Raspberry-pi beam profiler

Manasa V S, Pondicherry University, India

September 02, 2022

Abstract

The Raspberry-pi beam profiler is a high-resolution automated laser-beam profiler with axial translation. It can be used to determine the effective focal length and focal spot size of the beam. As the name suggests it is based on a Raspberry Pi, Pi Noir camera, stepper motor and commercial translation stage. The software associated with the device was designed to acquire the data, analyse it by curve fitting and give out the required results. The objective of my project was to find solutions to some of the problems associated with the software and improve its performance.

Contents

1. Introduction .... Error! Bookmark not defined.
2. Theory 3
   2.1. Hardware 3
   2.2. Software 4
   2.3. User interface 4
   5
3. RESULTS 6
   3.1. Converting files from ‘.pkl’ to ‘.png’ format. 6
   3.2. 2D Gaussian fit 6
   3.2.1. Fitting the spot size data 8
4. Conclusion and Future plans 9
5. References 9

1. INTRODUCTION

It is always important to precisely know the beam profile of the laser that is being used in any experiment. Apart from the intensity profile, knowing the position of the focus and the spot size is necessary for several experiments involving fibres. Traditional methods of spot size measurement like knife-edge techniques are time consuming. Even, using a camera manually to move from one point to another and capturing the beam at different points could also bring about a lot of errors.

An automated device with well-designed software could possibly be an extremely helpful tool to find out these parameters of the beam. Although several commercial translating beam profilers are available these days, these are usually expensive and bulky.

In this project, I have worked on improving the data acquisition and fitting of the program associated with the beam profiling device that works with the help of Raspberry pi. Raspberry pi is a low-cost, single-board computer, and hence the whole setup could be constructed by anyone easily, at an affordable rate.

2. THEORY

2.1. Hardware

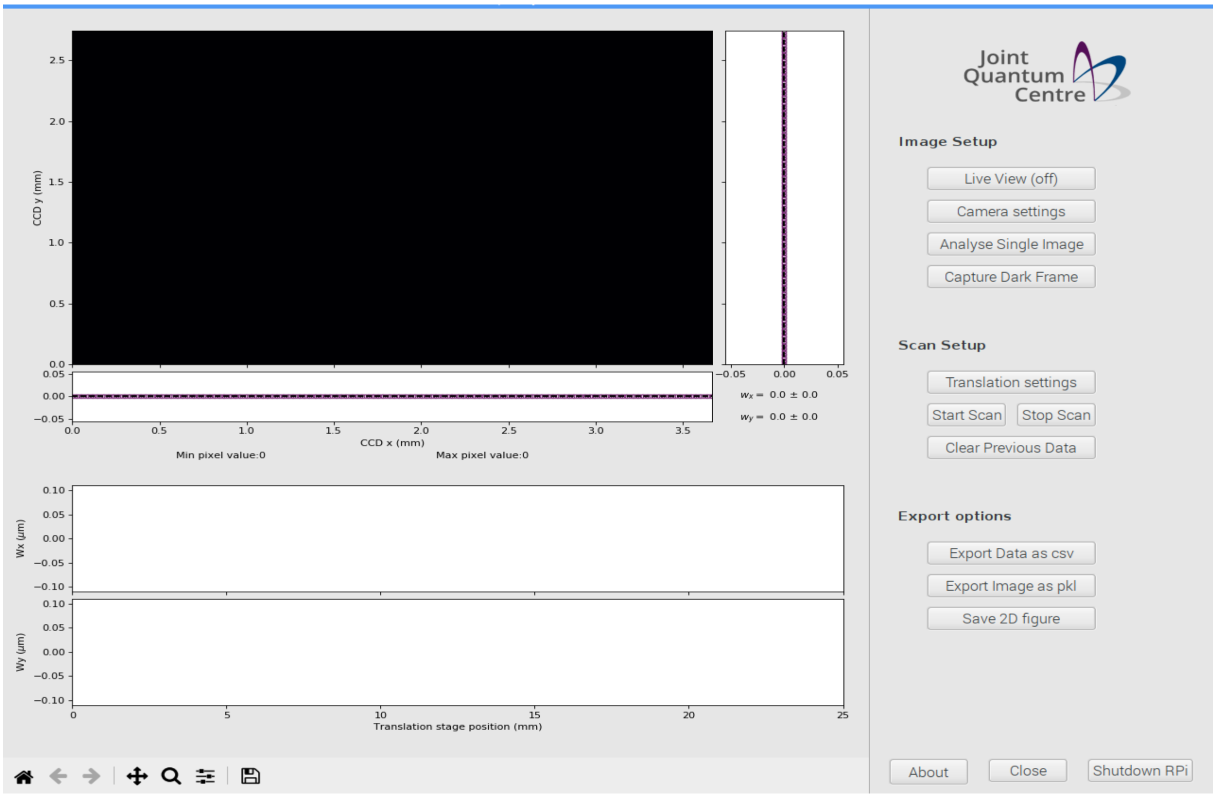
- Raspberry pi
- PiNOIR camera:
    - Sensor area: 3.67 x 2.54 mm
    - Total number of pixels: 3280 x 2464
    - Pixel pitch: 1.12 µm
- Translation stage (Thorlabs PT1/M, with a custom 0.5 mm pitch screw)
    - single step resolution of 0.9 degrees
    - 0.5 mm pitch screw
    - spatial step size of 500 x 0.9 / 360 = 1.25 µm
- Monitor with HDMI input
- HDMI to HDMI / DVI cable (for connecting a monitor)
- Keyboard and Mouse (both USB)
- USB memory stick for retrieving data
- A micro-SD card (at least 8 GB)

- power supply

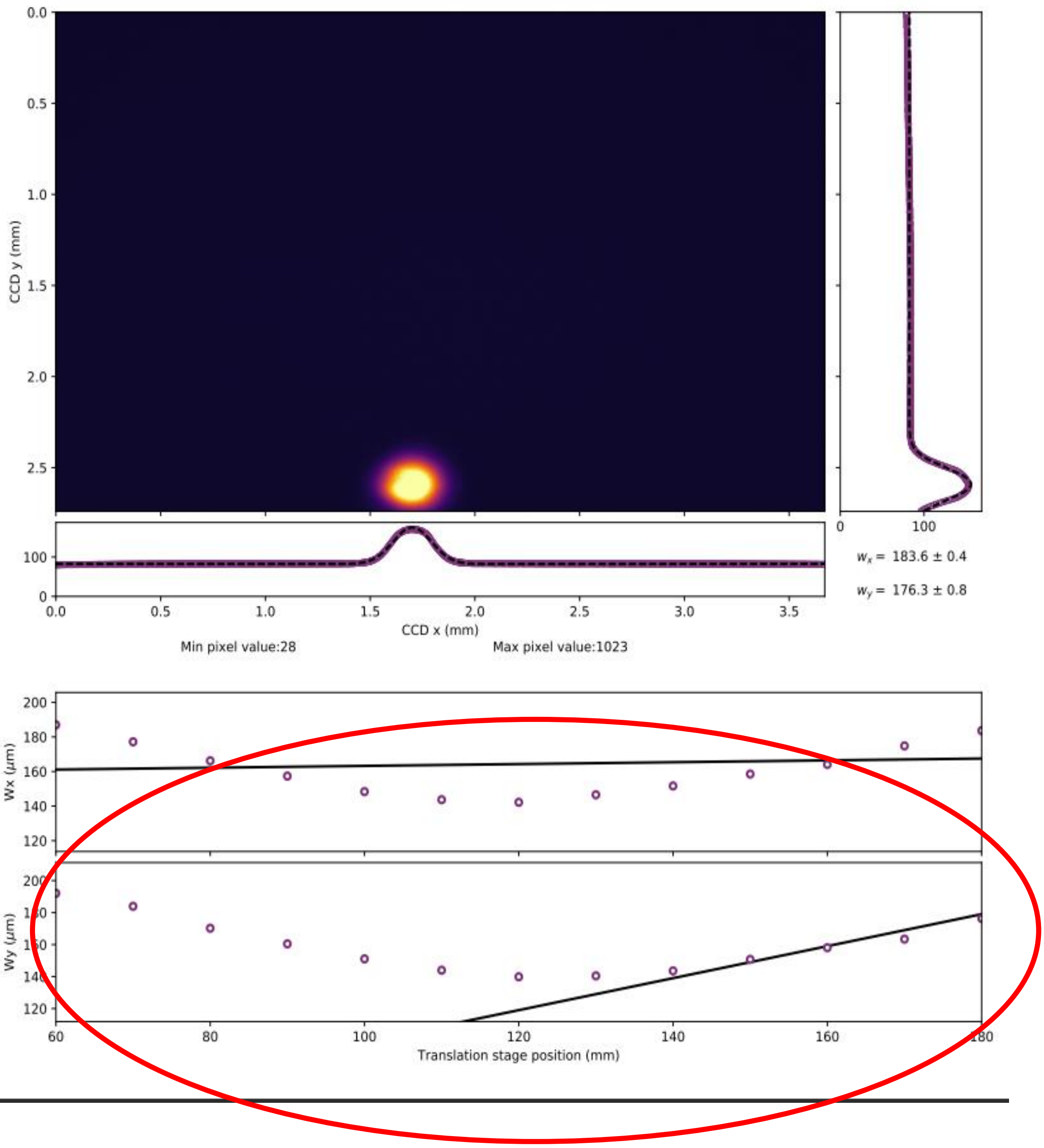
2.2. Software

- Operating system: Raspbian
- Python modules:
    - python-matplotlib
    - python-scipy
    - python-numpy
    - python-picamera
    - python-rpi.gpio
- The program is currently written in Python2.

2.3. User interface



- Camera settings: Exposure time, ROI
- Translation settings: Start position, stop position, step size

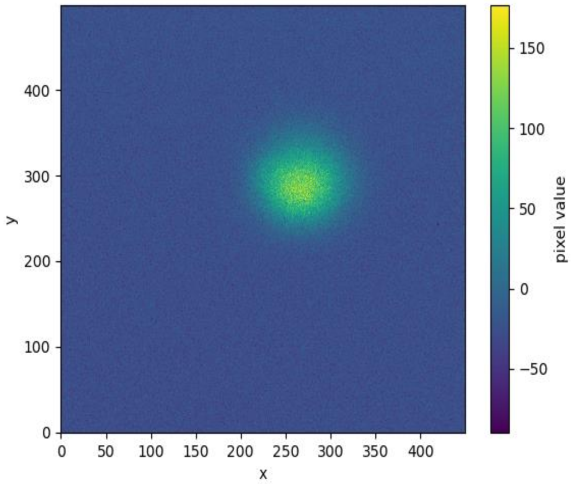


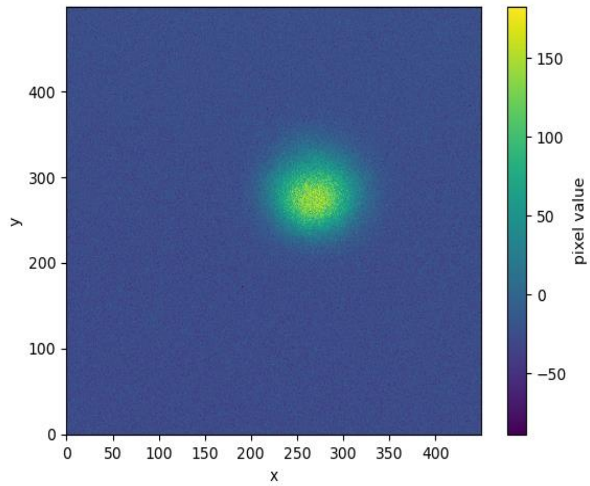
3. RESULTS

3.1. Converting files from '.pkl' to '.png' format.

The images acquired were saved as .pkl files. These files were of size ~30MB. This was a problem since the memory gets fully occupied just by 2 or 3 sets experiments. The data was converted to .png files.

Examples:





3.2. 2D Gaussian fit

- $I(r) = I_O e^{-\left(\frac{2r^2}{(w(z))^2}\right)} = \frac{2P}{\pi(w(z))^2} e^{-\left(\frac{2(x^2+y^2)}{(w(z))^2}\right)}$

where I_0 : peak irradiance at the center of the beam

r : radial distance away from the axis

w(z) : radius of the laser beam where the irradiance is $1/e^2$ (13.5%) of I_0

z : distance propagated

P : total power of the beam

- $w(z) = w_0\sqrt{1 + \left(\frac{\lambda z}{\pi w_0^2}\right)^2}$ where, w_0 : the waist radius

- $w(z) = w_0\sqrt{1 + \left(\frac{z}{z_R}\right)^2}$ where, z_R : Rayleigh range and $z_R = \frac{\pi w_0^2}{\lambda}$

- deviations from ideal : "M-squared" parameter (M^2)

$$z_R = \frac{\pi w_0^2}{M^2 \lambda}$$

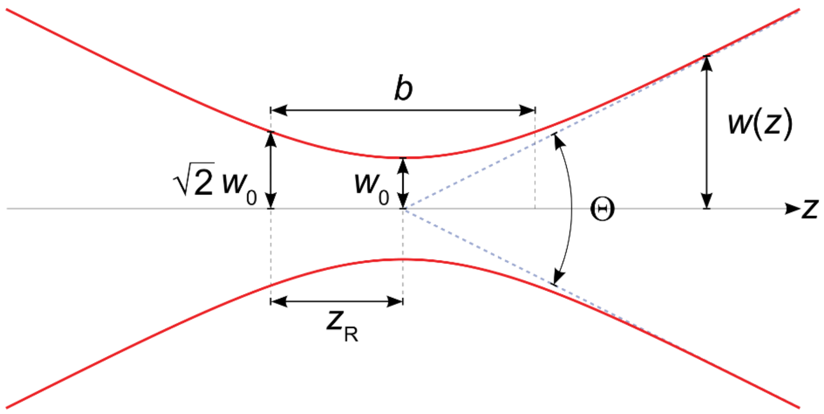


FIGURE: By Gaussianbeam.png: en:User:DrBob - Gaussianbeam.png, CC BY-SA 3.0, https://commons.wikimedia.org/w/index.php?curid=6002103

- Module used: lmfit

- Using built-in fitting models in the *models* module and *guess()* method to give a reasonable set of initial values considering the data given.

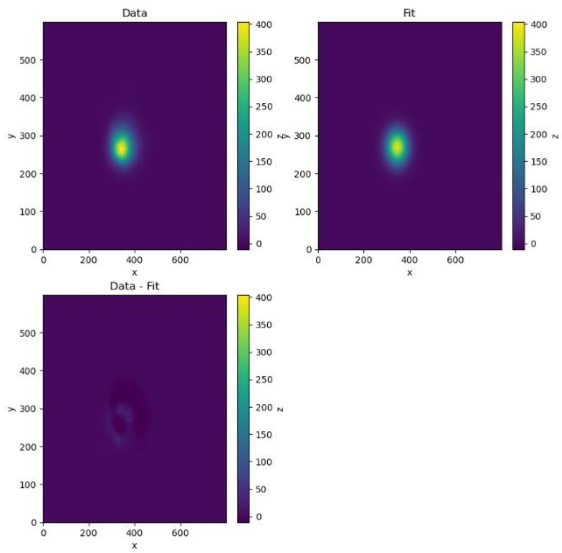
- $f(x, y; A, \mu.\sigma) = \frac{A}{2\pi\sigma_X\sigma_Y}\exp\left(-\left(\frac{x^2}{2\sigma_x^2} + \frac{y^2}{2\sigma_y^2}\right)\right)$, where

  A: amplitude

  μ: center

  σ: standard deviation

- On comparison, we see $w^2 = 4\sigma^2$ and $w = 2\sigma$



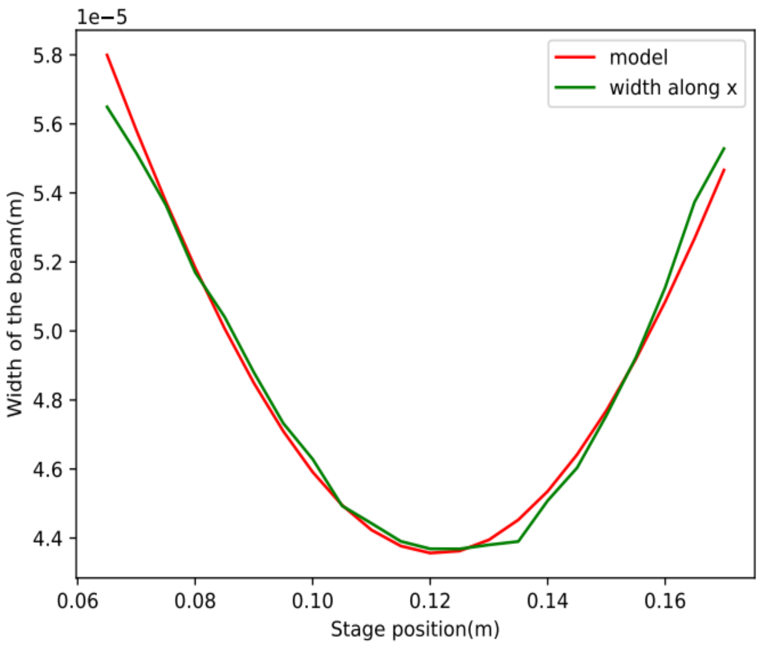
- Width of the beam along x and y with the error at each position (z) was obtained from the fit.

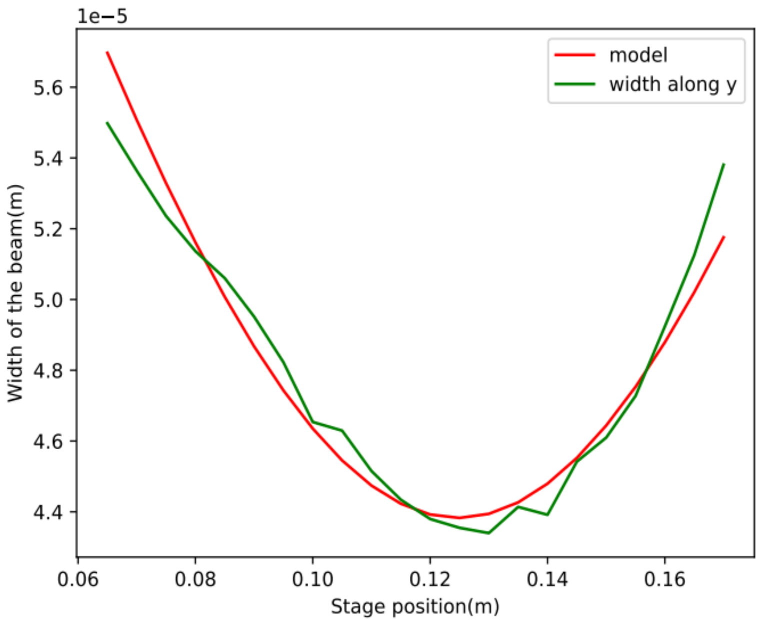
3.2.1. Fitting the spot size data

$$w = 2\sigma$$

$$w(z) = w_0 \sqrt{1 + \left(\frac{z - f}{z_R}\right)^2}$$

Model was built and initial value of the parameters (w_0, f) were assigned.





4. Conclusion and Future plans

The programme was made to successfully convert from .pkl to .png form (without significant change in the reading), do the 2D fitting and find the minimum value of width of the beam (spot size) and the focal position.

The following are my future plans:

- Converting the existing program from python2 to python3 and updating the necessary modifications in the software.
- Include M-square to the model and determine its value from the fit.
- Improve GUI.

5. References

- Automated translating beam profiler for in-situ laser beam spot-size and focal position measurements by James Keaveney
- https://github.com/koopaduo2/Beam-GUI/blob/main/beam_gui_intro_manual.pdf
- https://www.edmundoptics.eu/knowledge-center/application-notes/lasers/gaussian-beam-propagation/