Abstract

The current method for measuring profiles of proton bunches in accelerators is severely lacking. One must dedicate a great deal of time and expensive equipment to achieve meaningful results. A new method to complete this task uses a rotating mask with slots of three different orientations to collect this data. By scanning over the beam in three different directions, a complete profile for each bunch is built in just seconds, compared to the hours necessary for the previous method. This design was successfully tested using synchrotron radiation emitted by SPEAR3. The profile of the beam was measured in each of the three desired directions. Due to scheduled beam maintenance, only one set of data was completed and more are necessary to solve any remaining issues. The data collected was processed and all of the RMS sizes along the major and minor axes, as well as the tilt of the beam ellipse were measured.

INTRODUCTION

The Large Hadron Collider (LHC) is the premier high energy physics experiment, using some of the most advanced technology on the planet. This colossal machine, 27 kilometers around, now collides 3.5 TeV protons (7 TeV center of mass) and will collide at twice this energy in 2014, the highest collision energy ever achieved. At the opposite end of the scale spectrum, the proton beam is only a few microns across at an interaction point (IP, where the beams collide in the four detectors around the ring), requiring extremely precise control and measurements. It is, therefore, very useful to directly measure the beam cross-section. One would imagine that this task could prove very difficult, but with the clever application of a few pieces of basic hardware, one can easily achieve this goal at a low cost.

Outside the detectors, the transverse size of the beams is much larger. At the injection energy, 450 GeV, the beam size is approximately 1 mm RMS (root mean square, the standard deviation) and at full energy this size drops to 0.3 mm. As the protons fly around the enormous ring, they, like all charged particles, emit synchrotron radiation. When low energy charged particles are bent by an outside influence, such as a magnetic field, they emit radiation with an intensity that varies with the cosine of the angle to the direction of travel. However, when the particle is accelerated to near the speed of light, this wide spread is focused into a narrow, forward beam. The spectrum is also shifted to higher energies. For electrons the midpoint of the emission spectrum for a bending magnet is called the "critical frequency" follows:

$$\omega_c = \frac{3}{2}\gamma^3 \frac{c}{\rho} \qquad (1)$$

Where ω_c is the critical frequency, γ is the relativistic factor, c is the speed of light, and ρ is the radius of the bending magnet. The x-ray beam is commonly used for experiments in a wide range of disciplines, from condensed matter to biological studies, but synchrotron radiation, particularly the visible emission, is often used to understand the profile of the beam of particles that created it. Optics can image the synchrotron light onto an ordinary CCD to provide video of the transverse profile of the beam, but this requires electron beams or very high energy proton beams, and these still often need image intensifiers. Measurement of these profiles usually requires a camera with a gate faster than the spacing between bunches. The

camera is placed in the beam of synchrotron light, at an image point of the particle beam, and records the profile generated by each proton bunch. These bunches traveling around the ring are separated only by a few nanoseconds, requiring an incredibly fast shutter. In this very brief period of time, very little light is gathered by the camera. To compensate for this, many exposures are taken over many separate passes of the same bunch. This means that several seconds are necessary to obtain meaningful statistics. Since there are hundreds or even thousands of bunches in most accelerator rings, measuring a complete profile of every bunch can take several hours. The LHC is a particularly difficult case, with 2808 bunches spread over 89 microseconds with 25 nanoseconds between each bunch. With a typical integration time of two seconds per bunch, a full measurement of every bunch would take nearly two hours. This is obviously not ideal, since the bunch size can change drastically in that time.

One alternative to this method significantly cuts down on the time necessary to take data, but only provides a mathematical description of the bunch cross-sections, not a tangible image. By passing a rotating mask consisting of three slots of different orientation, one horizontal, vertical, and one at a 45° angle, in front of the beam, one can build an intensity profile in the three different dimensions. From the light that passes through the slot, one can find the Gaussian shape of the bunch that created it. With this information in three dimensions, the shape and tilt of the overall Gaussian ellipse can be calculated.

Each bunch returns to the optical system every 89 microseconds. During this time, the wheel has only moved a small amount. This provides a very large number of data points to reconstruct the shape of the bunch. Using simple computer software, one can index each point to a specific bunch and then reconstruct all of the bunches separately from a single set of three slots. This entire process can be conducted in less than a second, improving performance time by orders of magnitude.

MATERIALS AND METHODS

The main materials necessary for this project are a photomultiplier tube, rotating mask, and a driving motor. To test this method, visible light from the SPEAR3 storage ring at SSRL was

used in place of LHC synchrotron radiation. More details regarding these components are found below.

PHOTOMULTIPLIER TUBE

This project requires a small, fast photomultiplier tube to collect intensity data from bunches. One other necessary feature of this device is resistance to radiation damage. The entire apparatus being developed in this project will eventually sit in the tunnel at the LHC. This is a very extreme environment, with very high levels of ambient radiation from the accelerator. The current choice for PMT is the Hamamatsu R7400U. The "U" specifies a fused-silica window rather than glass to avoid radiation damage, since glass is easily darkened by the ambient radiation that is present in the LHC tunnel. This PMT is small, less than 20 mm. in any dimension, and also has a very fast response time, just below 2 ns[1]. Since the radiation being measured is entirely in the visible spectrum, no additional equipment, such as scintillators, is necessary.

ROTATING MASK

Another very important component of this project is the rotating mask. An earlier version of this component was designed and tested by another student working under Dr. Fisher in a previous SULI summer program[1]. The wheel has a radius of 100 mm with slots centered at 80 mm. There are four sets of three slots, each cut so that they will be at precisely the right orientation as they pass over the image of the beam, which intercepts the wheel at a 45° angle from its X or Y axis. The slots are laser cut to be approximately 15 microns in a very thin foil which is then sandwiched between two plates of aluminum to protect it. The design schematics for this component are shown in Fig. 1.

OSCILLOSCOPE

To record the data taken from a scan of the mask, an oscilloscope with a very long memory and high sample rate was used. The Tektronix DPO4104 was chosen on account of its wide range of features. This oscilloscope is able to scan five gigasamples per second and record ten million points in its memory. The high resolution, resulting from the sampling rate, allows for signals from individual bunches to be resolved it high detail. Since the signal resolution is so high, however, a large amount of memory is needed to record all the data from one slot.

MOTOR

The final mechanical component of this project is the motor that turns the mask. The motor chosen is the Maxon Sensor DC Motor 118748. This motor is then attached to a gear-head with a 53:1 reduction factor. This provides the motor with a huge range of speeds, allowing for a great deal of versatility in the number of data points taken. Additionally, a resolver head is attached to the rear of the motor which outputs angular position in the form of a cosine and sine signal. These two signals can be converted to an angular position using simple software to track which slot is providing a signal at a given time. An example of this output is shown in Fig. 2.

SSRL

Due to the high levels of ambient radiation in the LHC tunnel which this apparatus will eventually be used in, it is convenient to test the setup in a safer and more accessible environment. The Stanford Synchrotron Radiation Lightsource (SSRL) provides an excellent opportunity to test the behavior and diagnostic abilities of the rotating wheel system. At SSRL, electrons orbit a storage ring at 3 GeV, creating extremely intense beams of radiation. This radiation ranges from infrared to hard x-rays. For this experiment, the range of radiation was restricted to visible light, since that is the only type available for detection at the LHC. Although the energy range is vastly different, most other beam properties are similar between the LHC and SSRL. Both have bunches of particles that emit radiation, which is the central focus of this project. However, one major drawback is that SSRL has no signal digitizer that can record data from each bunch as it passes in real time. Therefore, at SSRL, the data must be recorded and processed at a later time. This results in a very large data file that is difficult to work with.

One should note that there are significant differences between the light measured at SSRL compared to the ultimate goal of diagnostics at LHC. The intensity and energy of the radiation produced at SSRL is orders of magnitude higher than that of the LHC. Since this light was so intense, several filters were necessary to prevent the PMT used in this project from being overwhelmed. The saturation of the PMT caused a major distortion in the output signal. This was caused by insufficient time between an successive signals for the capacitors within the

PMT to recharge. To remedy this issue, the rotational speed of the wheel was increased, causing the PMT to be exposed to light for a shorter time, the driving voltage of the PMT was decreased, reducing the output gain, and a filter was placed in the beam path, decreasing the amount of light being measured. All of these changes, although resolving the PMT discharge issue, significantly reduced the signal intensity. Therefore, an amplifier was used to magnify the output signal. Additionally, the time necessary for a bunch to travel around the storage ring at SSRL is approximately 100 times faster than at the LHC. This makes the signal analysis much more difficult because the peaks generated by each bunch are compressed very closely together.

METHODS

The test of this apparatus was conducted on the diagnostic beam line at SSRL. This is a beam line that is limited to only visible radiation. A large optical table was shared with two other experiments to provide a stable, organized test area. A light tight box was constructed around the mask and PMT housing to reduce ambient light. This setup is shown in Fig. 3. The light coming from the beam line was first focused with a large lens to a point approximately two meters away. This lens results in a demagnification of the image by a factor of 8. Along this axis, several mirrors were inserted at different points so each experiment could acquire the light as well as a neutral density filter to reduce the amount of light reaching the PMT, preventing overload. The light for this project was then passed through another lens to magnify the image by a factor of 3.25, resulting in a net magnification of 0.4063.

In order to obtain a precise focus and magnification measurement, an electronic camera was placed at the exact point where the mask would be located. A glass slide with a USAF resolution test pattern, consisting of a series of lines with different sizes, was placed in the path of the beam. Using software written in MatLab by Jeff Corbett, both the beam and this image were accurately focused. The test pattern also provided a very precise measurement of the magnification power of the optical set up.

After the focus and magnification were successfully measured, a light-tight enclosure was built to ensure that only photons from the synchrotron source were being recorded. The PMT was isolated in a specially designed box and mounted to an optical breadboard. The

motor and wheel were then carefully aligned so that the PMT would focus at 45° from the horizontal axis of the wheel. This ensured that the angles of the slots were exactly 0°, 90° and 45° when they passed in front of the PMT. Another layer of light reduction was then put in place using black cardboard and tape to further seal the PMT from ambient light. In addition to all of these measures, the room in which the experiment was conducted was also dark during data collection.

Finally, to collect data, a high voltage power supply applied -600 volts to the PMT. A DS 345 signal generator provided a 10 kHz sine wave signal to the motor resolver, allowing for position measurements. Four signals were then recorded using the oscilloscope: the resolver drive signal, the cosine output, the sine output, and the PMT signal. A slight modification to the wheel was made in the form of dark tape placed over all but one slot. The record stored by the oscilloscope was then limited to the time interval around the passage of just one slot. To resolve the fast PMT signals, the data was digitized at two gigsamples per second. A full wheel rotation would create a record of billions of points. By examining one slot at a time, the resolution was maintained while being limited by the oscilloscope's ten megasample memory. This insured that position of the slot was known very accurately, since the data was not being processed in real time. Under normal working conditions, the sampling rate at the LHC will be such that only the relevant data is accessed. However, at SSRL, this is not the case, so a large amount of extraneous data is recorded, making precise position measurements difficult.

The final step of this project was the development of a Python script to process these signals. The latest version of the script reads all of the data, and then finds the average angular velocity of the wheel using the cosine, sine, and resolver input signals. Since the cosine and sine signals are simply amplitude modulations of the drive signal, finding the peak of the drive signal and checking the value of the cosine and sine signals will give an accurate angular position.

The script then scans the PMT data for the peak generated by the timing bunch, which is a large, isolated bunch of electrons used for diagnostic purposes. By isolating each of the timing bunch peaks, a Gaussian profile is built that corresponds to that bunch only, allowing for the imaging of single bunches. This can be repeated with any of the bunches, but the timing

bunch is the simplest for testing purposes. The profile from each scan were then fit to find the size of the beam in real space.

RESULTS AND DISCUSSION

This project has successfully shown that this technique is a viable method for measurement of individual bunch profiles. A scan in each of the three axes was successfully obtained. From that data, the Gaussian profiles were fit and their sizes were obtained. Figure 4, 5, and 6 each display the scans after being converted into real space with position on the x-axis and intensity on the y-axis. The Y profile scan is particularly interesting due to the non-gaussian shape. This abnormality is most likely the result of improper focusing of the beam or aberrations caused by the optics used to focus the beam.

After fitting a Gaussian peak to each of these profiles, several beam parameters were extracted for each including height and σ. These values for each axis are shown in Table 1. The two sets of values for σ result from two different calculation methods. The first, labeled as 'automated', uses the motor resolver head signal to calculate angular velocity. Due to the sampling method used in this project discussed above, this has a large amount of error. One can see in Fig. 2 that the exact angular position is not well defined, so this measurement, especially when automated with computer software, is not particularly accurate. The second value uses a hard coded value for angular velocity. This value was measured using an oscilloscope to find the time necessary for a complete revolution of the wheel.

One should note the relatively large discrepancy between the expected values and those which were measured. This error results from diffraction effects of the beam as it passes through the numerous optical devices before being imaged as well as the incredibly high amount of precision necessary to properly focus the beam. A lens out of place by less than a millimeter can change the image size by an enormous magnitude. This is a result, however, of the current setup and not a defect with the apparatus.

ACKNOWLEDGEMENTS

I would like to thank my mentor, Alan Fisher, for allowing me to work on this project. It was a very valuable experience. I would also like to thank Jeff Corbett for assisting me in optical alignment when Alan was not available. Finally, I would like to thank the US Department of Energy for providing funding for this project.

REFERENCES

[1] Lee, Christopher J. *Bunch by Bunch Profiling with a Rotating X-Ray Mask*. Science Undergraduate Laboratory Internship (SULI), 22 Aug. 2007. Web. 15 Aug. 2011.

APPENDIX A: Figures and Tables

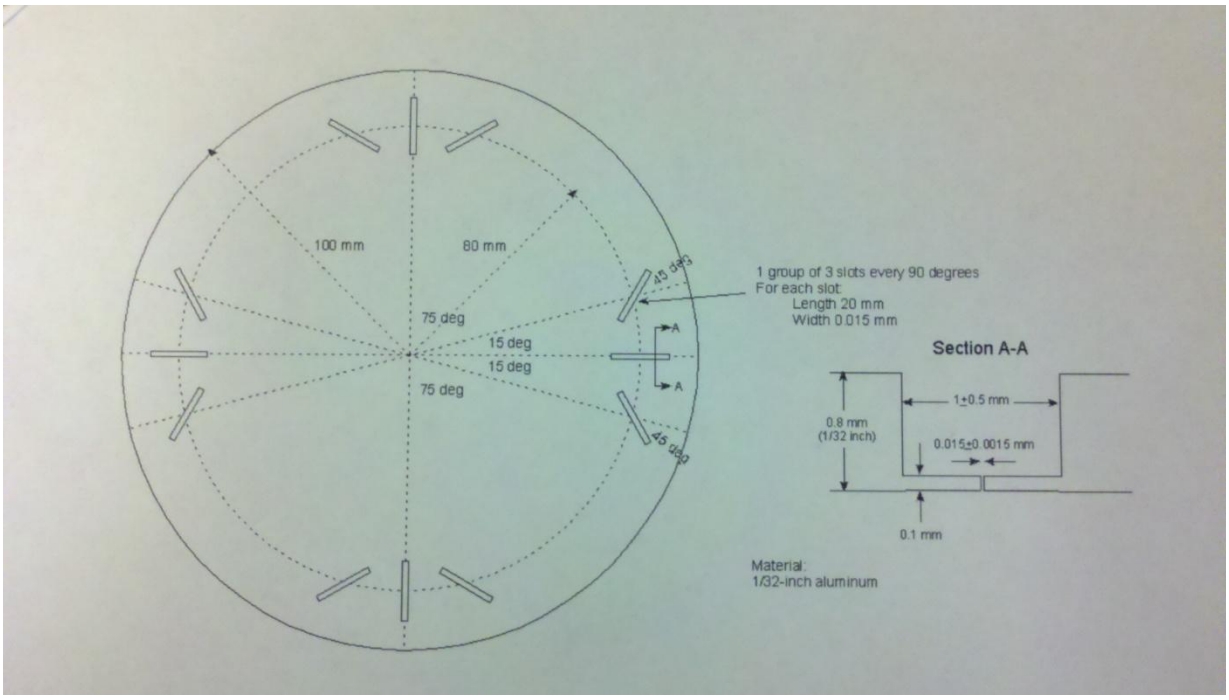

Figure 1: Design schematics for the rotating light mask. One can see four sets of slots at different angles, generating each of the three profiles.

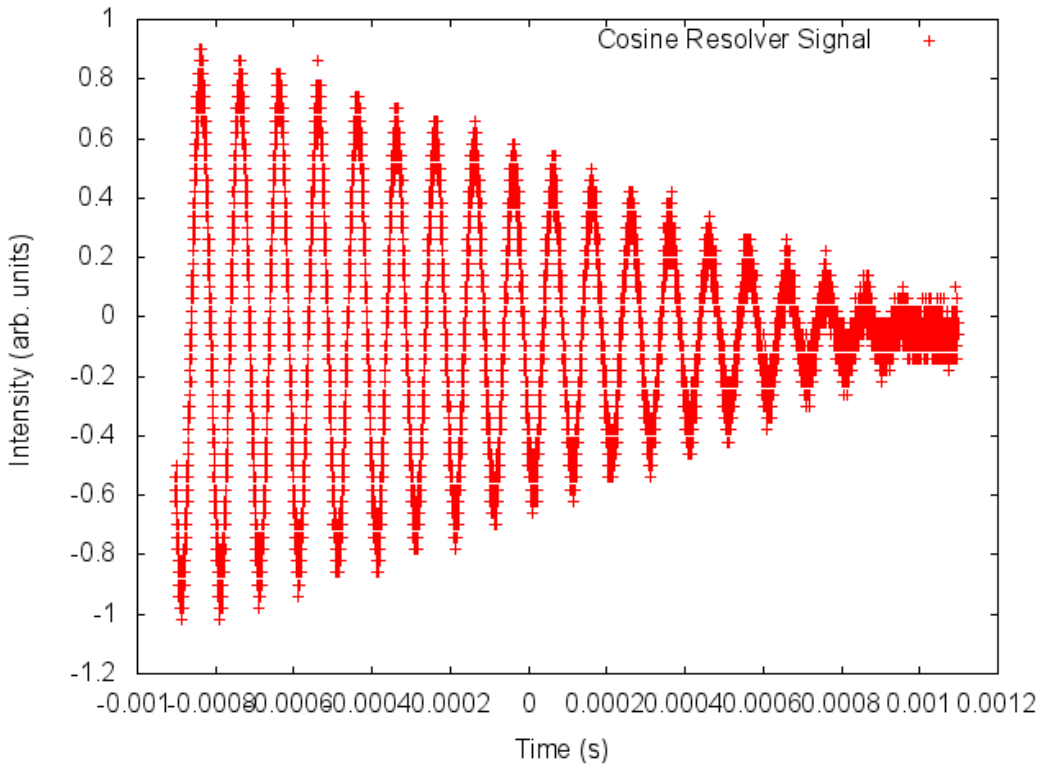

Figure 2: The cosine output from motor resolver for the X Profile scan. Note the envelope in which the 10 kHz resolver sits within.

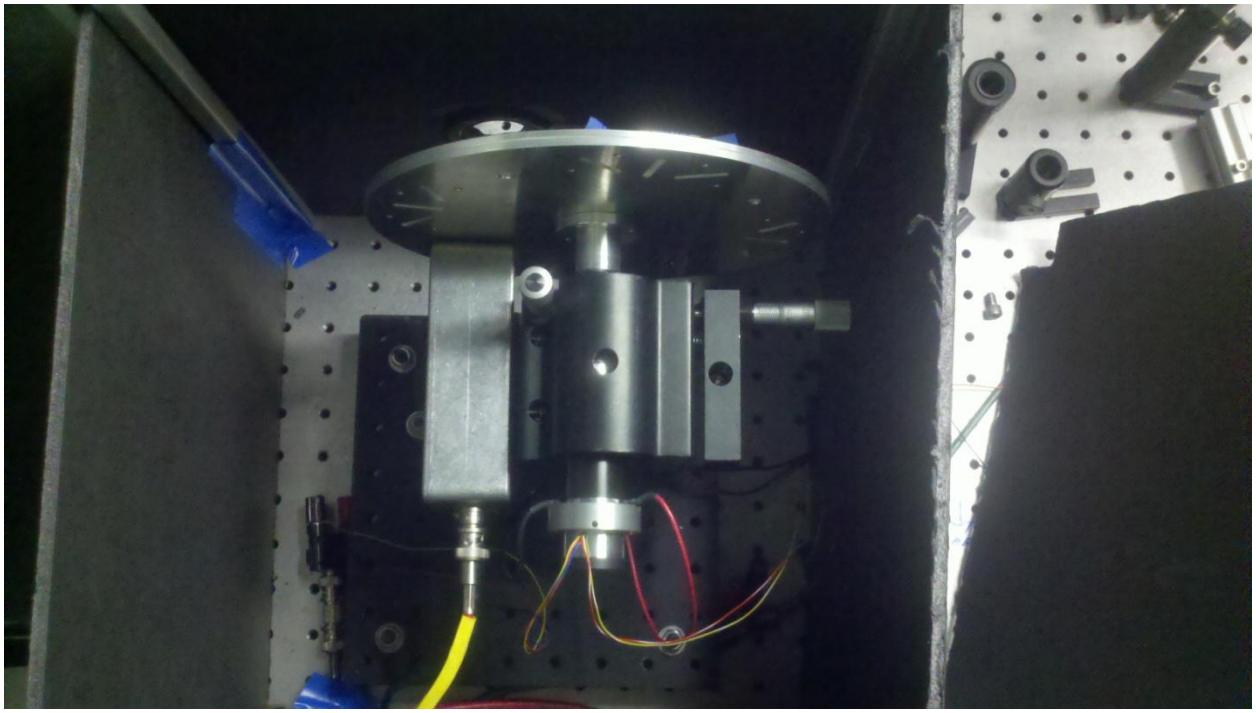

Figure 3: Light tight box setup. One can see the PMT box (large yellow cable) to the left of the motor and resolver outputs (thin, multi-colored wires) in the lower section of the image.

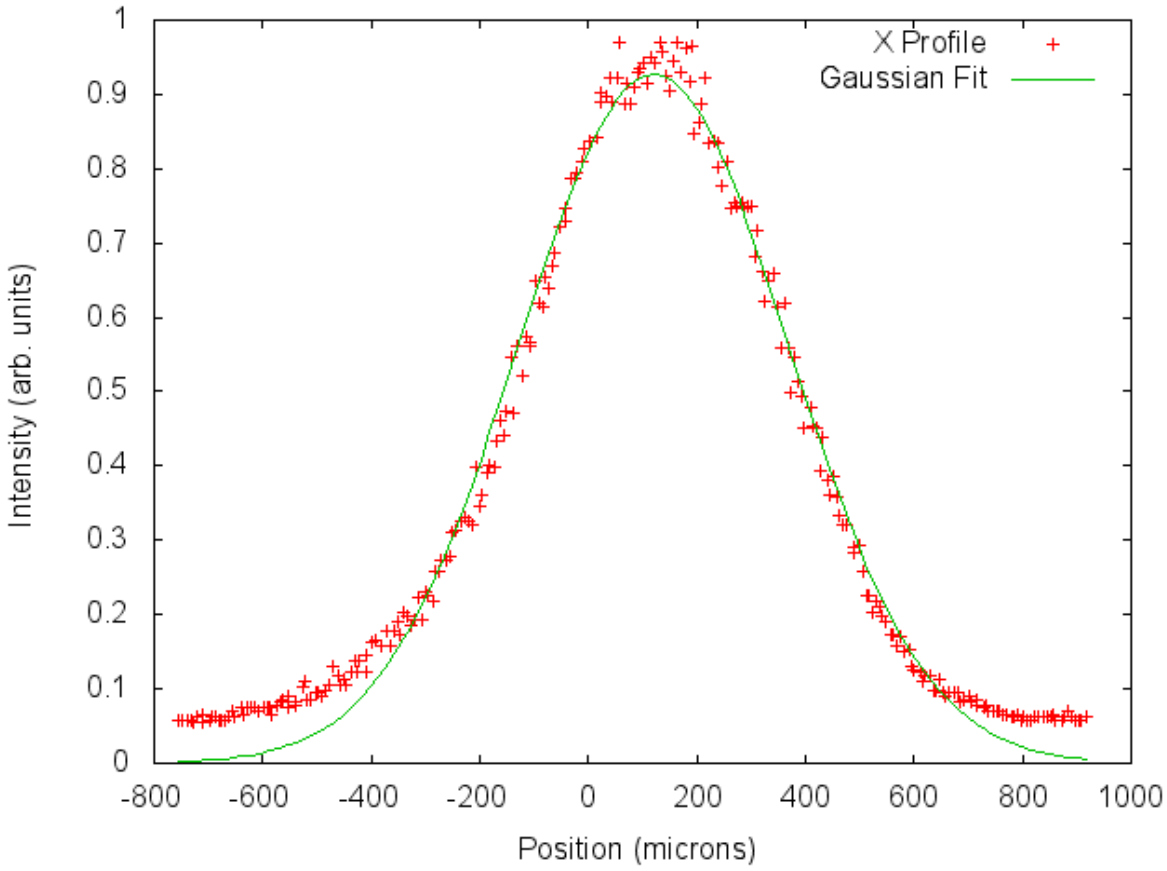

Figure 4: The X profile scan taken on the SSRL Diagnostic Beamline showing the Gaussian fit.

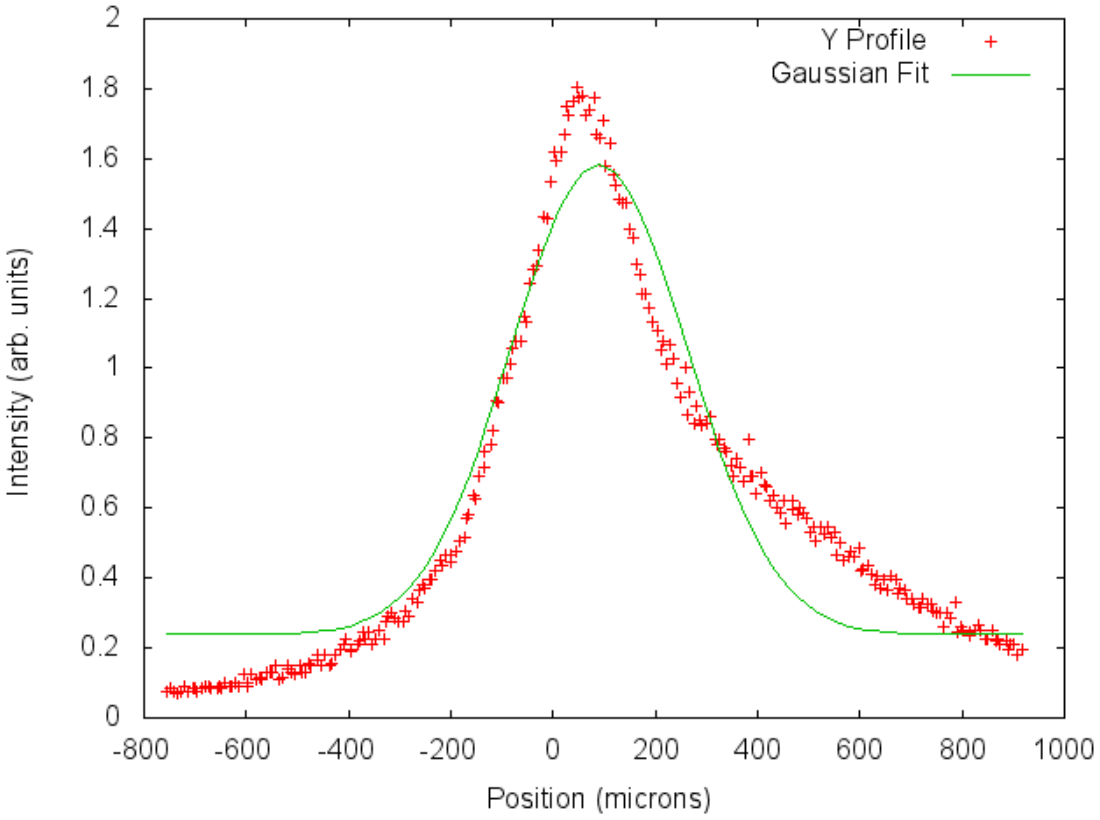

Figure 5: The Y profile scan taken on the SSRL Diagnostic Beamline showing the Gaussian fit.

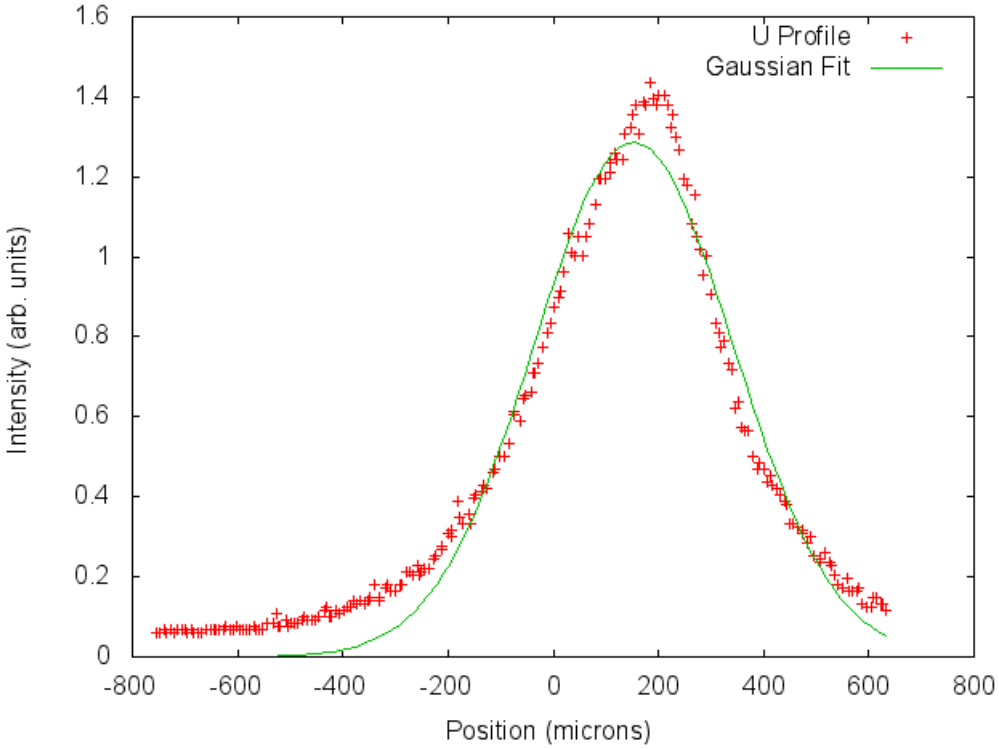

Figure 6: The U profile scan taken on the SSRL Diagnostic Beamline showing the Gaussian fit.

Table 1: Profile σ values

Profile	Expected σ	Measured σ (automated)	Measured σ (measured ω)
X	60.94	321.98	234.76
Y	20.31	282.22	169.98
U	Unknown	262.86	189.13

APPENDIX B: Code

```
# Read data from scope containing 5 signals
# Converts time to distance using angle resolver
#
# Data is read using numpy.genfromtxt which requires a file open in
# 'binary read' mode.  The cosine signal is then scanned for maxima
# to build a carrier frequency profile.  This is used to calculate
# an average angular frequency for use in time to position conversion
# Next, the time and intensity data are read and maxima are found.
# These points correspond to the 'mondo' bunches that were observed
# during data collection.  The time scale is then converted to
# position using the previously calculated angular frequency.  This
# data is all saved to a text file for further use.
#
# Mitch Miller
# 7/22/11

import sys
import numpy as np
import math as m

##pathList = ["E:\TestProfile.csv"]
##pathList =
[r"E:\XProfile_600V_1ND_721.csv",r"E:\YProfile_600V_1ND_721.csv",r"E:\UProfile_600V_1ND_721.csv"]
pathList = ["/media/2423-A37F/XProfile_600V_1ND_721.csv","/media/2423-
A37F/YProfile_600V_1ND_721.csv","/media/2423-A37F/UProfile_600V_1ND_721.csv"]

for path in pathList:

    ## Define Variables
    numberOfPeaks = 256  ## 256 for full scan
    inFile = open(path,'rb')
    outPath = path.replace('.csv','.peaks.csv')
    anglePath = path.replace('.csv','.anglepeaks.csv')
    outFile = open(outPath,'wb')
    angleOutFile = open(anglePath,'wb')
    max = 0
    sinMax = 0
    cosMax = 0
    counter = 1
    peakCounter = 0
    omegaCounter = 0
    omegaPeaks = 20  ## 20 for full scan
    gearRatio = (64. / 3375.)
    maxCounter = 0
    pointCounter = 0
    maxArray = np.array([[0,0,0,0,0,0]])
    angleArray = np.array([[0,0,0]])
    radius = 80000.  ## In microns
    omega = 0
    avgOmega = 0
```

```
    ## Read data from file
    sys.stdout.write('Begin Reading Data \n')
    sys.stdout.write('This will take up to 30 minutes \n')
    ##data = np.genfromtxt(inFile, dtype=None, delimiter=",")#, skip_header=16,
names=['time','cos','sin','res','inten'])  ##WINDOWS
    data = np.genfromtxt(inFile, dtype=None, delimiter=",", skiprows=16, names=['time','cos','sin','res','inten'])  ##
LINUX

    numberOfPoints = np.size(data,axis=0)




    ## Find peaks for Cos and Sin
    ## NOTE: Only cos peaks are measured directly,
    ## sin peaks may have some error.  Use cos for
    ## calculations if possible.
    sys.stdout.write('Begin Angle Peak Search\n')
    pointsPerSection = numberOfPoints/omegaPeaks
    while omegaCounter < omegaPeaks:

        counter = omegaCounter * pointsPerSection
        cosMax = 0
        sinMax = 0

        while counter < ((omegaCounter + 1) * pointsPerSection):

            if data[counter][1] > cosMax:

                cosMax = data[counter][1]
                sinMax = data[counter][2]
                time = data[counter][0]

            counter  = counter + 1

        dummyArray = np.array([[time,(cosMax),sinMax]])
        angleArray = np.concatenate((angleArray,dummyArray))
        omegaCounter = omegaCounter + 1



    ## Calculate average omega for time to distance conversion
    sys.stdout.write('Begin Omega Calculation\n')
    counter = 1
    angleArray = np.delete(angleArray, [0], axis = 0)
    # Find the maximum of the array and normalize to ensure [-1 < data < 1]
    absoluteValueArray = np.absolute(angleArray)
    arrayMax = absoluteValueArray.max(axis=0)
    if arrayMax[1] < 1:
```

```
        arrayMax[1] = 1
    if arrayMax[2] < 1:
        arrayMax[2] = 1

    while counter < omegaPeaks:

        deltaT = angleArray[counter][0] - angleArray[counter-1][0]
        deltaTheta = m.tan((angleArray[counter][2] / arrayMax[2])/(angleArray[counter][1] / arrayMax[1])) -
((angleArray[counter-1][2] / arrayMax[2])/(angleArray[counter-1][1] / arrayMax[1]))          ##deltaTheta =
m.acos(angleArray[counter][1]) - m.acos(angleArray[counter-1][1])
        if deltaTheta != 0 and m.sqrt(deltaTheta**2) < 0.25:
            omega = deltaTheta / deltaT
            avgOmega = avgOmega + omega
            pointCounter = pointCounter + 1
        counter = counter + 1

    avgOmega = avgOmega / pointCounter
    avgOmega = avgOmega * gearRatio
    avgOmega = avgOmega * avgOmega
    avgOmega = m.sqrt(avgOmega)
    sys.stdout.write('Avg Omega = ')
    sys.stdout.write(str(avgOmega))
    sys.stdout.write('\n')


    ## Save angle peaks
    np.savetxt(angleOutFile, angleArray, delimiter = ',')



    ## Scan each section for largest value and save inensity
    ## and position.
    sys.stdout.write('Begin Main Function\n')
    counter = 0
    pointsPerSection = numberOfPoints/numberOfPeaks
    while peakCounter < numberOfPeaks:

        counter = peakCounter * pointsPerSection
        max = 0

        while counter < ((peakCounter + 1) * pointsPerSection):

            ## Record point if it is larger than the previous
            if data[counter][4] > max:

                max = data[counter][4]
                time = data[counter][0]
                maxCounter = counter

            counter = counter + 1

        ## Convert time to position and record all relevant data in maxArray
        ## [Time, Position, Cos, Sin, Drive, Intensity]
```

```
            position = radius * avgOmega * time
            dummyArray =
np.array([[time,position,data[maxCounter][1],data[maxCounter][2],data[maxCounter][3],max]])
            maxArray = np.concatenate((maxArray,dummyArray))

            ## Print current peak number and omega for debug
            ##sys.stdout.write('Counter = ')
            ##sys.stdout.write(str(peakCounter))
            ##sys.stdout.write('\n')

            peakCounter = peakCounter + 1



    ## Delete first row [0,0,0,0,0,0] from array
    maxArray = np.delete(maxArray, [0], axis = 0)



    ## Save array of maximums to a text file
    ##output = ''.join('Average Omega = ',str(avgOmega),'\n')
    ##outFile.write(output)
    sys.stdout.write('Saving Data\n')
    np.savetxt(outFile, maxArray, delimiter = ',')



    sys.stdout.write('File Complete\n')
    inFile.close()
    outFile.close()
```