# Chapter 1

# Measuring Beam Waist

Sometimes it's important to measure the waist of your laser beam. This explains exactly how I do it since there is plenty of opportunity for messing it up.

According to Wikipedia [1], the profile for the intensity of the beam is just

$$I(r,z) = I_0 \left(\frac{w_0}{w(z)}\right)^2 \exp\left(\frac{-2r^2}{w^2(z)}\right) \tag{1.1}$$

Extrapolating this to when there are two different waists in different directions:

$$I(r,z) = (\text{constants}) \exp\left(-2\frac{x^2}{w_x^2(z)} - 2\frac{y^2}{w_y^2(z)}\right). \tag{1.2}$$

Now, total power ($P$) can be obtained by integrating the the intensity:

$$P = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} (\text{constants}) \exp\left(-2\frac{x^2}{w_x^2(z)} - 2\frac{y^2}{w_y^2(z)}\right) dxdy \tag{1.3}$$

$$P = \frac{\pi}{2}(\text{constants}) w_x(z) w_y(z) \tag{1.4}$$

Now, we figure out that those constants will be

$$(\text{constants}) = \frac{2P}{\pi w_x(z) w_y(z)}. \tag{1.5}$$

Also, those constants represent the maximum intensity that occurs within the beam (i.e. when $x = y = z = 0$, or in other words, at the center of the narrowest part of the beam.)

Now, to take measurements with a knife, I block some fraction of the beam with a translation stage. Thus, if, for example, the knife is mounted vertically, I get the total power in the beam from $x = -\infty$ to $x_0$ where $x_0$ is the location of the knife. Thus, if I plot this power as a function of the spatial position of the knife, I expect in the $x$ direction to get:

[1] this is just common knowledge, right?

$$P(x_0) = \int_{\infty}^{\infty} \int_{-\infty}^{x_0} \frac{2P}{\pi w_x(z) w_y(z)} \exp\left(-2\frac{x^2}{w_x^2(z)} - 2\frac{y^2}{w_y^2(z)}\right) dxdy \tag{1.6}$$

$$P(x_0) = \frac{P}{2}\left(\text{erf}\,\frac{\sqrt{2}x}{W_x} + 1\right) \tag{1.7}$$

So, now we have a function that we can fit. The code in masterFILEhoriz14Nov.m calculates the approximate beam waist based on a series of knife cuts. First, one loads the intensity (in arbitrary units) and the position (in the units that the waist will ultimately be given in) into a number of arrays named x1, x2, .. . etc. The user types the position of the knife along the waist (show picture) into the vector called positions.

The function d4sigma attempts to calculate the second moment width using just the raw data. This is notoriously inaccurate and also it is probably misnamed because it probably just calculates something that is different from the standard D4$\sigma$ by some factor of 2 or something. Also, we don't use this.

Ultimately, the program does a best fit approximation to a Gaussian for each beam profile. The function we fit to is contained in the file beamWaistCalculator, which takes as its arguments the raw position data and the raw intensity data. This function is

$$f(x) = a_1 \,\text{erf}\left(\frac{x - a_3}{a_2}\right) + a_4. \tag{1.8}$$

beamWaistCalculator thus returns a vector containing $(a_1 a_2 a_3 a_4)$. Thus, to get the beam waist, we will take $a_2$ and multiply by $\sqrt{2}$.

According to Siegman [**?**], the second moment width always propagates according to this equation:

$$W_x^2(z) = W_{0x}^2 + M_x^4 \left(\frac{\lambda}{\pi W_{0x}}\right)^2 (z - z_{0x})^2. \tag{1.9}$$

Of course, for a Gaussian beam, we recognize that allowing $M_x^2 \to 1$ gives us the equation for how the waist changes as it propagates.

This function then tries to fit to this:

$$W_x = a_1 \sqrt{1 + \left(\frac{(x - a_2)\lambda}{\pi a_1^2}\right)^2} \tag{1.10}$$

and this:

$$W_x = a_1 \sqrt{1 + (1 + a_3^2)\left(\frac{(x - a_2)\lambda}{\pi a_1^2}\right)^2}. \tag{1.11}$$

The first case we are forcing $M^2$ to be one. This gives us a worst-case scenario. A few test cases convinced us that if $M^2$ is higher than we think it is, we have a less-tightly focused

center. Thus, for the purposes of not putting too much power into the AOM, we go with the fit to the first equation.

In principle, we should be able to fit the calculated second moment widths to the equations above, but they don't work overly well. Thus, we usually use the second moment width of the Gaussian we fit it to.

Using this method, we have determined the waist to be

## 1.1 Using the Camera

We initially wanted to measure the $M^2$ of our beam. The measurements are hard to do accurately. We tried to use the camera and got some crap that didn't seem very meaningful.

### 1.1.1 The Camera is discovered to be quite non-linear

By comparing images of the same beam sent through different filters on the camera, we began to suspect that there was some kind of threshold of light below which the camera won't register anything. Additionally, we had some anecdotal evidence that the pixels act more or less independently.

To investigate this, we took several measurements of the same beam attenuated to different degrees using our polarizing beam cube and our two crappy waveplates setup[2] We took 20 measurements. Each time we measured the total intensity of the beam using a photodiode.

For the analysis, we assumed that the intensity of light on any given pixel was proportional to the overall light in the beam. Thus, for example, if we see that the pixel located at $(214, 442)$ has a pixel count of 127 in image 15, on which we measured the intensity to be proportional to 85.350, then we assume that we can compare it directly to a pixel located at the same location $(214, 442)$ on image 20 (which had an intensity proportional to 119.70). If the reading from the $(214, 442)$ pixel on image 20 was 155.46, we assume that a pixel count of 155.46 corresponds to $119.70/85.350 = 1.4025$ times as much intensity as a pixel count of 127.

We went through the 20 images like this: We selected some reference pixel. We then added that pixel to a master list. We then found the corresponding pixels in the other images whose position on the grid match that of the reference pixel. They were then added to the master list, making note of their position, count and the assumed ratio between their intensity and that of the reference pixel. Then, we select a random pixel from the master list and look for other pixels that have the same counts. These pixels are then added to the

[2]We need a better name for this technique, but the gist of it is that normally you can change the polarization of your light using one waveplate. However, if you order the wrong waveplate, you can't do that. However, if you order two of the wrong waveplate, you can set them both up so that you can at the very least have perfect control over how much light goes through your beam cube.

list. Each time we look at a pixel, we see if we can reference it to either add corresponding pixels from the other images to our list or to find other pixels with the same count.

At the end, we got this plot, which seems to validate the assumption of pixel independence.

We did a standard fit to a 4th order polynomial and got a reasonable function to correct for the pixel values. We were a little worried about bias in the selection of pixels and things like that. We wasted some time trying to figure out how to do this better.

We then inverted this function by using an interpolation algorithm.

Now, we have made a function that basically takes our image data and produces some figure of merit to tell us how our test function is doing at making all of the pixels have the right ratio. The way it works is this. Suppose that $M_{ij}$ is a big list of our pixels. $j$ runs from 1 to 20, corresponding to the 20 different images. $i$ represents all the pixels in our image[3]. Let $f(p)$ be the proposed function that takes as its argument the count on a given pixel and gives as its output something that scales with the actual intensity of the light. The figure of merit that we can use to optimize $f(p)$ is calculated as follows:

$$\text{Figure of merit} = \sum_{M_{ij}\neq 0} \left( \frac{f(M_{ij})/P_j}{\text{average of all nonzero } f(M_{ij})/P_j \text{ for the ith pixels}} - 1 \right)^2 \quad (1.12)$$

where $P_j$ is the power of the jth image.

First, act on the pixels with the proposed function. Then, scale them according to the intensity ratios (i.e. the ith pixel in image 1 gets divided by $P_1$ while the ith pixel in image 3 gets divided by $P_3$). Then, see by what fraction the intensity of that pixel differs from the average of the intensities calculated using data from the pixels corresponding to that one. If the function really were to find the ratio perfectly, then $f(M_{ij})/P_j$ should be exactly the same for the ith pixel from each of the j images.

Because the camera throws away information and has a minimum intensity threshold to register, we throw out any $M_{ij} = 0$. This way our function doesn't get penalized for showing a non-zero intensity for pixels that read out zero. The number of $M_{ij} = 0$ obviously doesn't change as we try different iterations of $f(M_{ij})$, so there's no way to game our little metric to make it not work.

We used this function to calculate what correction function to use. We did a couple of them and found great agreement with the inversion of the previous function–thus suggesting that our previous method was acceptable.

[3]Thus, if we are analyzing 8x8 bins, we should have 480/8*640/8=4800 pixels and i will go from 1 to 4800