Imaging Optical Scatter in Advanced LIGO

Adalyne Cummins, Mentors: Joseph Betzwieser and Ryan DeRosa

LIGO-T1600388

July 27, 2016

Abstract

Due to surface imperfections, some laser radiation will deviate from its original, desired path—making the light disperse and appear as a deterministic angular distribution of intensity rather than a single, sharp reflection. A percentage of light is thus lost, or scattered away from the specular beam, causing a loss in energy and thereby decreasing the sensitivity of the laser. Concerns over the apparent deterioration of power stemming from the Pre-Mode Cleaner (PMC) led to an investigation focusing on characterizing this scatter by calibrating a CCD and modeling a bidirectional scattering distribution function of the data. In addition to the PMC, various optics (test masses and mode cleaners) were also analyzed, this time using a High Dynamic Range (HDR) computer program that takes images at different exposures and merges them together. By comparing HDR data with theoretical models, one can easily indicate any beam irregularities, thus my ongoing research further develops a better understanding of the total amount of noise optical scattering inflicts, and finds a way to decrease its influence on the detection of gravitational waves.

Contents

1 Introduction

Due to surface, bulk, and/or contaminate imperfections, some laser radiation will deviate from its original, desired path—making the light disperse and appear as a deterministic angular distribution of intensity rather than a single, sharp reflection. A percentage of light is thus lost, or scattered away from the specular beam, causing a loss in energy and thereby decreasing the sensitivity of the laser. My research addresses this scatter from various optics in order to gain a better understanding of the total amount of noise it inflicts.

Optical scattering is one of several possible areas (such as diffraction deterioration and coating absorption) that attribute to the Laser Interferometer Gravitational Wave Observatory's (LIGO) power losses. These usually stem from either imperfections on the four main test masses—40 kg mirrors—or internal scattering of devices such as the pre-mode mirrors and cleaners. The loss of power leads to losses in LIGO's infrared (IR) laser light, precision, and energy—thereby disrupting and further complicating detection.

1.1 Objectives

I will be focusing on just two areas of research regarding optical scattering. The primary topic was first addressed in a LIGO Livingston Observatory (LLO) and LIGO Hanford Observatory (LHO) wide meeting, where concerns arose over the apparent deterioration of power stemming from the Pre-Mode Cleaner (PMC), a bowtie cavity, illustrated in Figure 1, in the Pre-Stabilized Laser (PSL) system designed to strip higher-order modes and filter out radio frequencies from the main beam [1]. At the start of detection, less than 1% of power was originally lost in the PMC—now it is approximately 6%. My investigation of the PMC thereby focuses on characterizing this optical scatter by digitally capturing it, analyzing the images using geometric techniques, and modeling a Bidirectional Scattering Distribution Function (BSDF) of the data.

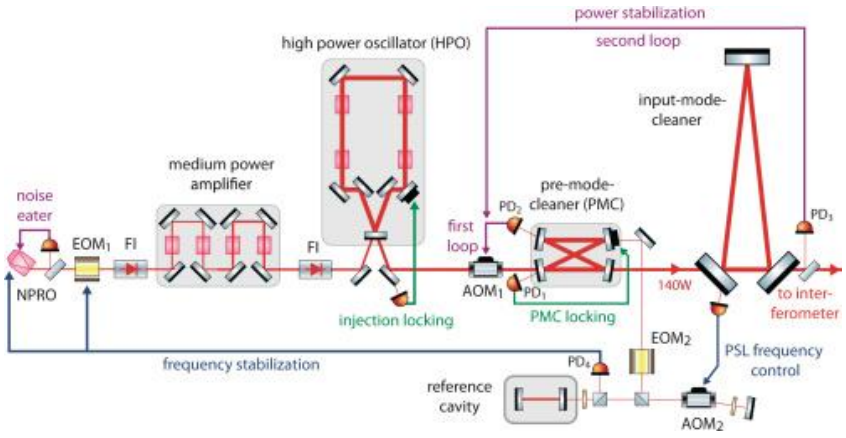

Figure 1: Schematic of the PSL system [1]

The second area of research began in the attempt to find missing power originating from various optics including the four primary test masses and the mode cleaner mirrors. Instead of capturing images of these masses at a single exposure, which limits one's ability to see a greater dynamic range of luminosity, we can make a program that takes photographs at different exposure times and then merges these images together. This is called High Dynamic Range (HDR), and it is a much more comprehensive look at scatter because of its accuracy in reproducing detailed images. My work with HDR imaging thus aims on using HDR to map the laser beam at various locations along its path, compare it to theoretical models, and note any irregularities in its shape.

2 Pre-Mode Cleaner Scatter

2.1 Background

In order to measure optical scattering, one needs to measure the power of the observable scatter at a resolute area, distance, and angle from the source. My main method of determining this lost power is accomplished using a camera that converts pixels into power through

$$Power = \frac{Intensity(counts) \times constant}{Exposure(\mu s)} \tag{1}$$

where Intensity is the sum of relevant pixel values on the charge-coupled device (CCD), and Exposure is the length of time the camera's shutter is open, thus determining the amount of light that reaches the image sensor. The constant in Equation 1 is the Calibration Factor (CF), in μs×W/counts, which converts the CCD's reading from counts into watts.

To analyze these images, a BSDF is calculated depending on whether the observed scatter is a result from reflected (BRDF) or transmitted (BTDF) light. These functions describe the angular distribution of radiation scattered from a particular surface, generally using

$$BSDF(\theta,\phi) \equiv \frac{P_s/\Omega}{P_i cos(\theta_s)} \tag{2}$$

where P_s is the scattered power reaching the CCD (watts), P_i is the incident power on the surface (watts), and Ω is the solid angle of collection, which is simply the area of the detector divided by the squared distance from the CCD to the surface [2]. Finally, the $\cos(\theta_s)$ term corrects for the obliquity factor, thus making this formula account for a certain projected area. Equation 2 only describes a very small fraction of the total scatter, thus we must integrate over the half hemisphere above the surface multiplied by the cosine obliquity factor:

$$\Sigma Scatter = \int_0^{2\pi}\int_0^{\frac{\pi}{2}} BSDF \times cos(\theta) sin(\theta) d\theta d\phi \tag{3}$$

The BSDF has units of str^{-1} or inverse steradians.

2.2 GigE Calibration Factor

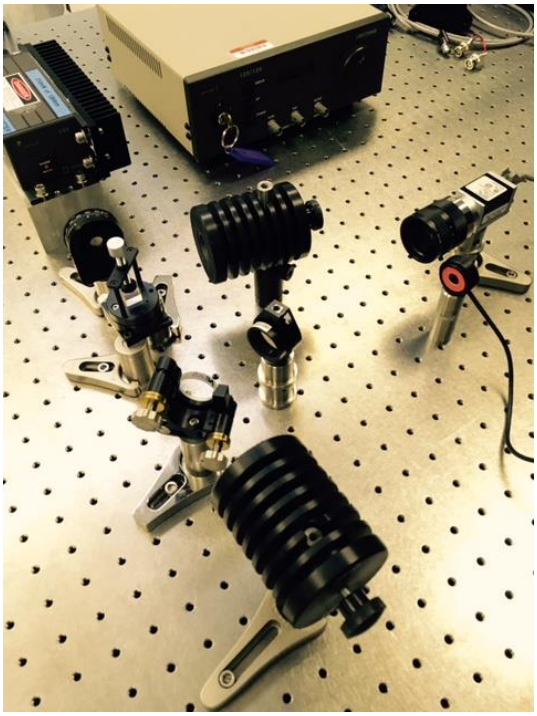

Figure 2: CF raw experimental setup, at the top we have the laser and its power source, then,

Figure 3: Total CF setup. In addition to the optics displayed in Figure 2, we have (top to bottom)

following the beam path, the ½ λ plate, two PBS and their corresponding dumps, an ND filter, and finally the CCD. The small red sensor is a photodiode.

two laser alignment devices—one is a thermal sensitive paper the other is a focusable scope—1064nm safety goggles, PM100D Power Meter and sensor, an extra ND filter, and a PC.

To find CF, I used the equipment illustrated in Figure 2 and Figure 3, which was set up in the optics lab. First a 1064nm laser powered from a 200W source is directed towards a ½ waveplate, which allows one to alter the polarization state of the linearly polarized beam, thereby granting total control of the power incident on the camera. Then the laser moves through two polarizing beam splitters (PBS), a cube/lens used to transmit p-polarized waves while reflecting s-polarized waves, the latter of which are dumped. Next, the laser makes its way through a Neutral Density (ND) filter, a wavelength absorber that dims the entire CCD. The camera used was a Basler Ace acA640-100gm with a 25mm F1.3 macro lens. By virtually controlling the camera's exposure from a PC, manually adjusting the focus of the beam, and physically varying the power of the laser, I was able to take pictures with the following settings: data format = mono8, gain = 100 (minimum value), black level = 64 (default). After discarding all pixels less than 2 in order to eliminate blooming (or a glow due to focusing errors), I calculated the average number of counts for a single pixel using a modified Equation 1 that is solved for the constant. Table 1 displays the data:

Table 1: Data for CF

Transmitted (Counts)	**Total** (Counts)	**Power** (μW)	**Exposure** (μs)	**Power** (μW) × **Exposure** (μs)
1437	143700	1.24	40	1.70
3106	310600	1.24	80	2.00
4893	489300	1.24	120	2.17
6621	662100	1.27	160	2.31
8470	847000	1.25	200	2.40
781	7810000	0.36	240	86
1014	10140000	0.39	280	109
1163	11630000	0.37	320	118
1328	13280000	0.37	360	133

1624	16240000	0.37	440	163
1804	18040000	0.36	500	180
3649	36490000	0.37	1000	370
6126	61260000	0.39	1600	624

The numbers in green are the trials taken under the influence of an ND 20 filter, while the rest were taken with an ND 40. Two filters were used because as the exposure increased, more pixels were being saturated, thus the CCD had to be relieved of light. The ND 40 has an optical density (OD) of 4, which only transmits 10^{-2} % of scattered light, while the ND 20, with an OD of 2, transmits 1%. For the last column, I had to take the log of the green power × exposure values in order to compensate for the change in OP values. Below, in Figure 4, is a graph of Exposure*Power vs. Counts/Pixel, where the total number of pixels in each image is 325546, that more clearly illustrates the linear CF.

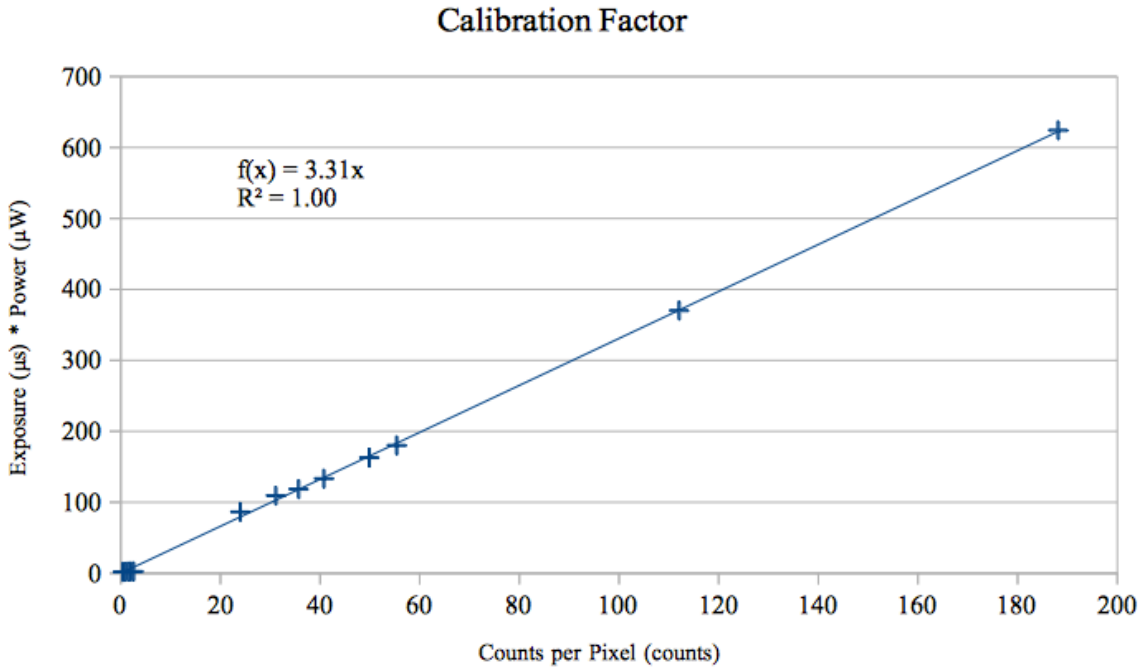

Figure 4: Graph of CF

Finally, in order to get the CF from this graph, one must take the slope of 3.31 µs*µW/counts and multiply it by the limits of the devices used in the experiment. These limits would be the area of a single pixel (ASP) divided by the area of the photodiode (APD). The ASP relies on the

Basler Ace specifications, 5.6µm × 5.6µm, while the APD depends on the Ø9.5mm diameter of the PD sensor. Converting the units of µW to the more convenient W, we obtain a CF equal to 1.46×10^{-9} µs*W/counts. This value is in the range of calibration factors measured previously for GigE cameras: in August 2013, CF = 2.19×10^{-9} µs*W/counts for the 640-100um model and a black level of 0 [3]; in May 2014, CF = 1.6×10^{-10} µs*W/counts for the 640-120gm model with a black level of 0 [4]; and in July 2014, CF = 1×10^{-10} µs*W/counts for the 640-100gm model [5].

2.3 PMC Data

Once the camera was calibrated, I went into the PSL with the Basler Ace (same settings and lens as before) to investigate any potential scatter. By placing an ND 40 filter to decrease the saturation of the sensitive CCD, I was able to take 8 bit pictures from two primary angles of the PMC: one of the main output port (MOP) taken from the table, the other of the input taken by hand.

2.3.1 Main Output Port Analysis

I took four pictures at different exposures of the MOP. As an example, Figure 5 is a raw .tiff image, and Figure 6 is a MATLAB zoom of Figure 5 using imread(data) where data = (85:415, 180:430).

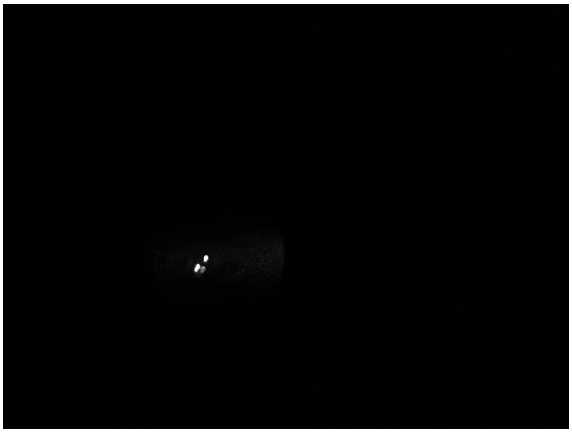

Figure 5: MOP, 1600µs, 40 ND filter

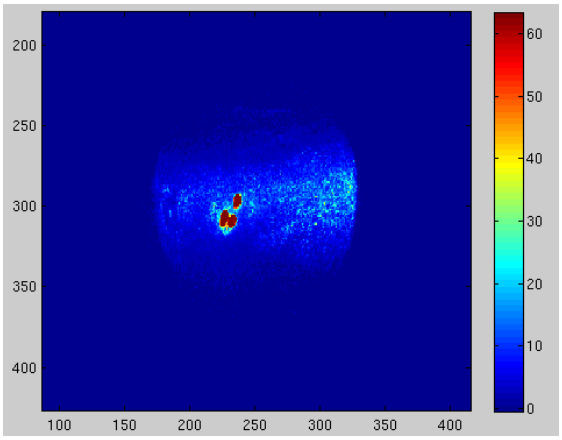

Figure 6: MOP, 1600µs, 40 ND, zoom

Table 2 shows the conversion of the data from image form to numeric display. There are two numbers in parentheses—these are the number of pixels that are considered saturated (or greater than 250 counts) despite the heavy filter. Due to these pixels, the calculated power will be an underestimate of the true power incident on the CCD. The pixel count was made using

sum(data(:)), where all pixels less than 2, data(find(data < 3)) = 0, were discarded just like in the calibration method. Thus, the average scattering power transmitted is 6.60×10^{-6} W. Theoretically, the power should be the same for each exposure trial, but naturally we have some measurement error, with a standard deviation of 1.4×10^{-6} W. The BTDF is 0.0001 sr^{-1}, with only 0.0035W scattering from the MOP (0.01%).

Table 2: CCD power data for the MOP

Exposure (μs)	**Pixels** (counts)	**Power Transmitted** (W)
40	1568	5.74×10^{-6}
80	3235	5.92×10^{-6}
160 (3)	6563	6.01×10^{-6}
1600 (54)	95252	8.72×10^{-6}

2.3.2 Input Analysis

The input of the PMC was harder to obtain pictures of due to the lack of space on the PSL table and restrictive movement of the laser beam path. I was, however, able to take 4 pictures without much saturation. Table 3 the data, taken with the same considerations as the MOP, just without the influence of an ND filter.

Exposure (μs)	**Pixels** (counts)	**Power Reflected** (W)
4	198186	7.26×10^{-5}
10	205842	3.02×10^{-5}
20 (8)	227463	1.67×10^{-5}
30 (25)	272117	1.33×10^{-5}

Table 3: CCD power data for the input port

The average power reflected is 3.32×10^{-5} W, with a standard deviation of 2.7×10^{-5} W, although the error is much larger due to the fact that the camera was held by hand and perhaps slightly out of focus. These measurements were taken 12 inches from the port, 3 inches to the side, and 3 inches up. The incident power was 35W, thus, the BRDF of the input port is 0.0012 str^{-1}, which

considers that approximately half of the optic was captured on the CCD. This is only 0.12% of the incident power, with a total power scattered of the input port being 0.043W.

2.4 PMC Conclusions

In total, only 0.047W, out of 35W, was found lost to optical scattering from the two main ports. This 0.13% loss means that the 6% deterioration of power must be coming from a source inside the PMC, not from imperfections in the ports and mirrors. Nevertheless, the 0.047W loss could be attributed to the build up of residuals on the windows where the main beam enters and leaves. This was the case in 2014 at LLO, when power of the beam reflected from the PMC window increased several watts after nine months of operation [6]. A fraction of the incident power could also be lost on the way towards the PMC, such as by distortion off other optics or reflections from imperfect surfaces.

3 Scatter in High Dynamic Range

3.1 Background

Most cameras take photographs with a limited exposure range, resulting in a loss of detail (such as in shadows or highlights). The objective of HDR is to present a range of light that is very similar to the luminance experienced by the human eye, which adjusts constantly over a broad range of exposures. One can replicate this visual system by using a camera that captures and then a program that combines images of several different narrow-range exposures of the same object at very small (0.5 second) consecutive intervals. Figure 7 illustrates the amount of depth an HDR image can have, the depth I will utilize in characterizing scatter from various optics.

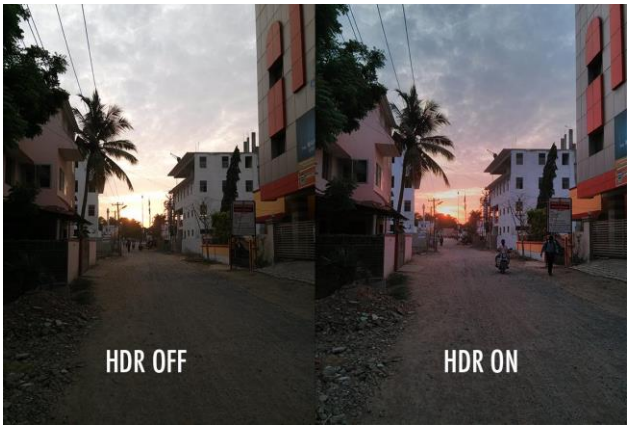

Figure 7: HDR comparison using a Oppo Find 5 [7]

3.2 Modeling

The initial, working HDR code was written by Matthew Evans and Stefan Ballmer of LHO [8] and then sent to LLO to be readjusted and tested. There are two main programs: mexp_test.py [9], and make_HDR_image_test.m [10]. The former code snaps six pictures at six different, increasing exposures (e.g. 30, 100, 300, 1000, 30000, 100000) 0.5 seconds apart from each other. When activated from terminal, using ./mexp_test.py, the program records the individual snapshots and saves each in a file named after the images' exposure, optic, and exact time the picture was taken. The pictures are stored in cd /data/camera_images/test, which is easily accessible to the latter program.

The first function in make_HDR_image.m stores information in three different variables: imHDR, im_data, and g. g is simply a list of the exposures, while im_data finds the files and creates an 480 by 640 by (numel(g) - 1) array of the .tiff figures using imread. imHDR merges the individual images together by taking the image at the lowest exposure (imA) and the second lowest exposure (imB) and their corresponding *valid* pixels (*valid* meaning that the program finds all saturated pixels (imx > 250) and potential blooming noise (imx < 50) and ignores these values) to compute mean(imB(nn) ./ imA(nn)), where nn are valid pixels. This mean value is the gain (electrons per counts), or the magnitude of the amplification in the image, and is very important in comparing two images because without it, comparing imA's number of counts against imB's is meaningless [11]. Thus, the new image, imAB, is equal to imB divided by the gain correction. The program then loops until all 5 images are merged together in HDR fashion.

Below are two images of the laser beam after it emerges from the Mode Cleaner Mirrors (MC1, MC2, and MC3), which act collectively as an optical resonator that filters out junk light before the laser reaches the arms. These were taken on July 26th 2016 under the channel MC_TRANS with a IR laser beam:

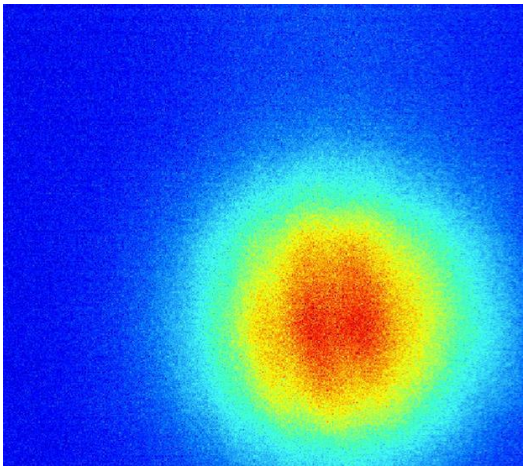

Figure 8: 2D HDR image looking directly at the CCD of a circular symmetric wavefront.

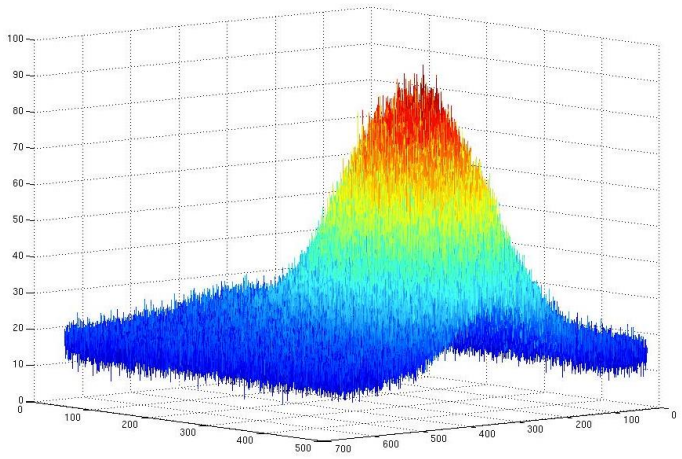

Figure 9: 3D HDR image of a TEM_{00} laser

3.3 Theory

MC_TRANS is the cleanest beam at LIGO because it is assumed that the laser has an ideal Gaussian profile corresponding to the theoretical TEM_{00} mode. TEM_{00} (or transverse electromagnetic mode) is the lowest order mode that forms a radially symmetrical intensity distribution given by

$$I(r) = Ce^{\frac{-2r^2}{\omega^2}} \tag{4}$$

where r is the distance from the center of the beam, C is the amplitude, and ω is the radius where the intensity has fallen $1/e^2$ of its axial value [12]. Since $I \propto P^2$ and C = the gain of imHDR, we can form a function that represents the theoretical TEM_{00} model.

$$F(x) = C(e^{\frac{-(x-x_O)^2}{\omega^2}})^2 \tag{5}$$

is the theoretical equation, where $2r^2 = (x - x_o)^2$ in order to compensate for transverse motion. Below is code translating Equation 5 into a form the computer can process

```
Function F = myfun(x, xdata)
        F = x(1) × (exp( - (xdata – x(2))).^2./x(3).^2)).^2
                                                        | and...
xdata = [1 : 640]
x0 = [max(max(imHDR)), 300, 150]                        | guess parameters
```

```
myfun(x0, data)
cut = imHDR(150, :)
x = lsqcurvefit(@fun, x0, xdata, squeeze(cut))          | best fit of data
theoretical = myfun(x, xdata)                           | theoretical curve
```

From this we get the nice gaussian distribution in red below, which is comparable to the cut imHDR plot in blue.

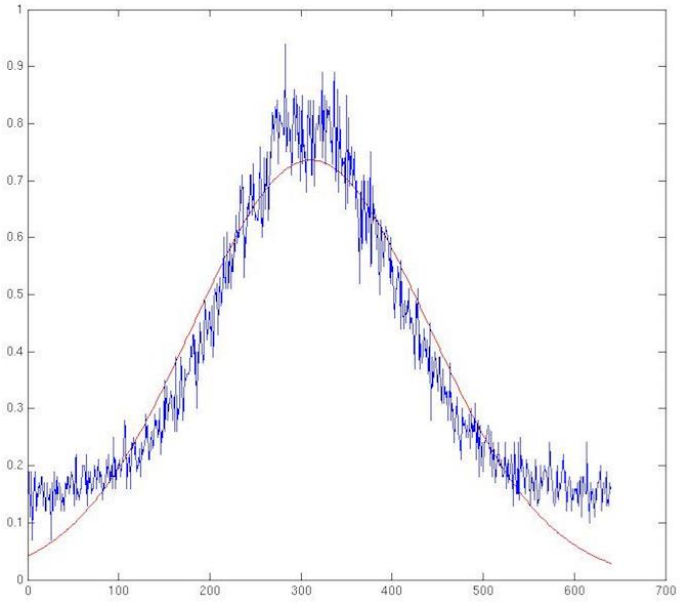

Figure 10: Cut of the imHDR and theoretical imHDR curve

3.3.1 Theory vs. TEM_{00}

From Figure 10, we can pull up a residual plot of the theoretical data minus the actual images taken from MC_TRANS. Here we see a non-random w-shaped curve, which suggests that there is a well defined interference pattern influencing the CCD. This perhaps can be explained by internal reflections inside the camera, which cause an addition or subtraction of light waves, resulting in the observed fringing pattern. To mitigate this problem and improve the HDR program, further researchers could tilt the camera off the normal while keeping the center of the image sensor stationary, so that focus and framing are unchanged. The tilt would adjust the angle of incidence and optical path inside the camera, thus lessening the influence of the fringes.

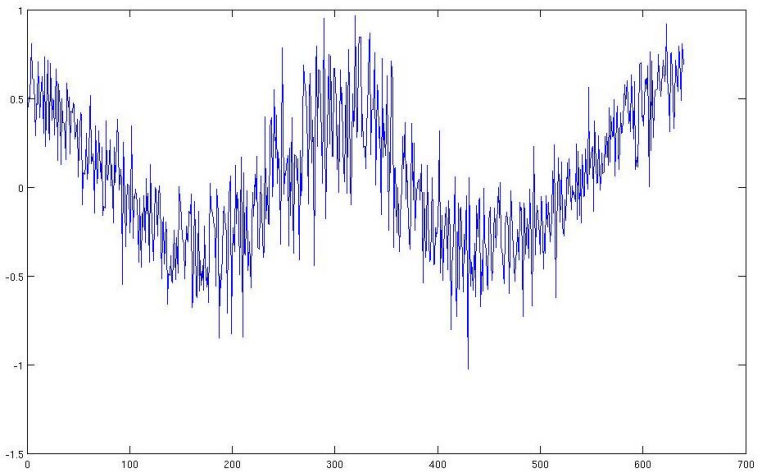

Figure 11: Residual Plot of MC_TRANS

3.4 Input Test Mass X

After examining the theoretical case of MC_TRANS, we took images of the IR laser from the Input Test Mass X (ITMX). The following figures show these images, which are dominated by high saturating spots.

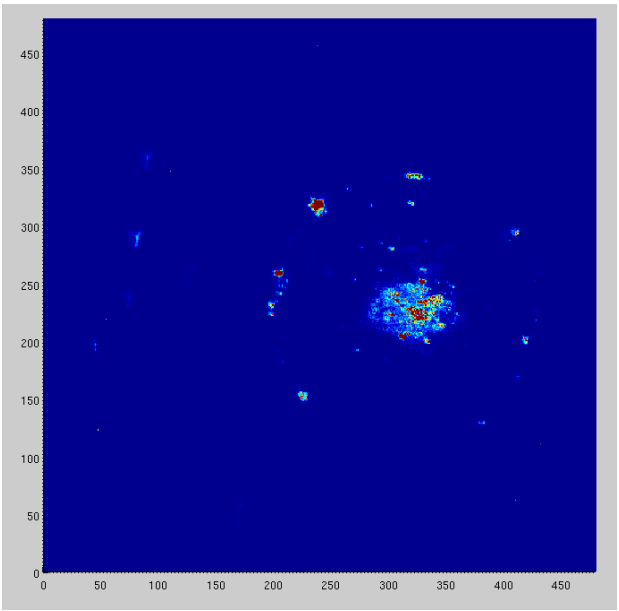

Figure 12: 2D image of ITMX

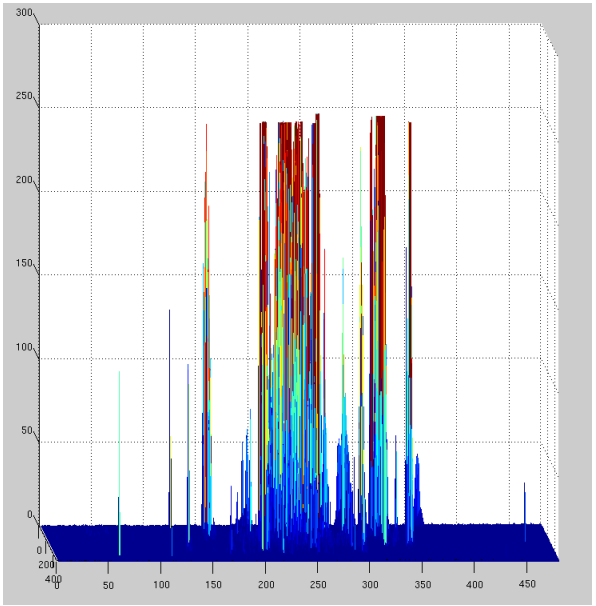

Figure 13: 3D image of ITMX

These photos were taken at g = [2000, 4000, 8000, 16000, 32000, 64000] and look very different from the theoretical TEM_{00} images. Around the incident laser beam, spots of scatter are observable in the ITMX mirror. These irregularities might be produced from residuals, like the

MC_TRANS, and/or mirror coating errors. If we take a cut through the center of the main beam path, we do obtain a plot that resembles a gaussian curve:

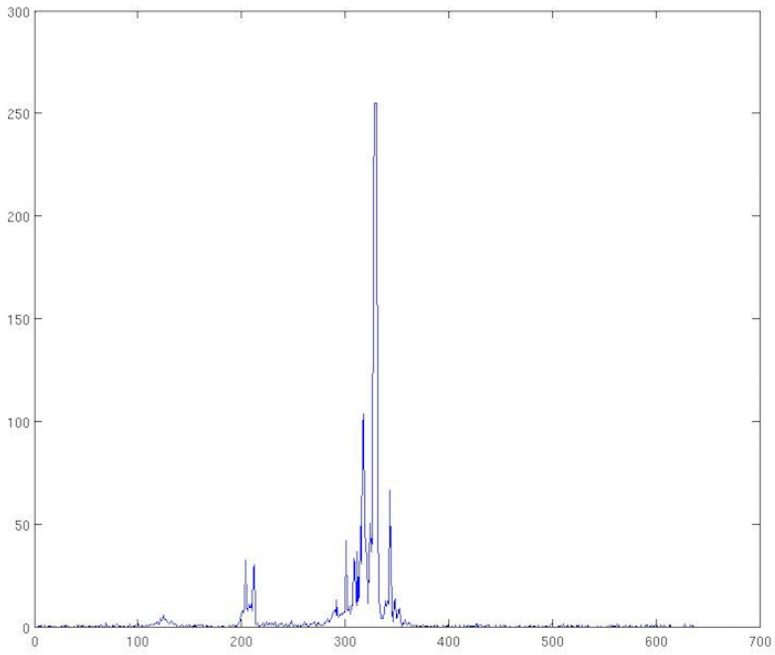

Figure 14: (250, :) cut of ITMX

While the HDR program is not perfect, and does require more improvement and further tweaking, it easily and conventionally displays images of various optics in LIGO that show the shape and characteristics of the laser beam. This is very useful in diagnosing potential problems with the test masses, and supplementing other optical research to come.

4 Conclusions and Further Research

Although a massive flood hindered most of the last week of my stay at LLO, I believe good progress was made. With the PMC, I was able to capture the scatter from two major ports, analyze the images by converting them to numerical data that describe the total power lost, and estimate the total integrated scatter stemming from the PMC. I found that only 0.13% of power was lost, meaning that the rest of the 6% must be lost to an internal source. With imaging scatter by HDR programming, I was able to help make a working model using python and matlab, make a theoretical comparison between MC_TRANS' TEM_{00} mode and a gaussian curve, and note irregularities in the ITMX having to do with residuals on the mirror.

Further research and experimentation with the HDR code, including removing any grid interference patterns on the CCD, smoothing out the scaling of the scatter, and testing the other test mass mirrors would be very useful and productive next step. For the PMC, opening the tank and collecting data from inside perhaps might solve the missing 6% of power lost. Also, by clearing more space around the very crowded PMC tank and taking more photos of all four ports at various angles can offer a more complete picture of reflective and transmitted light loss.

5 References

[1] LIGO-P1400177-v5. 2014. LIGO Scientific Collaboration. "Advanced LIGO," LIGO Scientific Collaboration, http://arxiv.org/pdf/1411.4547v1.pdf.

[2] John Stover. Optical Scattering: Measurement and Analysis. The International Society for Optical Engineering. Second Edition, 1995

[3] Eleanor King, David Feldbaum. Calibration factor for GiGE cameras. https://alog.ligo-la.caltech.edu/aLOG/index.php?callRep=8321. LLO Logbook, Accessed: 07-27-2016

[4] David Feldbaum. Recalibration of the GigE cameras sensitivity. https://alog.ligo-la.caltech.edu/aLOG/index.php?callRep=12947. LLO Logbook, Accessed: 07-20-2016

[5] Hunter Rew, Joseph Betzwieser, and David Feldbaum. Analysis of etmy scatter. https://alog.ligo-la.caltech.edu/aLOG/index.php?callRep=13414. LLO Logbook, Accessed: 07-01-2016.

[6] Jan Põld. Design, Implementation and Characterization of the Advanced LIGO. Leibniz University of Hannover University. 2014.

[7] Bharadwaj Chandramouli. Oppo Find 5 Review. June 9th 2013. Forearena. http://www.fonearena.com/blog/71624/oppo-find-5-review.html. Accessed: 07-21-2016

[8] Matthew Evans and Stefan Ballmer. High Dynamic Range Images. https://alog.ligo-wa.caltech.edu/aLOG/index.php?callRep=28454. LHO Logbook, Accessed: 07-26-2016

[9] Adalyne Cummins, Joseph Betwieser, Matthew Evans, and Stefan Ballmer. Appendix A.

[10] Adalyne Cummins, Joseph Betwieser, Matthew Evans, and Stefan Ballmer. Appendix B.

[11] Michael Richmond. CCD Gain. RIT. http://spiff.rit.edu/classes/phys445/lectures/gain/gain.html. Accessed: 07-27-2016

[12] IDEX Optics and Photonics. Gaussian Beam Optics. Marketplace. https://marketplace.idexop.com/store/SupportDocuments/All_About_Gaussian_Beam_OpticsWEB.pdf. Accessed: 07-27-2016

6 Acknowledgments

I really want to thank my mentors Joe and Ryan for being patient with me along the way, and for answering all of my questions. I also want to thank Caltech and their SURF program for giving me the amazing opportunity to work at LIGO.

7 Appendix A

mexp_test.py

```
# starts snapping test mass camera pictures every 10 seconds
# pictures end up in /data/camera_images/test
# for MC_TRANS

import time
import numpy as np
import ezca as ez
ezca = ez.Ezca()

file_tag = ' '
cam_num_list = ['MC_TRANS']
cam_prefix = 'CAM-'
cam_list = []

exposure_time_list = [30, 100, 300, 1000, 3000, 10000]
```

```
exposure_name_list = ['A', 'B', 'C', 'D', 'E', 'F']
exposure_list = zip(exposure_time_list, exposure_name_list)
exposure_final_list = []

# make list of camera names
for cam_num in cam_num_list:
        cam_list.append(cam_prefix + cam_num);

# record exposure times
print ' '
print 'Initial exposure times'
for cam in cam_list:
        exp_val = ezca[cam + '_EXP']
        print('{0} initial exposure is {1}').format(cam, exp_val)
        exposure_final_list.append(exp_val)
exposure_final = zip(cam_list, exposure_final_list)

# take pictures at various exposures
for exposure_time, exposure_name in exposure_list:
        print ' '
        print 'Exposure ' + exposure_name

        # set file names and exposure times
        for cam in cam_list:
                ezca[cam + '_FILE'] = 'test/multiexp_' + file_tag + exposure_name
                ezca[cam + '_EXP'] = exposure_time
                time.sleep(0.5)
                if (not (ezca[cam + '_FILE'] == 'test/multiexp_' + file_tag + exposure_name)) or
                        (not (ezca[cam + '_EXP'] == exposure_time)):
                        print 'File or Exp not updated yet'
                        time.sleep(2)

        # pause for effect, then take pictures
        for cam in cam_list:
                while (ezca[cam + '_SNAP'] == 1):
        time.sleep(0.5)
        ezca[cam + '_SNAP'] = 1
        time.sleep(0.5)

        # print sum
        for cam in cam_list:
                print('{0} sum is {1}').format(cam, ezca[cam + '_SUM'])

# reset exposure times
print ' '
print 'Reset exposure times'
for cam, exposure_time in exposure_final:
```

```
        ezca[cam + '_EXP'] = exposure_time

print ' '
print 'Done.'
```

8 Appendix B

make_HDR_image.m

```
function [imHDR, im_data, g] = make_HDR_image(optic, t0)
        % [ix40, ix40_data] = make_HDR_image('MC_TRANS', '2016-07-16-00-21-45');
        % for MC_TRANS
        % look for files around the specified time
        file_tags = {'A', 'B', 'C', 'D', 'E'};
        num_file = numel(file_tags);

        im_data = zeros(480, 640, num_file);
        for n = 1: num_file
                im_data(:, :, n) = load_image(optic, t0, file_tags{n});
        end

        g = [30, 100, 300, 1000, 3000, 10000];
        imHDR = im_data(:, :, 1);
        for n = 1: (num_file - 1)
                % g(n) = get_gain(im_data(:, :, n), im_data(:, :, n + 1)) * g(n - 1);
                imHDR = merge_images(imHDR, im_data(:, :, n + 1), g(n + 1) / g(1));
        end
end

function im_data = load_image(optic, t0, file_tag)
        % make file tag
        switch optic
                case 'MC_TRANS'
                        file_mid = 'MC_TRANS'
        end
        file_head = ['/data/camera_images/test/multiexp_' file_tag '_' file_mid '_'];
        file_tail = '.tiff';

        % make date stamp
        sec_per_day = 24 * 60 * 60;
        date_format = 'yyyy-mm-dd-hh-MM-SS';
        date0 = datenum(t0, date_format) - 10 / sec_per_day;

        % find file
        im_data = [];
        for n = 1:60
```

```
            file_date = datestr(date0 + n / sec_per_day, date_format)
            file_name = [file_head file_date file_tail]
            if exist(file_name, 'file') == 2
                % im_data = rot90(double(imread(file_name)), 2);
                im_data = flipud(double(imread(file_name)));
                break
            end
        end

        if isempty(im_data)
            disp(file_name)
            error('Image not found!')
        end
end

function imAB = merge_images(imA, imB, gainBA)
        % gainBA is roughly mean(imB(nn) ./ imA(nn))
        % where nn are valid pixels
        %
        % it is assumed that gain > 1 (e.g., imB has higher gain
        % or longer exposure time and imA)

        imA(find(imA < 3)) = 0
        imB(find(imB < 3)) = 0

        % start with imB with gain correction
        imAB = imB / gainBA;

        % use points from imA where imB is saturated
        nn = find(imB > 250);
        imAB(nn) = imA(nn);
end

function gain = get_gain(imA, imB)

        % vectorize data
        imA = imA(:);
        imB = imB(:);

        % find valid points
        nn = find(imA > 50 & imA < 250 & imB < 250 & imB > 50);
        imA = imA(nn);
        imB = imB(nn);

        % compute rough ratio (biased, but good starting point)
        gain = mean(imB ./ imA);
```

```
        % make better cut to reduce bias
        radAB = sqrt((imB / gain).^2 + imA.^2);
        nn = find(radAB > 100 & radAB < 130);
        imA = imA(nn);
        imB = imB(nn);

        % final gain
        gain = mean(imB ./ imA);
end
```