n_med;
    %Geometrical Cross-sectional area of the particles
    A=pi*a^2; %m^2
    %Concentration or volume density or number density of particles in the medium
    p_s=number_concentration_(i); %in m-3
    %Scattering efficiency
    Res_temp= mie(n_r,x); %Run the mie theory algorithm
    Qs=Res_temp(5); %Scattering efficiency
    Qa=Res_temp(6); %Absorption efficiency
    %Effective scattering cross-section or scattering cross section
    sigma_s(i)=Qs*A;
    %Effective absorption cross-section or absorption cross section
    sigma_a(i)=Qa*A;
    %Scattering coefficient
    us(i)= p_s*sigma_s(i);
    %Absorption coefficient
    ua(i)= p_s*sigma_a(i);
    %Attenuation coefficient
    ut(i)=ua(i)+us(i); %attenuation coefficient in m^-1
end
%Total scattering coefficient
uS=sum(us);
%Total attenuation coefficient
uT=sum(ut);

%% Get the PSD
if probability_cal_crosssection==1
    %PSD using crossection of particles
    PSD_probabilty=(ut.*number_concentration_)/sum(ut.*number_concentration_);
%Probability obtained from crosssection
else
    %PSD using number conentration of particles
    PSD_probabilty=number_concentration_/sum(number_concentration_);
end

%%
figure
plot(particle_size,ut,'-','linewidth',5)
xlim([min(particle_size)-100 max(particle_size)])
xlabel('Particle Size (nm)')
ylabel('Attenuation Coefficient')
set(gca, 'FontSize', 25);
set(gcf,'color','w');
set(gca, 'FontName', 'Times New Roman')
```

```
set(gca,'linewidth',2);

figure
plot(particle_size,PSD_probabilty,'-','linewidth',5)
xlim([min(particle_size)-100 max(particle_size)])
xlabel('Particle Size (nm)')
ylabel('Scattering Probability')
set(gca, 'FontSize', 25);
set(gcf,'color','w');
set(gca, 'FontName', 'Times New Roman')
set(gca,'linewidth',2);
%% Particle Size Distribution and Probability
particle_size=[particle_size]';
PSD_probabilty=[PSD_probabilty]';
%% Add the Mie scattering from different particle size according to PSD and number
concentration
I_T=0;
Q_T=0;
U_T=0;
V_T=0;
%First cell of I1 and I2 corresponds to no scattering event for sunlight.
for mie_count=1:length(PSD_probabilty)
   I_T=I_T+(I1{mie_count,1}+I2{mie_count,1})*PSD_probabilty(mie_count);
   Q_T=Q_T+(Q1{mie_count,1}+Q2{mie_count,1})*PSD_probabilty(mie_count);
   U_T=U_T+(U1{mie_count,1}+U2{mie_count,1})*PSD_probabilty(mie_count);
   V_T=V_T+(V1{mie_count,1}+V2{mie_count,1})*PSD_probabilty(mie_count);
end
I_T_N=I_T./I_T;
Q_T_N=Q_T./I_T;
U_T_N=U_T./I_T;
V_T_N=V_T./I_T;
DoP= sqrt(Q_T_N.^2+U_T_N.^2+V_T_N.^2)./I_T_N;
DoLP=sqrt(Q_T_N.^2+U_T_N.^2)./I_T_N;
DoCP=V_T_N./I_T_N;

for w=1:length(phi_mirror_image)
   for v=1:length(theta_mirror_image)
      %angle of polarization
      AoP(w,v)=0.5*myatand_0to180(U_T_N(w,v),Q_T_N(w,v)); %for 0 to 180 degree
   end
end
%%
phi_mirror_scat_pol=skylight_angles_pol(:,1);
theta_mirror_scat_pol=skylight_angles_pol(:,2);

DoLP_mirror_scat_pol=DoLP';
```

```
DoLP_mirror_scat_pol=DoLP_mirror_scat_pol(:);
DoP_mirror_scat_pol=DoP';
DoP_mirror_scat_pol=DoP_mirror_scat_pol(:);
DoCP_mirror_scat_pol=DoCP';
DoCP_mirror_scat_pol=DoCP_mirror_scat_pol(:);
AoP_mirror_scat_pol=AoP';
AoP_mirror_scat_pol=AoP_mirror_scat_pol(:);
I_T_mirror_scat_pol=I_T';
I_T_mirror_scat_pol=I_T_mirror_scat_pol(:);

%% Full polarization mapping from mirror
figure
ploar_contour_plot(phi_mirror_scat_pol,theta_mirror_scat_pol,DoLP_mirror_scat_pol,8)
pax = gca;
% pax.ThetaDir='clockwise';
pax.ThetaZeroLocation = 'right'%'top';
dolp_ll=0; %DoCP lower limit (ll)
dolp_ml=.6; % DoCP max limit (ml)
dolp_step=0.1; %DoCP colorbar tick step
caxis([dolp_ll, dolp_ml]); %change colorbar limit
cbh=colorbar();
set(cbh,'YTick',[dolp_ll:dolp_step:dolp_ml]); % Chnage color bar ticks

figure
ploar_contour_plot(phi_mirror_scat_pol,theta_mirror_scat_pol,I_T_mirror_scat_pol,8)
%%
figure
contourf(phi_mirror_image,theta_mirror_image,real(AoP)',500,'edgecolor','none');
colorbar
colormap(jet(512))
xlabel('camra__azimuth');
ylabel('camera__zenith');
title('AoP')
set(gca, 'FontSize', 18);
set(gcf,'color','w');

figure
contourf(phi_mirror_image,theta_mirror_image,real(DoLP)',500,'edgecolor','none');
colorbar
colormap(jet(512))
xlabel('camra__azimuth');
ylabel('camera__zenith');
title('DoLP')
set(gca, 'FontSize', 18);
set(gcf,'color','w');
```

```
figure
contourf(phi_mirror_image,theta_mirror_image,real(DoCP)',500,'edgecolor','none');
colorbar
colormap(jet(512))
xlabel('camra__azimuth');
ylabel('camera__zenith');
title('DoCP')
set(gca, 'FontSize', 18);
set(gcf,'color','w');

figure
contourf(phi_mirror_image,theta_mirror_image,real(DoP)',500,'edgecolor','none');
colorbar
colormap(jet(512))
xlabel('camra__azimuth');
ylabel('camera__zenith');
title('DoP')
set(gca, 'FontSize', 18);
set(gcf,'color','w');

%% Sun_light: Unormalized Mirror polarization modified with sun_light
un_normalized_I_mirror_Pol=I_T';
un_normalized_I_mirror_Pol=un_normalized_I_mirror_Pol(:);
un_normalized_Q_mirror_Pol=Q_T';
un_normalized_Q_mirror_Pol=un_normalized_Q_mirror_Pol(:);
un_normalized_U_mirror_Pol=U_T';
un_normalized_U_mirror_Pol=un_normalized_U_mirror_Pol(:);
un_normalized_V_mirror_Pol=V_T';
un_normalized_V_mirror_Pol=un_normalized_V_mirror_Pol(:);

%%
%MIrror scattering data
data2=[phi_mirror_scat_pol theta_mirror_scat_pol un_normalized_I_mirror_Pol
un_normalized_Q_mirror_Pol un_normalized_U_mirror_Pol
un_normalized_V_mirror_Pol];

%Select the data within camera viewing angle
[new_data2,~]=select_data_within_limit_greater_az(data2,[],camera_az(1));
[new_data2,~]=select_data_within_limit_less_az(new_data2,[],camera_az(2));
[new_data2,~]=select_data_within_limit_greater_zen(new_data2,[],camera_zen(1));
[new_data2,~]=select_data_within_limit_less_zen(new_data2,[],camera_zen(2));

camera_az_mirror_all=new_data2(:,1);
camera_zen_mirror_all=new_data2(:,2);
camera_un_normalized_I_mirror_Pol=new_data2(:,3);
camera_un_normalized_Q_mirror_Pol=new_data2(:,4);
```

```
camera_un_normalized_U_mirror_Pol=new_data2(:,5);
camera_un_normalized_V_mirror_Pol=new_data2(:,6);
camera_mirror=[camera_un_normalized_I_mirror_Pol
camera_un_normalized_Q_mirror_Pol camera_un_normalized_U_mirror_Pol
camera_un_normalized_V_mirror_Pol];

%% Maping within camera viweing angle
figure
ploar_contour_plot(camera_az_mirror_all,camera_zen_mirror_all,camera_dolp_mirror_a
ll,8)

%% Total effect
camera_az_all=camera_az_skylight_all;
camera_zen_all=camera_zen_skylight_all;
%Camera acceptance angle is 10 degree so, out of 180 scattering angle 10
%degrees will be accepted in the camera as direct reflection. On average 6% or 10/180.
camera_total_un_normalized_S=camera_mirror./(camera_mirror(:,1))*sun_light*percent
_area_covered_by_particles+camera_skylight./(camera_skylight(:,1))*sky_light*(percent
_area_NOT_covered_by_particles+0.06*percent_area_covered_by_particles);

%% Calculate DoP, DoLP, AoP and DoCP after considering both sunlight scating and
skylight reflection
% DoP
DoP_camera_mod=sqrt(camera_total_un_normalized_S(:,2).^2+camera_total_un_norma
lized_S(:,3).^2+camera_total_un_normalized_S(:,4).^2)./camera_total_un_normalized_S
(:,1);
% DoLP
DoLP_camera_mod=sqrt(camera_total_un_normalized_S(:,2).^2+camera_total_un_norm
alized_S(:,3).^2)./camera_total_un_normalized_S(:,1);
% DoCP
DoCP_camera_mod=camera_total_un_normalized_S(:,4)./camera_total_un_normalized_
S(:,1);

for i=1:length(camera_az_all)
AoP_camera_mod(i,1)=0.5*myatand_0to180(camera_total_un_normalized_S(i,3),camer
a_total_un_normalized_S(i,2));
end

%% DoLP total
figure
ploar_contour_plot(camera_az_all,camera_zen_all,DoLP_camera_mod,8)
pax = gca;
pax.ThetaZeroLocation = 'right'%'top';
dolp_ll=0; %DoCP lower limit (ll)
dolp_ml=.6; % DoCP max limit (ml)
dolp_step=0.1; %DoCP colorbar tick step
```

```
caxis([dolp_ll, dolp_ml]); %change colorbar limit
cbh=colorbar();
set(cbh,'YTick',[dolp_ll:dolp_step:dolp_ml]); % Chnage color bar ticks

%%
avg_dolp_camera(mod_count)=mean(DoLP_camera_mod);
disp(strcat('Mirror Reflection:',num2str(R_collect)))
disp(strcat('Average DoLP:',num2str(avg_dolp_camera)))
clear I_T
clear Q_T
clear U_T
clear V_T
end

%%DoLP vs R
figure(1001)
hold on
plot(R_collect*100,avg_dolp_camera,'-o','linewidth',5)
% plot(R_collect*100,1.75*avg_dolp_camera,'-o','linewidth',5)
xlabel('R(%)');
ylabel('DoLP');
set(gca, 'FontSize', 45);
set(gcf,'color','w');
legend_collect{sun_l_count}=strcat('Sun:Sky=',num2str(sun_light*100),':',num2str(sky_l
ight*100));

end
%%
figure(1001)
% Reflection obtained from Large area reflection
hold on
plot([82.14 92.72 94.26 100 96.34 95.38 98.43 100 94.99 96.49 98.27 100 92.94 95.23
96.99 100],[[0.2359 0.3437 0.4518 0.6173]*(0.6099/0.6173) [0.4933 0.4808 0.5625
0.6163]*(0.6099/0.6178) [0.4447 0.4723 0.5519 0.6101]*(0.6099/0.6178) [0.4198 0.4814
0.5185 0.6178]*(0.6099/0.6178)],'*','linewidth',15)
legend('Model', 'Reference Sample', 'Sample Under Test 1', 'Sample Under Test 2',
'Sample Under Test 3','fontsize',40)
legend box off
yticks([0.2:.1:.6])
xticks([80:2:100])
xlim([80 100.5])
ylim([0.2 .65])
xtickangle(0)
set(gca, 'FontName', 'Times New Roman');
set(gca, 'FontSize', 45);
box on
```

```
set(gca, 'linewidth', 5);

%%
if save_result==1
    new_directory_name=strcat('source_AZ_',num2str(phi),'_ZEN_',num2str(theta));
    mkdir(main_folder_location,new_directory_name);
    main_folder_location_new=strcat(main_folder_location,'\',new_directory_name);
    if probability_cal_crosssection==0
save(strcat(main_folder_location_new,'\data_PSD_',PDC_1{pdc},'_new2.mat'),'AoP','DoLP','DoCP','DoP','phi_mirror_image','theta_mirror_image','DoLP_pol','AoP_pol','DoCP_pol','DoP_pol','phi_mirror_image_pol','theta_mirror_image_pol')
    elseif probability_cal_crosssection==1
save(strcat(main_folder_location_new,'\data_PSD_',PDC_1{pdc},'_ScatCrossection_new2.mat'),'AoP','DoLP','DoCP','DoP','phi_mirror_image','theta_mirror_image','DoLP_pol','AoP_pol','DoCP_pol','DoP_pol','phi_mirror_image_pol','theta_mirror_image_pol')
    end
end
toc
          %%%%%%%%%%% Functions used in this Code%%%%%%%%%%%

function [I_total,Q_total,U_total,V_total] =
get_mirror_stokes_map_size_distribution(phi_mirror,theta_mirror,phi,theta,m,n_med,x,k0,S)
%% Get the rotation matrix
%Looking or camera direction
phi_S=phi_mirror;
theta_S=theta_mirror;
L_sigma_S=cell(length(phi_S),length(theta_S));
L_sigma=cell(length(phi_S),length(theta_S));
for i=1:length(phi_S)
    for j=1:length(theta_S)
        %As some of the phi_S are problematic (produces invalid rotation angles) we try to calculate close to these values
        %but not exactly at those values of phi_S
        if phi_S(i)==phi
            phi_S(i)=phi_S(i)+0.001;
        elseif phi_S(i)==phi-180
            phi_S(i)=phi_S(i)+0.001;
        elseif theta_S(j)==0
            theta_S(j)=theta_S(j)+0.001;
        elseif theta_S(j)==90
            theta_S(j)=theta_S(j)-0.001;
        end
        if theta==0
            theta=theta+1e-3;
        elseif theta==90
```

```
            theta=theta+1e-3;
        elseif theta==180
            theta=theta+1e-3;
        end
        mu(i,j)=(acosd(sind(theta)*sind(theta_S(j))*cosd(phi_S(i)-
phi)+cosd(theta)*cosd(theta_S(j))));
        [L_sigma_S{i,j}, L_sigma{i,j}]=get_rotation_matrix(phi,theta,phi_S(i),theta_S(j),
mu(i,j));
    end
end
scattering_angle=mu;
n_r=m/n_med;
for p_count=1:length(x)
    x_select=x(p_count);
    %% Single Mie Scattering
    I=zeros(length(phi_S),length(theta_S));
    Q=zeros(length(phi_S),length(theta_S));
    U=zeros(length(phi_S),length(theta_S));
    V=zeros(length(phi_S),length(theta_S));
    for w=1:length(phi_S)
        for v=1:length(theta_S)
            if x_select~=0
                u=cos(pi/180*scattering_angle(w,v));
                %% S1 and S2
                result_s1_s2=mie_S12(n_r, x_select, u);
                S1=result_s1_s2(1);
                S2=result_s1_s2(2);
                %% Mie Scattering Matrix
                M11=1/2*(abs(S1)^2+abs(S2)^2);
                M12=1/2*(abs(S2)^2-abs(S1)^2);
                M33=1/2*(S2*conj(S1)+S1*conj(S2));
                M34=1/2*(S2*conj(S1)-S1*conj(S2))*1i;
                M=[M11 M12 0 0
                    M12 M11 0 0
                    0 0 M33 M34
                    0 0 -M34 M33];
                S_S=(L_sigma_S{w,v}*M*L_sigma{w,v})*S; % scattering in air (skylight)
                I(w,v)=S_S(1);
                Q(w,v)=S_S(2);
                U(w,v)=S_S(3);
                V(w,v)=S_S(4);
            else
                if phi_S(w)==phi && theta_S(v)==theta
                    I(w,v)=1;
                    Q(w,v)=0;
                    U(w,v)=0;
```

```
                V(w,v)=0;
            end
        end
    end
    I_total{p_count,1}=I;
    Q_total{p_count,1}=Q;
    U_total{p_count,1}=U;
    V_total{p_count,1}=V;
end
end

function
[cam_az,cam_zen,I_Pol,Q_Pol,U_Pol,V_Pol,cam_zen_Pol,cam_az_Pol,DoLP_Pol,AoP_
Pol,DoCP_Pol,DoP_Pol,un_normalized_I_pol,un_normalized_Q_pol,un_normalized_U_
pol,un_normalized_V_pol]=get_skylight_Rayleigh_Scat(theta,phi,cam_az_T,cam_zen_T
)
%%  Condition for the simulation
    error=0;
    lamda=480e-9; %%Wavelength
    Temp=5782; %Sun's blackbody temperature in K on TOA (top of Atmosphere)
    h=6.6261e-34; %plank's constant in Js
    c=3e8; %light velocity in ms-1;
    kb=1.3806e-23; %boltzmann's contant in JK-1
    %%% Solar irradiance measured at earth surface
    Rsun=6.95e5; %suns radius in km
    Rearth=1.496e8; % radius of earth's orbit in km
    I=(Rsun/Rearth)^2*2*pi*h*c^2/lamda^5*(1/(exp(h*c/(lamda*kb*Temp))-1))*lamda;
% in Wm-2
    % %particle detail
    r=50e-9; %Particle radius
    mr=1.53; %Real part of complex refractive index m = mr + i*mi
    mi=0.007;%Imaginary part of complex refractive index m = mr + i*mi
    m=mr+1i*mi;%Complex refractive index m = mr + i*mi
    % For unpolarized light I=I; For LP|| I,Q=I; For LP_per I=I,Q=-I;
    I=I; Q=0; U=0;V=0;
    S=[I;  Q;  U;  V];
    %%
cam_zen_T=cam_zen_T;
cam_az_T=cam_az_T;
flag1=0;
for ii=1:length(cam_az_T)
    if cam_az_T(ii)<0
        cam_az_T(ii)=cam_az_T(ii)+360;
        flag1=1;
    end
```

```
end
%if there is a negative number in the cam_az_T vector then we need to sort it
if flag1==1
   cam_az_T=sort(cam_az_T);
end
      S_M_T=cell(length(cam_az_T),length(cam_zen_T));
      DoLP_M_T=cell(length(cam_az_T),length(cam_zen_T));
      DoP_M_T=cell(length(cam_az_T),length(cam_zen_T));
      DoCP_M_T=cell(length(cam_az_T),length(cam_zen_T));
      Q_M_T=cell(length(cam_az_T),length(cam_zen_T));
      U_M_T=cell(length(cam_az_T),length(cam_zen_T));
      V_M_T=zeros(length(cam_az_T),length(cam_zen_T));
      AoP_M_T=cell(length(cam_az_T),length(cam_zen_T));
      I_M_T=cell(length(cam_az_T),length(cam_zen_T));
      Il_M_T=cell(length(cam_az_T),length(cam_zen_T));
      Ir_M_T=cell(length(cam_az_T),length(cam_zen_T));
      rho_M_T=cell(length(cam_az_T),length(cam_zen_T));
      Iunp_M_T=cell(length(cam_az_T),length(cam_zen_T));
      Il_unp_M_T=cell(length(cam_az_T),length(cam_zen_T));
      Ir_unp_M_T=cell(length(cam_az_T),length(cam_zen_T));
      Il_lin_M_T=cell(length(cam_az_T),length(cam_zen_T));
      Ir_lin_M_T=cell(length(cam_az_T),length(cam_zen_T));
      mu_M_T=cell(length(cam_az_T),length(cam_zen_T));
      beta_M_T=cell(length(cam_az_T),length(cam_zen_T));
      kR_M_T=cell(length(cam_az_T),length(cam_zen_T));
      sigma_M_T=cell(length(cam_az_T),length(cam_zen_T));
      sigma_S_M_T=cell(length(cam_az_T),length(cam_zen_T));
      if error==1
   er_DoP_M_T=cell(length(cam_az_T),length(cam_zen_T));
   er_AoP_M_T=cell(length(cam_az_T),length(cam_zen_T));
   er_Q_M_T=cell(length(cam_az_T),length(cam_zen_T));
   er_U_M_T=cell(length(cam_az_T),length(cam_zen_T));
      end
for i=1:length(cam_az_T)
   phi_S=cam_az_T(i);
   for j=1:length(cam_zen_T)
      theta_S=cam_zen_T(j);
      %As some of the phi_S are problematic (produces invalid rotation angles) we try to
calculate close to these values but not exactly at those values of phi_S
      if phi_S==phi
         phi_S=phi_S+0.001;
         cam_az_T(i)=phi_S;
      elseif phi_S==phi-180
         phi_S=phi_S+0.001;
         cam_az_T(i)=phi_S;
      elseif theta_S==0
```

```
        theta_S=theta_S+0.001;
        cam_zen_T(j)=theta_S;
    elseif theta_S==90
        theta_S=theta_S-0.001;
        cam_zen_T(j)=theta_S;
    end

% mu: scattering angle
mu=(acosd(sind(theta)*sind(theta_S)*cosd(phi_S-phi)+cosd(theta)*cosd(theta_S)));

%Rayleigh Scattering (partilce smaller than incident wavelength) Matrix--for
homogenous or radially inhomogeneous spherical particle
F11=(1+(cosd(mu)).^2)/2;
F12=-(1-(cosd(mu)).^2)/2;
F21=F12;
F22=F11;
F33=cosd(mu);
F34=0;
F43=0;
F44=F33;
F=[F11 F12 0 0; F21 F22 0 0; 0 0 F33 F34; 0 0 F43 F44];

a=2*pi*r/lamda; % Particle Size parameter
F_Rayl= a.^6*abs(((m.^2-1)/(m.^2+2)))^2.*F; %Rayley Scattering Matrix
%Stokes parameter after scattering (w.r.t scattering plane)
S_S=F_Rayl*S; %Matrix multiplication
%R is the radial distance from the scattering particle
kR=sqrt(F11*S(1)/S_S(1)); %Will be used to normalize F matrix

%Converting Scattering matrix in scattering plane to Phase matrix in local Meridian
Plane

%Calculation of ratation angle: incident (sigma) and scattering (sigma_S) rotation
angle[144]
phi_n=phi;
if phi>180
   phi=phi-180;
end
del_phi=phi_S-phi;
if ((mu ~= 0 || mu ~= 180) && ((0<=del_phi && del_phi<=180) && del_phi>=0))
    if (theta_S~=0 && theta~=0)
    sigma=acosd((cosd(theta_S)-cosd(theta)*cosd(mu))/(sind(mu)*sind(theta)));
    sigma_S=180+acosd((cosd(theta)-
cosd(theta_S)*cosd(mu))/(sind(mu)*sind(theta_S)));
   end
```

```
    if theta_S==0 % we don't have the condition when phi_S==0 or 360
       sigma=0;
       sigma_S=acosd(cosd(theta)*cosd(phi_S-phi));
    end
    if theta==0
       sigma=acosd(cosd(theta_S)*cosd(phi_S-phi));
       sigma_S=0;
    end
    if (theta_S==0 && theta==0)
       sigma=(phi_S-phi);
    end
elseif ((mu ~= 0 || mu ~= 180) && ((del_phi>0 &&(180<del_phi && del_phi<360)) ||
del_phi<0))
    if (theta_S~=0 && theta~=0)
    sigma=180-acosd((cosd(theta_S)-cosd(theta)*cosd(mu))/(sind(mu)*sind(theta)));
    sigma_S=360-acosd((cosd(theta)-cosd(theta_S)*cosd(mu))/(sind(mu)*sind(theta_S)));
    end
    if ((phi_S==0 || phi_S==360) && theta_S==0)
       sigma=0;
       sigma_S=acosd(cosd(theta)*cosd(phi));
    end
    if ((phi_S~=0 || phi_S~=360) && theta_S==0)
       sigma=0;
       if phi_S>phi
          sigma_S=acosd(cosd(theta)*cosd(phi+(360-phi_S)));
       else
          sigma_S=acosd(cosd(theta)*cosd(phi-phi_S));
       end
    end
    if ((phi_S==0 || phi_S==360) && theta==0)
       sigma=acosd(cosd(theta_S)*cosd(phi));
       sigma_S=0;
    end
    if ((phi_S~=0 || phi_S~=360) && theta==0)
       sigma_S=0;
       if phi_S>phi
          sigma=acosd(cosd(theta_S)*cosd(phi+(360-phi_S)));
       else
          sigma=acosd(cosd(theta_S)*cosd(phi-phi_S));
       end
    end
    if (theta_S==0 && theta==0 && (phi_S==0 || phi_S==360))
       sigma=phi;
    elseif (theta_S==0 && theta==0 && (phi_S~=0 || phi_S~=360))
       if phi_S>phi
          sigma=phi+(360-phi_S);
```

```
        else
            sigma=phi-phi_S;
        end
    end
elseif (mu==0 || mu==180)
    sigma=0;
    sigma_S=0;
end
phi=phi_n;
%%% Rotation matrix calculated from sigma and sigma_S: we are looking to
%%% the beam (or we see that beam is coming towards us/detector)
L_sigma_S=[1 0 0 0; 0 cosd(2*(sigma_S)) sind(2*(sigma_S)) 0; 0 -sind(2*(sigma_S))
cosd(2*(sigma_S)) 0; 0 0 0 1];
L_sigma=[1 0 0 0; 0 cosd(2*(sigma)) sind(2*(sigma)) 0; 0 -sind(2*(sigma))
cosd(2*(sigma)) 0; 0 0 0 1];
if phi>180
    L_sigma_S=[1 0 0 0; 0 cosd(2*(180-sigma_S)) sind(2*(180-sigma_S)) 0; 0 -
sind(2*(180-sigma_S)) cosd(2*(180-sigma_S)) 0; 0 0 0 1];
    L_sigma=[1 0 0 0; 0 cosd(2*(180-sigma)) sind(2*(180-sigma)) 0; 0 -sind(2*(180-
sigma)) cosd(2*(180-sigma)) 0; 0 0 0 1];
end
%Phase matrix from scattering matrix
if theta==0 && theta_S==0
    P=F_Rayl*L_sigma;
else
    P=L_sigma_S*F_Rayl*L_sigma;
end
%Final stokes parameter
S_S_C=P*S;
%%%Uncomment if you don't want to normalize
kR=1;
S_S_F=S_S_C./(kR)^2; %Nomrlaized Equation: 2.88 (_Stokes parameters of skylight
based on simulations and polarized radiometer measurements)
non_normalized_S=S_S_F;
S_S_F=S_S_F./S_S_F(1); %This is to normalize the Stokes parameters
DoP_S= sqrt(S_S_F(2)^2+S_S_F(3)^2+S_S_F(4)^2)/S_S_F(1);
DoLP_S=sqrt(S_S_F(2)^2+S_S_F(3)^2)/S_S_F(1);
DoCP_S=S_S_F(4)/S_S_F(1);
Il_S=(S_S_F(1)+S_S_F(2))/2; %Parallel component of polarized radiance (I); Function
of scattering angle;
Ir_S=(S_S_F(1)-S_S_F(2))/2 ;%Perpendicular component of polarized radiance (I);
Constant w.r.t scattering angle;
rho_S=Ir_S/Il_S; %Linear depolarization ratio - unpolarized: Il_S=Ir_S; Perpendicular:
Il_S=0; Partially Pol: Il_S<Ir_S
%Scattered light is partially polarized so, I_S = Iunp_S + Ilin_S;
```

```
%Unpolarized portion of scattered light-> Iunp_S;
Iunp_S=S(1)*(1-DoLP_S);
Il_unp_S= Iunp_S/2;%Parralel portion of unpolarized light
Ir_unp_S= Iunp_S/2; %Perpendicular portion of unpolarized light
%Polarized portion of scattered light-> Ilin_S;
Il_lin_S=Il_S-Il_unp_S; %Parralel portion of polarized light
Ir_lin_S=Ir_S-Ir_unp_S ;%Perpendicular portion of polarized light
%angle of polarization
xhi_S=0.5*myatand_0to180(S_S_F(3),S_S_F(2)); %for 0 to 180 degree

%Elipticity angle
beta_S= 0.5*((atand(S_S_F(4)./sqrt(S_S_F(2).^2+S_S_F(3).^2))));
S_M_T{i,j}=S_S_F;
if isnan(DoP_S)==1
   DoP_M_T{i,j}=DoP_S;
   DoLP_M_T{i,j}=DoLP_S;
   DoCP_M_T{i,j}=DoCP_S;
   AoP_M_T{i,j}=xhi_S;
   beta_M_T{i,j}=beta_S;
   Q_M_T{i,j}=S_S_F(2);
   U_M_T{i,j}=S_S_F(3);
   V_M_T(i,j)=S_S_F(4);
else
   error_percent=0.1;
   er_DoP=error*(error_percent*(-1+2 * rand));
   DoP_M_T{i,j}=DoP_S+er_DoP; %iff error is 1 then it will add 1% random error
between -1 and 1
   DoLP_M_T{i,j}=DoLP_S+error*(error_percent*(-1+2 * rand));%iff error is 1 then it
will add 1% random error between -1 and 1
   DoCP_M_T{i,j}=DoCP_S+error*(error_percent*(-1+2 * rand));%iff error is 1 then it
will add 1% random error between -1 and 1
   er_AoP=error*(error_percent*(-180+360 * rand));
   xhi_S=xhi_S+er_AoP;%iff error is 1 then it will add 1% random error between -180
and 180
   %as we are adding/subtratcting 1% of 180, sometimes the AoP will become
   %more than 180 or less than 0. So, to bring them in the range of 0 to 180
   %we do the following
   if error==1 && xhi_S>180
      xhi_S=xhi_S-180;
   elseif error==1 && xhi_S<0
      xhi_S=xhi_S+180;
   end
   AoP_M_T{i,j}=xhi_S; %error added in the previous line

beta_M_T{i,j}=beta_S+error*(error_percent*(-1+2 * rand));%iff error is 1 then it will
add 1% random error between -45 and 45
```

```
er_Q=error*(error_percent*(-1+2 * rand));
   Q_M_T{i,j}=S_S_F(2)+er_Q;%if error is 1 then it will add 1% random error between
-1 and 1
   er_U=error*(error_percent*(-1+2 * rand));
   U_M_T{i,j}=S_S_F(3)+er_U;%iff error is 1 then it will add 1% random error between
-1 and 1
end
if error==1
   er_DoP_M_T{i,j}=er_DoP;
   er_AoP_M_T{i,j}=er_AoP;
   er_Q_M_T{i,j}=er_Q;
   er_U_M_T{i,j}=er_U;
end
   un_normalized_I(i,j)=non_normalized_S(1);
   un_normalized_Q(i,j)=non_normalized_S(2);
   un_normalized_U(i,j)=non_normalized_S(3);
   un_normalized_V(i,j)=non_normalized_S(4);
   I_M_T{i,j}=S_S_F(1);
   Il_M_T{i,j}=Il_S;
   Ir_M_T{i,j}=Ir_S;
   rho_M_T{i,j}=rho_S;
   Iunp_M_T{i,j}=Iunp_S;
   Il_unp_M_T{i,j}=Il_unp_S;
   Ir_unp_M_T{i,j}=Ir_unp_S;
   Il_lin_M_T{i,j}=Il_lin_S;
   Ir_lin_M_T{i,j}=Ir_lin_S;
   mu_M_T{i,j}=mu;
   sigma_M_T{i,j}=sigma;
   sigma_S_M_T{i,j}=sigma_S;
   kR_M_T{i,j}=kR;
   end
end
cam_zen1=cam_zen_T(1:length(cam_zen_T));
cam_az1=cam_az_T(1:length(cam_az_T));
%%
cam_zen=cam_zen1';
cam_az=cam_az1';
%%%%For polar plot (in origin)
AoP_Pol=reshape(cell2mat(AoP_M_T)',1,length(cam_az)*length(cam_zen))';
DoP_Pol=reshape(cell2mat(DoP_M_T)',1,length(cam_az)*length(cam_zen))';
DoLP_Pol=reshape(cell2mat(DoLP_M_T)',1,length(cam_az)*length(cam_zen))';
DoCP_Pol=reshape(cell2mat(DoCP_M_T)',1,length(cam_az)*length(cam_zen))';
Q_Pol=reshape(cell2mat(Q_M_T)',1,length(cam_az)*length(cam_zen))';
U_Pol=reshape(cell2mat(U_M_T)',1,length(cam_az)*length(cam_zen))';
if error==1
   er_DoP_Pol=reshape(cell2mat(er_DoP_M_T)',1,length(cam_az)*length(cam_zen))';
```

```
    er_AoP_Pol=reshape(cell2mat(er_AoP_M_T)',1,length(cam_az)*length(cam_zen))';
    er_Q_Pol=reshape(cell2mat(er_Q_M_T)',1,length(cam_az)*length(cam_zen))';
    er_U_Pol=reshape(cell2mat(er_U_M_T)',1,length(cam_az)*length(cam_zen))';
else
    er_DoP_Pol=zeros(length(cam_az)*length(cam_zen),1);
    er_AoP_Pol=zeros(length(cam_az)*length(cam_zen),1);
    er_Q_Pol=zeros(length(cam_az)*length(cam_zen),1);
    er_U_Pol=zeros(length(cam_az)*length(cam_zen),1);
end
mu_Pol=reshape(cell2mat(mu_M_T)',1,length(cam_az)*length(cam_zen))';
sigma_Pol=reshape(cell2mat(sigma_M_T)',1,length(cam_az)*length(cam_zen))';
sigma_S_Pol=reshape(cell2mat(sigma_S_M_T)',1,length(cam_az)*length(cam_zen))';
I_Pol=reshape(cell2mat(I_M_T)',1,length(cam_az)*length(cam_zen))';
for m1=1:length(cam_az)
    for m2=1:length(cam_zen)
        cam_zen_Pol(m2+length(cam_zen)*(m1-1))=cam_zen(m2);
        cam_az_Pol(m2+length(cam_zen)*(m1-1))=cam_az(m1);
    end
end
cam_zen_Pol=cam_zen_Pol';
cam_az_Pol=cam_az_Pol';
V_M_T=V_M_T';
V_Pol=V_M_T(:);
un_normalized_I=un_normalized_I';
un_normalized_I_pol=un_normalized_I(:);
un_normalized_Q=un_normalized_Q';
un_normalized_Q_pol=un_normalized_Q(:);
un_normalized_U=un_normalized_U';
un_normalized_U_pol=un_normalized_U(:);
un_normalized_V=un_normalized_V';
un_normalized_V_pol=un_normalized_V(:);
end
```

APPENDIX H

MATLAB CODE TO SIMULATE POLARIZATION PATTERNS IN RECEIVER TUBES

Main Code:

```
%%Use skylight as light source [different S at different position on sky]
clc
clear all
close all

lamda=530e-9; %Wavelength
n_medium=1; %index of surrounding medium i.e air
a=35e-9; %radii of the particle
k0=2*pi/lamda;
m=1.72+0.65i;
x=k0*a*n_medium;

%%Date
month=11;
day=12;
year=2021;
%Time of day
hour=16;
minute=05;
second=05;
time=hour+(minute/60)+(second/60/60); % in hour (11:30am is 11.5hr)

%%%Position on earth; Church Parking
phi_lat=33.419541;
phi_lon=-111.929530;
GMT=-7;
DLS=0;

[theta,phi] = sun_position(month,day,year,time,phi_lat,phi_lon,GMT,DLS);
theta;
cam_az_T=0:1:360;
cam_zen_T=0:1:90;
[cam_az,cam_zen,I_source_Pol,Q_source_Pol,U_source_Pol,V_source_Pol,cam_zen_Po
l,cam_az_Pol]=get_skylight_Rayleigh_Scat(theta,phi,cam_az_T,cam_zen_T);
S_source=[I_source_Pol Q_source_Pol  U_source_Pol  V_source_Pol];
source_angles=[cam_az_Pol cam_zen_Pol];

%% Tube phi and theta coverage
number_of_tubes=5;
start_phi=184; %always the left end from looking direction
end_phi=174; %always the right end from looking direction
phi_rng=abs(start_phi-end_phi);
start_theta=80;
end_theta=90;
```

```
each_tube_covers_phi=abs(end_phi-start_phi)/number_of_tubes; %in degrees
phi_Tube=start_phi:-each_tube_covers_phi:end_phi;
phi_mid_temp=movmean(phi_Tube,2);
phi_tube_mid_pos=phi_mid_temp(2:end);
how_many_steps_in_one_degree=10;
angle_res=1/how_many_steps_in_one_degree;
phi_image=start_phi:-angle_res:end_phi;
%Left side of tube
phi_tube_left=0;
for i=1:number_of_tubes
   [new]=select_data_within_limit_less_or_equal_az(phi_image',phi_Tube(i));
   [new]=select_data_within_limit_greater_or_equal_az(new,phi_tube_mid_pos(i));
   phi_tube_left(length(phi_tube_left):length(phi_tube_left)+length(new)-1)=new;
end
%Right side of tube
phi_tube_right=0;
for i=1:number_of_tubes
   [new]=select_data_within_limit_less_or_equal_az(phi_image',phi_tube_mid_pos(i));
   [new]=select_data_within_limit_greater_or_equal_az(new,phi_Tube(i+1));
   phi_tube_right(length(phi_tube_right):length(phi_tube_right)+length(new)-1)=new;
end
theta_image=start_theta:angle_res:end_theta;
%% Polarization mapping
   mid_phi_direction_for_sources=(start_phi+end_phi)/2+180;
   if mid_phi_direction_for_sources>360
      mid_phi_direction_for_sources=mid_phi_direction_for_sources-360;
   end
   how_many_sources=20*50;
%% Left side of Tube
I_left_total=zeros(length(phi_tube_left),length(theta_image));
Q_left_total=zeros(length(phi_tube_left),length(theta_image));
U_left_total=zeros(length(phi_tube_left),length(theta_image));
V_left_total=zeros(length(phi_tube_left),length(theta_image));
if mid_phi_direction_for_sources>=0+phi_rng/2 &&
mid_phi_direction_for_sources<180+phi_rng/2
   phi_left_source_start=start_phi;
   phi_left_source_temp=359;
   phi_left_source_end=mid_phi_direction_for_sources;
   ratio_R_left=abs(phi_left_source_start-phi_left_source_temp)/180;
   how_many_sources_L1=ceil(ratio_R_left*how_many_sources);
   how_many_sources_L2=how_many_sources-how_many_sources_L1;
   rand_poses_left1=sort(randi([find(source_angles(:,1)==phi_left_source_start,1)
(find(source_angles(:,1)>phi_left_source_temp,1)-1)],how_many_sources_L1,1));
   phi_left_source_all(1:how_many_sources_L1,1)=source_angles(rand_poses_left1,1);
   theta_left_source_all(1:how_many_sources_L1,1)=source_angles(rand_poses_left1,2);
   S_source_left(1:how_many_sources_L1,:)=S_source(rand_poses_left1,:);
```

```
    rand_poses_left2=sort(randi([find(source_angles(:,1)==0,1)
(find(source_angles(:,1)>phi_left_source_end,1)-1)],how_many_sources_L2,1));
phi_left_source_all(how_many_sources_L1+1:how_many_sources,1)=source_angles(ran
d_poses_left2,1);
theta_left_source_all(how_many_sources_L1+1:how_many_sources,1)=source_angles(ra
nd_poses_left2,2);
S_source_left(how_many_sources_L1+1:how_many_sources,:)=S_source(rand_poses_le
ft2,:);
else
    phi_left_source_start=start_phi;
    phi_left_source_end=mid_phi_direction_for_sources;
    rand_poses_left=sort(randi([find(source_angles(:,1)==phi_left_source_start,1)
(find(source_angles(:,1)>phi_left_source_end,1)-1)],how_many_sources,1));
    phi_left_source_all=source_angles(rand_poses_left,1);
    theta_left_source_all=source_angles(rand_poses_left,2);
    S_source_left=S_source(rand_poses_left,:);
end
%%
parfor i=1:length(phi_left_source_all)
    %Source direction
    phi_source_left=phi_left_source_all(i);
    theta_source_left=theta_left_source_all(i);
    S=S_source_left(i,:)';
    [~,~,~,I_left,Q_left,U_left,V_left] =
get_tube_pol_map(phi_tube_left,theta_image,phi_source_left,theta_source_left,m,n_med
ium,x,k0,S);
    I_left_total=I_left_total+I_left;
    Q_left_total=Q_left_total+Q_left;
    U_left_total=U_left_total+U_left;
    V_left_total=V_left_total+V_left;
end
I_left_total_normalized=I_left_total./I_left_total;
Q_left_total_normalized=Q_left_total./I_left_total;
U_left_total_normalized=U_left_total./I_left_total;
V_left_total_normalized=V_left_total./I_left_total;
DoP_T_left=sqrt(Q_left_total_normalized.^2+U_left_total_normalized.^2+V_left_total_
normalized.^2)./I_left_total_normalized;
DoLP_T_left=sqrt(Q_left_total_normalized.^2+U_left_total_normalized.^2)./I_left_total
_normalized;
DoCP_T_left=V_left_total_normalized./I_left_total_normalized;
for v=1:length(phi_tube_left)
    parfor w=1:length(theta_image)
AoP_T_left(v,w)=0.5*myatand_0to180(U_left_total_normalized(v,w),Q_left_total_norm
alized(v,w)); %for 0 to 180 degree
    end
end
```

```
%% Right side of Tube
I_right_total=zeros(length(phi_tube_right),length(theta_image));
Q_right_total=zeros(length(phi_tube_right),length(theta_image));
U_right_total=zeros(length(phi_tube_right),length(theta_image));
V_right_total=zeros(length(phi_tube_right),length(theta_image));
if mid_phi_direction_for_sources>=0+phi_rng/2 &&
mid_phi_direction_for_sources<180+phi_rng/2
   phi_right_source_start=mid_phi_direction_for_sources;
   phi_right_source_end=end_phi;
   rand_poses_right=sort(randi([find(source_angles(:,1)==phi_right_source_start,1)
(find(source_angles(:,1)>phi_right_source_end,1)-1)],how_many_sources,1));
   phi_right_source_all=source_angles(rand_poses_right,1);
   theta_right_source_all=source_angles(rand_poses_right,2);
   S_source_right=S_source(rand_poses_right,:);
else
   phi_right_source_start=mid_phi_direction_for_sources;
   phi_right_source_mid=359;
   phi_right_source_end=end_phi;
   ratio_R_right=abs(phi_right_source_start-phi_right_source_mid)/180;
   how_many_sources_R1=ceil(ratio_R_right*how_many_sources);
   how_many_sources_R2=how_many_sources-how_many_sources_R1;
   rand_poses_right1=sort(randi([find(source_angles(:,1)==phi_right_source_start,1)
(find(source_angles(:,1)>phi_right_source_mid,1)-1)],how_many_sources_R1,1));
phi_right_source_all(1:how_many_sources_R1,1)=source_angles(rand_poses_right1,1);
theta_right_source_all(1:how_many_sources_R1,1)=source_angles(rand_poses_right1,2)
;
   S_source_right(1:how_many_sources_R1,:)=S_source(rand_poses_right1,:);
   rand_poses_right2=sort(randi([find(source_angles(:,1)==0,1)
(find(source_angles(:,1)>phi_right_source_end,1)-1)],how_many_sources_R2,1));
phi_right_source_all(how_many_sources_R1+1:how_many_sources,1)=source_angles(ra
nd_poses_right2,1);
theta_right_source_all(how_many_sources_R1+1:how_many_sources,1)=source_angles(
rand_poses_right2,2);
S_source_right(how_many_sources_R1+1:how_many_sources,:)=S_source(rand_poses_
right2,:);
end
%%
parfor i=1:length(phi_right_source_all)
   %Source direction
   phi_source_right=phi_right_source_all(i);
   theta_source_right=theta_right_source_all(i);
   S=S_source_right(i,:)';
[~,~,~,I_right,Q_right,U_right,V_right] =
get_tube_pol_map(phi_tube_right,theta_image,phi_source_right,theta_source_right,m,n_
medium,x,k0,S);
```

```
I_right_total=I_right_total+I_right;
Q_right_total=Q_right_total+Q_right;
U_right_total=U_right_total+U_right;
V_right_total=V_right_total+V_right;
end
I_right_total_normalized=I_right_total./I_right_total;
Q_right_total_normalized=Q_right_total./I_right_total;
U_right_total_normalized=U_right_total./I_right_total;
V_right_total_normalized=V_right_total./I_right_total;
DoP_T_right=sqrt(Q_right_total_normalized.^2+U_right_total_normalized.^2+V_right_t
otal_normalized.^2)./I_right_total_normalized;
DoLP_T_right=sqrt(Q_right_total_normalized.^2+U_right_total_normalized.^2)./I_right
_total_normalized;
DoCP_T_right=V_right_total_normalized./I_right_total_normalized;
for v=1:length(phi_tube_right)
   parfor w=1:length(theta_image)
AoP_T_right(v,w)=0.5*myatand_0to180(U_right_total_normalized(v,w),Q_right_total_n
ormalized(v,w)); %for 0 to 180 degree
   end
end
%% Full tube mapping
DoLP_tube_all=zeros(length(phi_image),length(theta_image));
AoP_tube_all=zeros(length(phi_image),length(theta_image));
DoCP_tube_all=zeros(length(phi_image),length(theta_image));
for i=1:number_of_tubes
   parfor cphil=1:length(phi_tube_left)
      left_pos_temp(cphil)=find(phi_image==phi_tube_left(cphil));
   end
   left_pos=left_pos_temp(1:end-1);
   parfor cphir=1:length(phi_tube_right)
      right_pos(cphir)=find(phi_image==phi_tube_right(cphir));
   end
   DoLP_tube_all(left_pos,:)=DoLP_T_left(1:end-1,:);
   AoP_tube_all(left_pos,:)=AoP_T_left(1:end-1,:);
   DoCP_tube_all(left_pos,:)=DoCP_T_left(1:end-1,:);
   DoLP_tube_all(right_pos,:)=DoLP_T_right(1:end,:);
   AoP_tube_all(right_pos,:)=AoP_T_right(1:end,:);
   DoCP_tube_all(right_pos,:)=DoCP_T_right(1:end,:);
end
%% Left side of tube
figure
contourf(phi_tube_left,theta_image,real(AoP_T_left)',500,'edgecolor','none');
colorbar
colormap(parula(512))
xlabel('camra__azimuth');
ylabel('camera__zenith');
```

```
title('AoP')
set(gca, 'FontSize', 20);

figure
contourf(phi_tube_left,theta_image,real(DoLP_T_left)',500,'edgecolor','none');
colorbar
colormap(parula(512))
xlabel('camra__azimuth');
ylabel('camera__zenith');
title('DoLP')
set(gca, 'FontSize', 20);

%% Right side of tube
figure
contourf(phi_tube_right,theta_image,real(AoP_T_right)',500,'edgecolor','none');
colorbar
colormap(parula(512))
xlabel('camra__azimuth');
ylabel('camera__zenith');
title('AoP')
set(gca, 'FontSize', 20);

figure
contourf(phi_tube_right,theta_image,real(DoLP_T_right)',500,'edgecolor','none');
colorbar
colormap(parula(512))
xlabel('camra__azimuth');
ylabel('camera__zenith');
title('DoLP')
set(gca, 'FontSize', 20);

%% Full tube
figure
contourf(phi_image,theta_image,real(AoP_tube_all)',500,'edgecolor','none');
cbh=colorbar;
colormap(jet(512))
xlabel('camra__azimuth');
ylabel('camera__zenith');
title('AoP')
set(gca, 'FontSize', 25);
set(gcf,'color','w');

figure
contourf(phi_image,theta_image,real(DoLP_tube_all)',500,'edgecolor','none');
cbh=colorbar;
colormap(jet(512))
```

```
xlabel('camra__azimuth');
ylabel('camera__zenith');
title('DoLP')
set(gca, 'FontSize', 25);
set(gcf,'color','w');

figure
contourf(phi_image,theta_image,real(DoCP_tube_all)',500,'edgecolor','none');
cbh=colorbar;
colormap(jet(512))
xlabel('camra__azimuth');
ylabel('camera__zenith');
title('DoCP')
set(gca, 'FontSize', 25);
set(gcf,'color','w');


        %%%%%%%%%% Functions used in the main code%%%%%%%%%%

function [DoLP_T,AoP_T,DoCP_T,I,Q,U,V] =
get_tube_pol_map(phi_tube_right,theta_image,phi,theta,m,n_med,x,k0,S)
%% Get the rotation matrix
%Looking or camera direction
phi_S=phi_tube_right;
theta_S=theta_image;
L_sigma_S=cell(length(phi_S),length(theta_S));
L_sigma=cell(length(phi_S),length(theta_S));
for i=1:length(phi_S)
   for j=1:length(theta_S)
      %As some of the phi_S are problematic (produces invalid rotation angles) we try to
calculate close to these values
      %but not exactly at those values of phi_S
      if phi_S(i)==phi
         phi_S(i)=phi_S(i)+0.001;
      elseif phi_S(i)==phi-180
         phi_S(i)=phi_S(i)+0.001;
      elseif theta_S(j)==0
         theta_S(j)=theta_S(j)+0.001;
      elseif theta_S(j)==90
         theta_S(j)=theta_S(j)-0.001;
      end
      mu(i,j)=(acosd(sind(theta)*sind(theta_S(j))*cosd(phi_S(i)-
phi)+cosd(theta)*cosd(theta_S(j))));
      [L_sigma_S{i,j}, L_sigma{i,j}]=get_rotation_matrix(phi,theta,phi_S(i),theta_S(j),
mu(i,j));
   end
```

```
end
scattering_angle=mu;
%% Single Mie Scattering
for w=1:length(phi_S)
   for v=1:length(theta_S)
u=cos(pi/180*scattering_angle(w,v));
%% Mie Scattering cross section
result=mie(m/n_med,x);
qsca=result(5);
%% S1 and S2
result_s1_s2=mie_S12(m/n_med, x, u);
S1=result_s1_s2(1);
S2=result_s1_s2(2);
S1_T(w,v)=S1;
S2_T(w,v)=S2;
%% Mie Scattering Matrix
M11=2*pi/k0^2/qsca*(abs(S1)^2+abs(S2)^2);
M12=2*pi/k0^2/qsca*(abs(S2)^2-abs(S1)^2);
M33=2*pi/k0^2/qsca*(S2*conj(S1)+S1*conj(S2));
M34=2*pi/k0^2/qsca*(S2*conj(S1)-S1*conj(S2))*1i;
M=[M11 M12 0 0
   M12 M11 0 0
   0 0 M33 M34
   0 0 -M34 M33];
%%
S_S=(L_sigma_S{w,v}*M*L_sigma{w,v})*S; % scattering in air (skylight)
S_S_unnormalized=S_S;
S_S=S_S./S_S(1);
DoP= sqrt(S_S(2)^2+S_S(3)^2+S_S(4)^2)/S_S(1);
DoLP=sqrt(S_S(2)^2+S_S(3)^2)/S_S(1);
DoCP=S_S(4)/S_S(1);
%angle of polarization
AoP=0.5*myatand_0to180(S_S(3),S_S(2)); %for 0 to 180 degree
DoLP_T(w,v)=DoLP;
DoCP_T(w,v)=DoCP;
AoP_T(w,v)=abs(AoP);
cam_phi_polar(w,v)=phi_S(w);
cam_theta_polar(w,v)=theta_S(v);
I(w,v)=S_S_unnormalized(1);
Q(w,v)=S_S_unnormalized(2);
U(w,v)=S_S_unnormalized(3);
V(w,v)=S_S_unnormalized(4);
   end
end
end
```