

Analyzing the Radiation Emission Pattern of an X-Ray Tube's Focal Spot

Grant Budge

Carleton University

Phys 4909 A

April 2025

Contents

Abstract	3
1.0 Introduction to X-ray Focal Spots and Nominal Values	4
1.1 Background Information	4
1.2 Objective	5
2.0 Theoretical Framework.....	7
2.1 Pinhole Imaging.....	7
2.2 Blur and Detector Resolution.....	8
2.2.1 Pinhole Blur.....	8
2.2.2 Detector Resolution.....	9
2.2.3 Detector Blur.....	11
2.2.4 Total Blur.....	11
2.3 Flat Field Correction	12
2.4 Row Sequential Readout and Normalization.....	13
3.0 Methods and Materials.....	16
3.1 Experimental Set up.....	16
3.2 Capturing a Radiographic Pinhole Image.....	18
3.3 Measuring Nominal Value.....	19
4.0 Analysis	21
4.1 Matrix Analysis	21
4.1 Analyzing the Effective Focal Spot Data.....	24
4.3 Nominal Value Analysis.....	33
5.0 Results and Discussion.....	39
5.1 Results	39
5.2 Discussion	42
5.3 Acknowledgements.....	43
6.2 References.....	44
Appendix A.....	45
A.1 Nominal_Value_Finder.m.....	45
A.1.i length_average.m	53
A.1.ii width_average.m.....	53
A.2 GB_FFC.m.....	54
A.2.iGB_AVG_images	58

A.2.ii GB_calculate_weighted_average_image	61
A.3 GB_image_rotatation.m.....	63
A.3.i GB_image_window_and_level.m.....	68
A.3.ii GB_crop_image	71
Appendix B.....	72
B.1 GB_minimize_noise.m.....	72

Abstract

The focal spot of an x-ray tube is the region of an x-ray machine at which the x rays are generated. The International Electrotechnical Commission (*IEC*) assigns numerical nominal focal spot values to the size of x-ray tube focal spots to ensure consistent measurement across different machines and manufacturers.

Using pinhole imaging, the digital mapping of the focal spots by the Hamamatsu S11684-12 CMOS detector was evaluated to enhance the efficiency of measuring the nominal focal spot value in accordance with the IEC Standard 60336:2020. A key objective was to establish a clear reference for future testing by evaluating various combinations of imaging parameters such as enlargement, pinhole size, current, and exposure time - to determine which configurations reliably met IEC requirements. The criteria of the IEC include: a focal spot width at 15% of maximum intensity spanning at least 60 pixels, a signal-to-noise ratio (SNR) of at least 100, and a minimum of 200 signal levels between the background and maximum signal.

Using MATLAB, fixed pattern noise from the detector's row-sequential readout was addressed and reduced through a matrix normalization technique. Two normalization methods were compared: averaging then normalizing (ATN) and normalizing then averaging (NTA), with NTA offering slightly superior noise reduction for smaller images.

Additionally, a method for generating two line spread functions was developed to measure focal spot size. By averaging the focal spot width wise and height wise, the two line spread functions allowed for the measurement of focal spot width and height at 15% maximum intensity, as per IEC requirements.

The measured nominal value for the large focal spot was $f_W = 1.5$ and $f_H = 1.3$.

The measured nominal value for the small focal spot was $f_W = 0.7$ and $f_H = 0.9$.

These values were inconsistent with the manufacturer's serial plate which shows $f_{(large)} = 1.2$ and $f_{(small)} = 0.6$.

This work provides a practical framework to guide future evaluations and ensure consistent, IEC compliant measurements.

1.0 Introduction to X-ray Focal Spots and Nominal Values

1.1 Background Information

X-ray machines generate x-rays by accelerating electrons towards a metal anode using a high-voltage electric field, known as the accelerating potential or kilovoltage (kV).¹ These electrons originate at the cathode, where a heated tungsten filament, powered by an electric current, releases electrons through thermionic emission. As the electrons accelerate, they form a moving beam of charge, known as the tube current, typically measured in milliamperes (mA). When the electron beam strikes the focal spot on the anode (usually an alloy of roughly 90% tungsten and 10% rhenium), the electrons undergo a sudden deceleration. This generates x-rays primarily through Bremsstrahlung (braking radiation), which occurs when the electrons are decelerated and deflected by the strong electric fields of the atomic nuclei in the tungsten (or other target material), converting their kinetic energy into electromagnetic radiation.² The efficiency of Bremsstrahlung is relatively low for the electron energies in an x-ray tube—less than 1% of the electron's kinetic energy is converted into x-rays, while the remaining energy is converted into heat. Tungsten is the preferred material due to its high atomic number (74), which increases the likelihood of Bremsstrahlung. Also, its ability to withstand extreme temperatures, with a melting point of 3,422°C, make it ideal for handling the heat generated during x-ray production.

Previous studies at Carleton University, under the guidance of Dr. Paul Johns, have contributed to the field of focal spot imaging. Eva Anderson evaluated the Hamamatsu S11684-12 CMOS detector, focusing on its practicality for imaging x-ray focal spots. Her work resulted in a focal spot size measurement that met IEC tolerance limits using the Hamamatsu detector.³⁴ André Ramos Moreno developed a design for aligning detector, pinhole, and the x-ray focal spot, performing mathematical and numerical analyses to improve measurement accuracy. His design is a pinhole and detector stand, which will be useful in capturing higher-precision images when it is finished construction.⁵ Building on the prior work done, this work aims to enhance the efficiency of producing focal spot measurements that meet the IEC's standards for medical imaging.

Different focal spot sizes are necessary because they cater to various imaging applications. Different sizes allow for a balance between factors such as image resolution, heat dissipation, and exposure time. Smaller focal spots provide higher resolution but may limit the tube's capacity to handle high exposures. Larger focal spots allow for higher power outputs but reduce image sharpness.

To ensure that an x-ray tube's source is suitable for its intended application, its focal spot must be accurately characterized in accordance with industry standards. The International Electrotechnical Commission (IEC) mandates that each focal spot be assigned a nominal value for consistency across manufacturers and equipment type.⁶

The effective focal spot size differs from its actual size, with the anode angle (ϕ) being crucial to this variation. Two anodes with the same actual focal spot size may have different effective focal spot sizes with a different anode angle ϕ (see **Figure 1**). A larger anode angle with a smaller actual focal spot may appear similar in size to a smaller anode angle with a larger actual focal spot. The effective focal spot size is what determines the nominal focal spot value.⁷

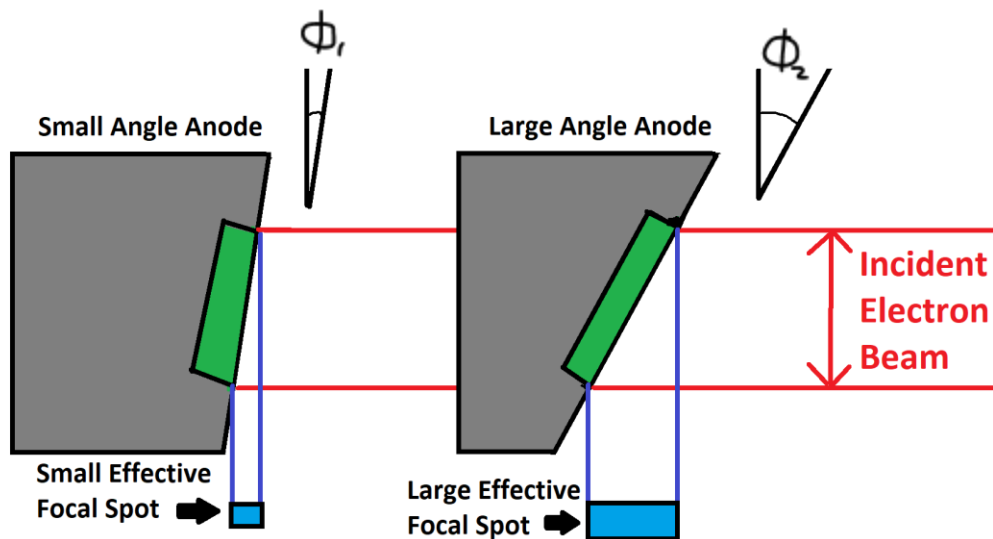


Figure 1 – The size of the effective focal spot depends on the angle of the anode. Although the small angle (ϕ_1) anode and large angle (ϕ_2) anode are struck by the same width electron beam, the effective focal spot size for the small angle anode (left) is smaller than for the anode with the larger angle (right).

1.2 Objective

The goal of this research is to develop a method for measuring the nominal value of a focal spot using the Hamamatsu digital detector. This method should consider the detector's row-sequential readout, detector blur, pinhole blur, enlargement, and the variables involved in producing radiographic images such as tube current, and exposure time. Taking all of this into account will provide a solid foundation for future research on focal spot imaging tailored to this detector.⁸

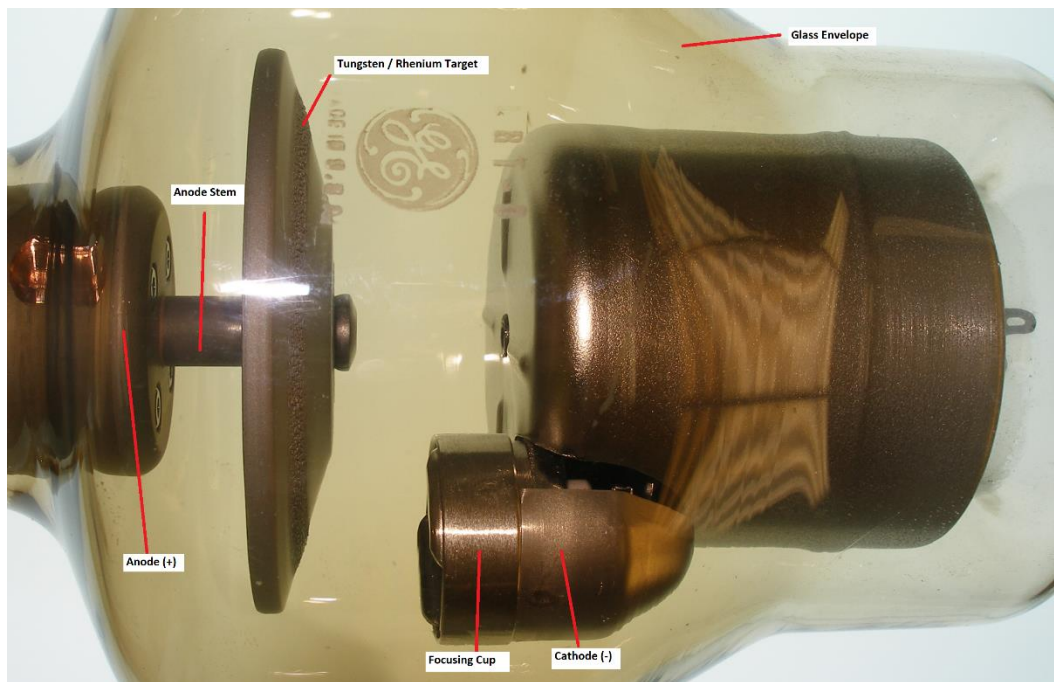


Figure 2 - Labelled x-ray tube with a rotating anode.

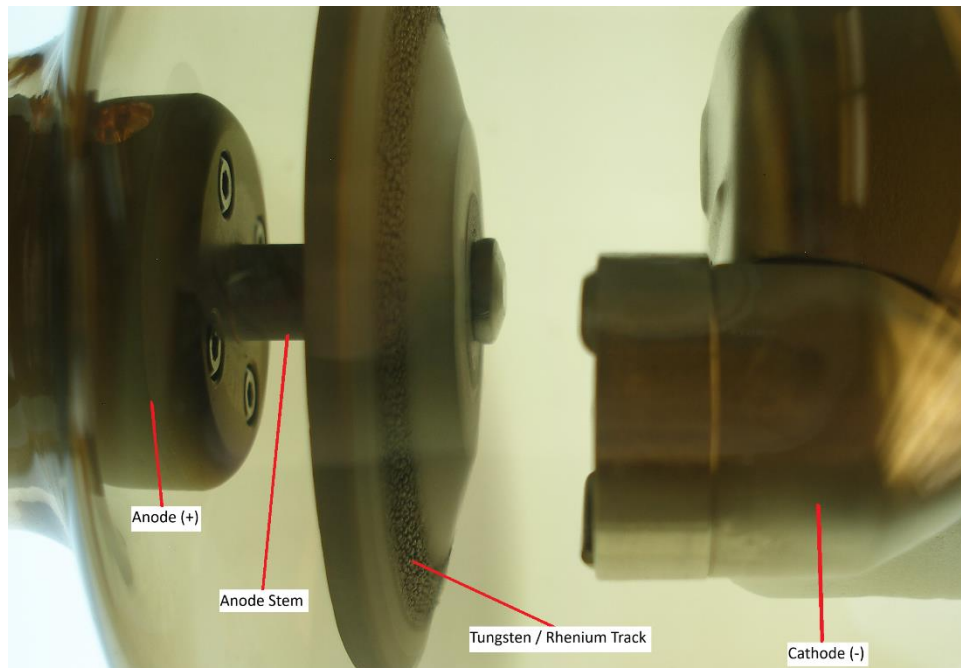


Figure 3 - Rotating anode with a tungsten / rhenium target disk. Most of the disk is Molybdenum, the circular track is the tungsten / rhenium alloy. X rays are generated when electrons from the cathode bombard the anode.

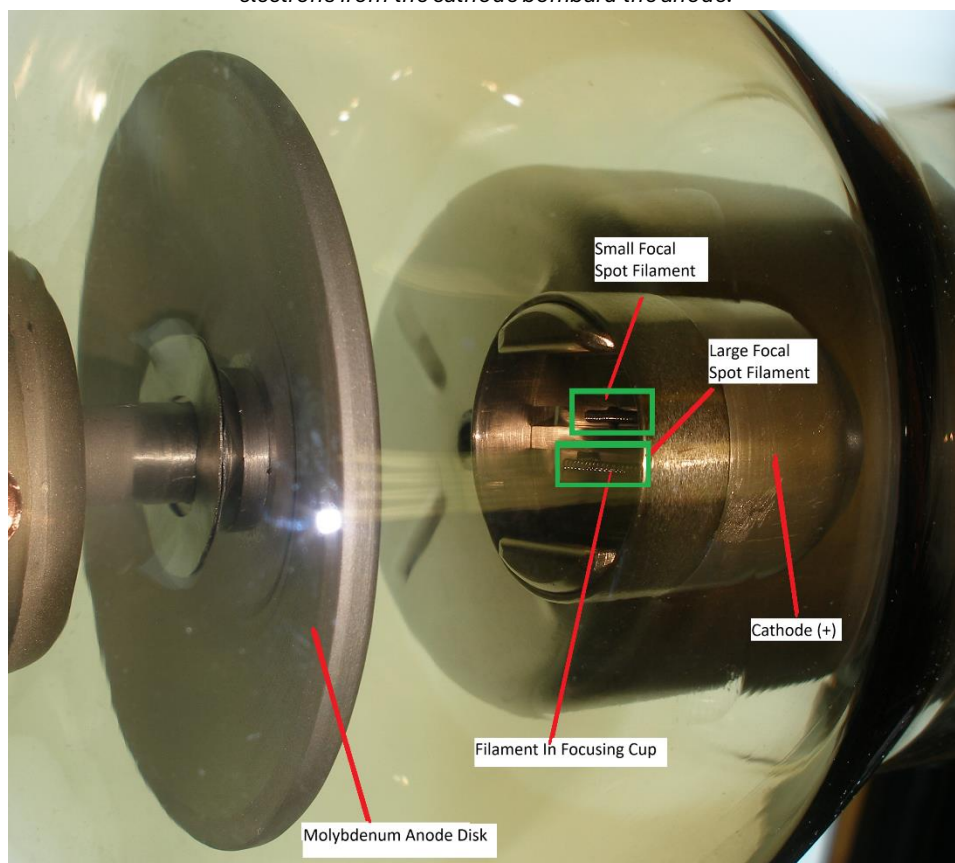


Figure 4 - X-ray tube showing the filament which is heated to produce electrons. The filament is located within the focusing cup, which directs the electron beam towards the focal spot track on the anode disk.

2.0 Theoretical Framework

2.1 Pinhole Imaging

A pinhole projects an inverted image from a light source onto a screen. The size of this image depends on the relative positions of the pinhole, the light source, and the screen. As the distance between the pinhole and the screen increases, the image becomes larger but also dimmer. This relationship is illustrated in **Figure 5**, where an increased pinhole-to-screen distance results in a larger but fainter image. The faintness arises because the same amount of light (represented by the coloured lines) is spread over a larger area. This behaviour describes photon fluence (Φ).

$$\Phi = \frac{\# \text{ of photons}}{\text{area}} \quad [1]$$

Since the number of photons passing through the pinhole remains constant if the source-to-pinhole distance does not change, increasing the screen distance causes the image to cover a larger area, reducing photon fluence and thus making the image dimmer.

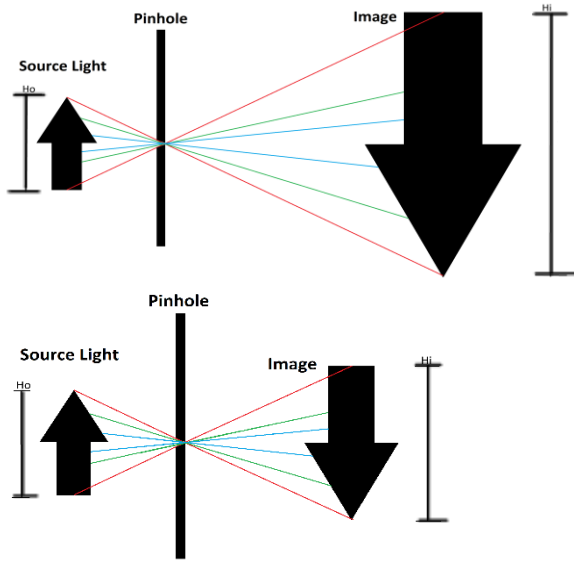


Figure 5 - Comparing enlargement and fluence for changing pinhole-to-detector distances.

The distance between the light source and the pinhole also affects the image. The intensity of light from a point source follows the inverse-square law, meaning it decreases as the inverse square of the distance from the source:

$$\phi = \frac{P}{4\pi r^2} \quad [2]$$

where ϕ is the fluence rate of the light, P is the power of the light source, and r is the distance from the source. As the source-to-pinhole distance decreases, more photons pass through the pinhole, increasing image brightness. However, for fixed detector locations, this also enlarges the image, causing the photons to be distributed over a greater area, reducing their concentration and making the image dimmer.

Conversely, increasing the source-to-pinhole distance reduces image enlargement if the source to detector distance is kept constant, but also decreases brightness due to the inverse-square law. This trade-off between image size and brightness is a fundamental characteristic of pinhole imaging. For math and simplicity, the described distances will be denoted as variables:

d_1 = source to pinhole distance

d_2 = pinhole to detector distance

d_3 = source to detector distance

Thus, $d_3 = d_1 + d_2$

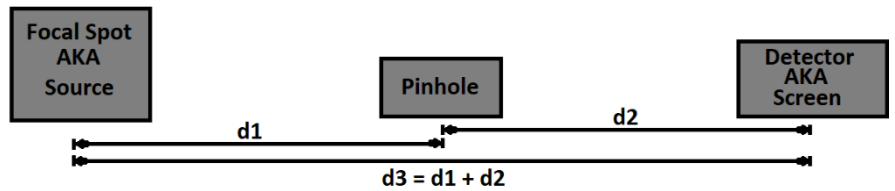


Figure 6 - Labelled distances.

As previously mentioned, changing the relative distances within the set up affects the size of the projected image. This enlargement is calculated using the following relationship:

$$E = \frac{H_i}{H_o}, \quad [3]$$

where H_i is the size of the image, and H_o is the size of the object. The image size and object size can be calculated algebraically, using d_2 and d_3 as:

$$E = \frac{d_2}{d_3 - d_2}. \quad [4]$$

As $d_2 \rightarrow d_3$, the enlargement factor $E \rightarrow \infty$. In this case, the image becomes excessively magnified and blurred, while the reduced photon fluence makes the image too dim to be distinguishable. Conversely, as $d_2 \rightarrow 0$, the enlargement factor approaches zero ($E \rightarrow 0$), causing the image to shrink until it is just the pinhole aperture filled with light.

2.2 Blur and Detector Resolution

Pinhole imaging introduces two primary sources of blur: pinhole blur and detector blur. Each type of blur is influenced by the geometric relationships between the pinhole, the source, and the detector.

2.2.1 Pinhole Blur

The blur caused by the finite size of the pinhole is known as the pinhole's point spread function (PSF). Since a physical pinhole has a finite diameter p , light rays from a single point of the focal spot spread out, forming a small, blurred circle on the detector. The size of this blur in the detector plane, B_p' , depends on both the pinhole diameter p and the distance d_2 between the pinhole and the detector:

$$B_p' = \frac{pd_3}{d_3 - d_2} \quad [5]$$

To express the blur in the source plane, B_p , we divide by the enlargement fact $E = \frac{d_2}{d_3 - d_2}$, yielding:

$$B_p = \frac{B_p'}{E} = \frac{pd_3}{d_3 - d_2} \times \frac{d_3 - d_2}{d_2} = p \left(\frac{1}{E} + 1 \right), \quad [6]$$

where, as d_2 increases, it enlarges the image while also spreading out the pinhole blur over a larger area. This effectively reduces the blur in the source plane. Decreasing the pinhole size p also reduces blur but at the cost of lower light throughput, which would make the image dimmer.

2.2.2 Detector Resolution

Detector resolution refers to the smallest detail that can be distinguished by the detector and is typically measured in line pairs per millimeter (lp/mm), where one line pair consists of one black line and one white line. In digital detectors, resolution is influenced by pixel size, as smaller pixels allow for better spatial resolution. However, in detectors using a scintillator, light spread within the scintillator causes blurring before reaching the pixels, which can limit the resolution independently of pixel size. Higher lp/mm values correspond to better resolution, as more line pairs can be resolved within one millimeter. Enlarging the image does not improve the detector's spatial resolution, but it does improve the image resolution by spreading the image over a larger area, allowing finer details to become more distinguishable.



Figure 7 - Resolution test strip placed directly on the Hamamatsu detector. The test strip is placed vertically, but the line pairs are horizontal along the detector.

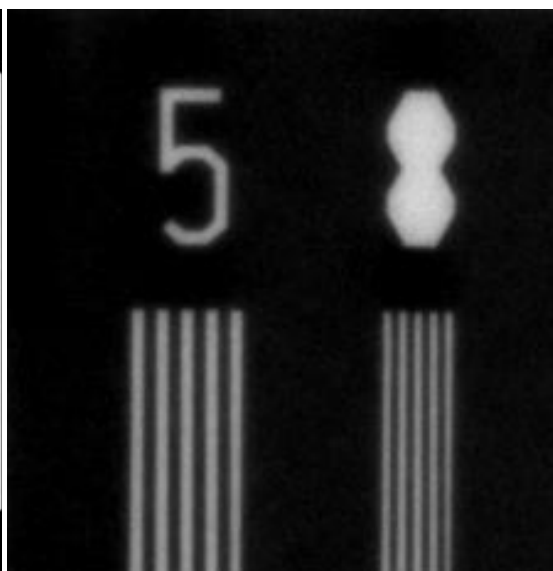


Figure 8 - Zoomed in resolution test strip showing distinct line pairs (5 and 8) per millimeter.

Figure 7 shows a resolution test strip placed directly on the Hamamatsu detector. The test strip consists of multiple labeled sections, each corresponding to a specific line pairs per millimeter (lp/mm) value. Each section contains exactly five line pairs (four black lines and five white spaces with a black background). **Figure 8** provides a magnified view of the 5 and 8 lp/mm sections. While each section contains only five line pairs, the lp/mm value represents how many of these pairs would fit within a millimeter if the pattern were extended.

Figure 9 shows three additional zoomed-in views of the test strip, each captured at a different angle relative to the detector's pixel pattern. The test strip, which was placed horizontally along the detector, meaning its line pairs are vertical along the detector, seems to show the best resolution; this is most likely due to the row-sequential readout of the detector.

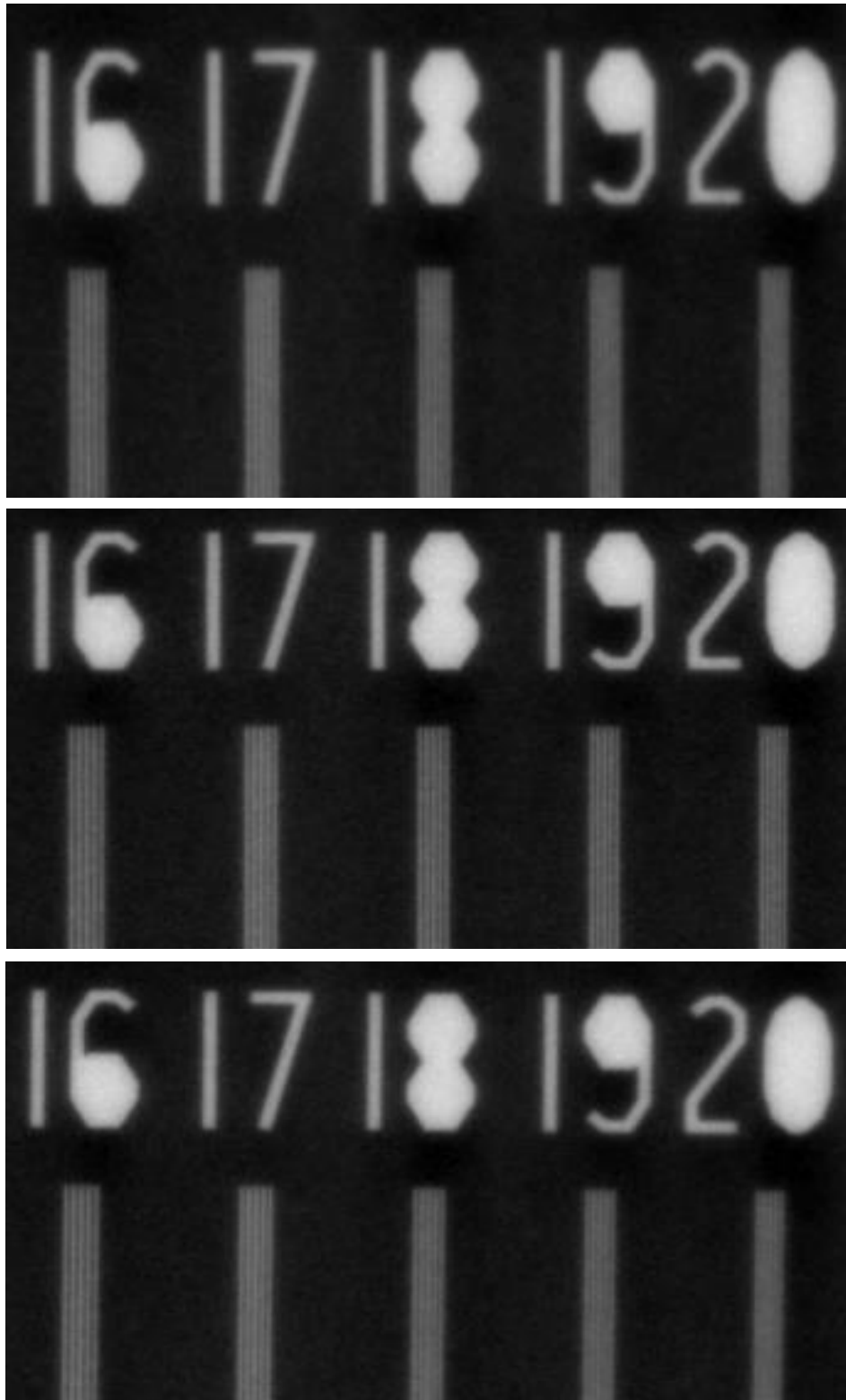


Figure 9 - Zoomed in resolution test strip images taken with the Hamamatsu S11684-12 detector. The images show the strip from 16 lp/mm to 20 lp/mm. The strip was placed diagonally (top), horizontally (middle), and vertically (bottom) along the detector.

2.2.3 Detector Blur

Detector blur arises from three main factors: pixel size, scintillator light spread, and noise. Each of these components influences the sharpness and clarity of the captured image.

The pixel size of the detector strongly influences the spatial resolution, but also contributes to blur by limiting the smallest detail that can be accurately captured. The Hamamatsu detector uses a thallium-doped cesium iodide ($CsI(Tl)$) scintillator.⁹ When x-rays hit the scintillator, they are converted into visible light, which is then detected by the underlying sensor. This conversion process is not localized because light spreads by a small amount within the scintillator, causing the image to blur.

The amount of light spread depends on the scintillator's thickness and material properties, with thicker scintillators producing more blur due to increased light diffusion.¹⁰ The $CsI(Tl)$ scintillator exhibits an afterglow of 0.3% at 100 ms,⁷ meaning a small residual emission of light persists briefly after exposure ends, which can further reduce image sharpness.

Noise in the detector system arises from several sources, including fixed pattern noise and readout noise. Fixed pattern noise stems from variations in pixel responsivity, meaning that identical radiation intensities produce varying output signals across the detector. For the Hamamatsu detector, this variation is given as a sensitivity fluctuation of $\pm 30\%$, meaning the pixel response can vary by up to 30% from the average value. Readout noise, on the other hand, results from fluctuations during signal transfer out of the chip and can introduce additional blur to the image.

The combination of these three effects blurs the image, reducing the ability to resolve fine details. While the intrinsic blur caused by the detector remains constant, magnification increases the number of pixels the image covers, allowing for finer details to become more distinguishable. Thus, detector blur (B_d) depends on the intrinsic blur of the detector, α , and the enlargement of the image:

$$B_d = \frac{\alpha}{E} \quad [7]$$

2.2.4 Total Blur

The total blur of the image is a combination of blur from the pinhole and blur from the detector. Since the two sources of blur are independent, their combined effect is calculated by adding them in quadrature. Adding the blurs directly would overestimate their total effect, so each contribution is squared to represent its area of influence, then summed, and the square root of this sum gives the total blur (B_{tot}) as a function of enlargement:

$$B_{tot}(E) = \sqrt{B_p^2 + B_d^2} = \sqrt{\left(\frac{p}{E} + p\right)^2 + \left(\frac{\alpha}{E}\right)^2} \quad [8]$$

Simplifying **Equation 8** gives us the total blur of the image in the focal spot plane:

$$B_{tot}(E) = \frac{\sqrt{\alpha^2 + p^2(E+1)^2}}{E} \quad [9]$$

We seek the enlargement, E , for which blur is minimized. Setting $\frac{\partial B_{tot}}{\partial E} = 0$ and solving for E :

$$\frac{\partial B_{tot}}{\partial E} = -\frac{p^2 E + p^2 + \alpha^2}{E^2 \sqrt{a^2 + p^2 (E+1)^2}} \quad [10]$$

$$-\frac{p^2 E + p^2 + \alpha^2}{E^2 \sqrt{a^2 + p^2 (E+1)^2}} = 0 \quad [11]$$

$$E = \frac{-p^2 - \alpha^2}{p^2} \quad [12]$$

With $E = \frac{d_2}{d_3 - d_2}$

$$d_2 = d_3 \left(1 + \frac{p^2}{\alpha^2}\right) \quad [13]$$

To minimize the total blur, apparently the pinhole must be behind the focal spot, which is impossible, thus from **Equation 9**, the most accurate focal spot image will have its blur minimized by taking E to be as large as possible.

2.3 Flat Field Correction

Flat Field Correction (*FFC*) is the standard calibration technique for digital imaging, which mitigates a detector's pixel-to-pixel variation and illumination irregularities.¹¹ It compensates for differences in pixel gains and dark currents, where dark current is a small random generation of electrons within the depletion region of the detector. Flat fielding an image will allow for a uniform signal to produce a uniform output. In an ideal FFC-corrected image, the background values will be uniform, and the imaged object will be clearly represented.

FFC works independently on each pixel of a detector. The process requires three raw radiographic inputs, where raw denotes that the images have not been processed. The three raw images are: a raw x-ray image (X), a dark image (D), and a gain image (G). In the case of pinhole imaging, X is a pinhole image. The total procedure follows:

$$FFC = \frac{(X-D)\langle G-D \rangle}{(G-D)} \quad [14]$$

The $(X - D)$ term removes the background, including contributions from detector noise and dark current, leaving just the relevant signal from the x-ray image. Then each pixel is multiplied by the average $\langle (G - D) \rangle$, which represents the average pixel sensitivity across the detector after accounting for the background. Multiplying by this average value ensures each pixel is properly scaled. Finally, dividing by the corrected gain $(G - D)$ ensures the individual pixel corrections are normalized and consistent across the image. As a result of this process, lower-level pixels are brightened, and higher-level pixels are darkened. This is shown in **Figure 10**.

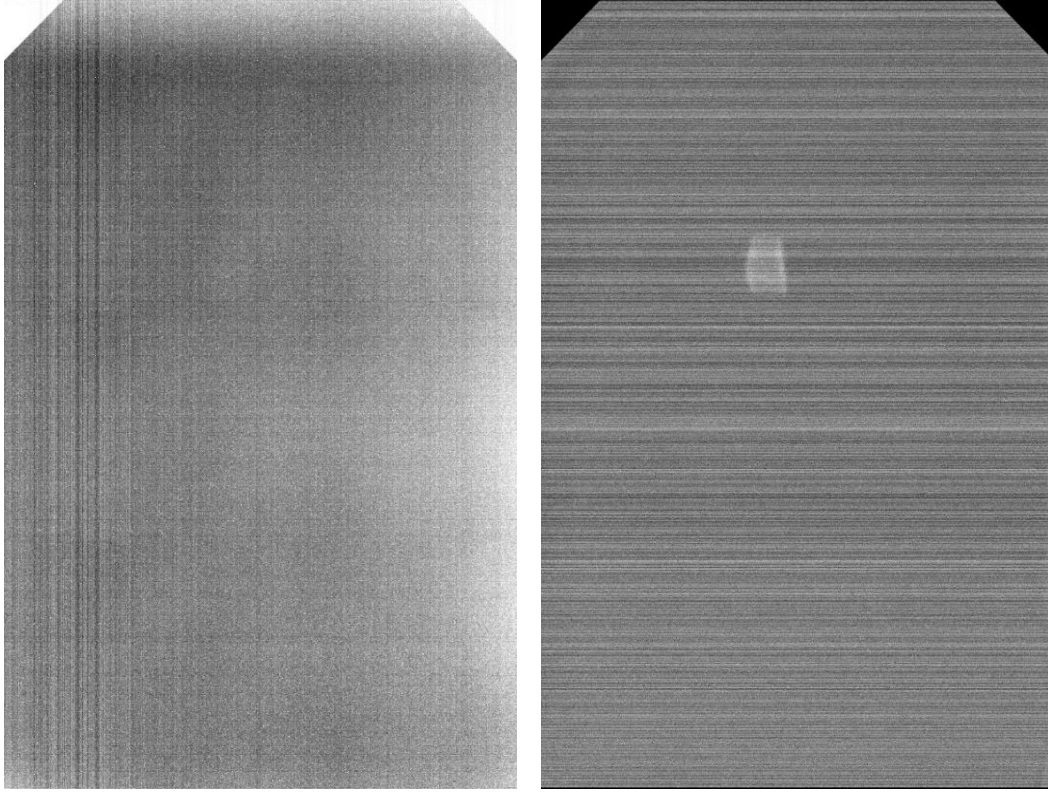


Figure 10 - Comparison of a raw pinhole x-ray image, and the same image after Flat Field Correction. The focal spot is almost completely obscured in the raw image.

2.4 Row Sequential Readout and Normalization

Detectors with row-sequential readout, such as complementary metal-oxide semiconductor (CMOS) sensors, often exhibit significant variations in readout timing between rows compared to the timing between columns. Row-sequential readout is a method where image data are extracted from the sensor line by line, progressing from one end of the sensor to the other. In sensors with 1000×1500 pixel array (like the Hamamatsu detector), this means that the data from the first pixel in the top left are read out 1000 pixels before the pixel directly below that. The time difference between rows is much larger than between columns, leading to visible variations in row intensity. This can result in images with noticeable differences, such as alternating dark and light rows.

Since the Hamamatsu sensor employs a row-sequential readout, this introduces the need for a row-specific normalization factor to correct for row-dependent variations in pixel values.

Some detectors (like the Hamamatsu detector) also contain shielded pixels, which are used in their built-in automatic correction systems. To ensure proper normalization from the raw image data (from manual mode), a mask is applied to eliminate the values of the shielded pixels from the manual calculation.⁹

There will be multiple acquisitions captured and processed to produce an averaged image. Denoting $_n A_{r,c}$ for the n^{th} matrix of R rows and C columns, each with pixel values $_n a_{r,c}$, a formula can be produced to apply a row normalization.

The average pixel value for row r is calculated as:

$${}_n(AC)_r = \frac{1}{C} \sum_{c=1}^C {}_nA_{r,c} , \quad [15]$$

where (AC) refers to the fact that the matrix A has been averaged over its columns. Thus, ${}_n(AC)_r$ is a column vector of size $R \times 1$. The average pixel value ${}_n\langle a \rangle$ of matrix n is:

$${}_n\langle a \rangle = \frac{1}{R} \sum_{r=1}^R {}_n(AC)_r. \quad [16]$$

Each row-specific normalization constant ${}_nd_r$ will be calculated using:

$${}_nd_r = \frac{{}_n\langle a \rangle}{{}_n(AC)_r}. \quad [17]$$

- If ${}_n(AC)_r > {}_n\langle a \rangle \rightarrow {}_nd_r < 1$
- If ${}_n(AC)_r < {}_n\langle a \rangle \rightarrow {}_nd_r > 1$
- If ${}_n(AC)_r = {}_n\langle a \rangle \rightarrow {}_nd_r = 1$

This can be expressed as:

$${}_nD = \begin{bmatrix} \frac{{}_n\langle a \rangle}{{}_n(AC)_1} \\ \frac{{}_n\langle a \rangle}{{}_n(AC)_2} \\ \vdots \\ \frac{{}_n\langle a \rangle}{{}_n(AC)_R} \end{bmatrix} = \begin{bmatrix} {}_nd_1 \\ {}_nd_2 \\ \vdots \\ {}_nd_R \end{bmatrix}, \quad [18]$$

where ${}_nD$ is an $R \times 1$ column vector with each normalization factor ${}_nd_r$ scaling the corresponding row r in the matrix. Using MATLAB's element-wise multiplication, the normalized matrix ${}_nA_{norm}$ is given by:

$${}_nA_{norm} = {}_nA_{r,c} .* {}_nD, \quad [19]$$

where ${}_nA_{r,c}$ is the original $R \times C$ matrix, and ${}_nD$ is the $R \times 1$ column vector. The element-wise multiplication performed in MATLAB can be expressed mathematically as the Hadamard product.¹² The Hadamard product (denoted by $\odot$) takes in two matrices and multiplies the corresponding entries of the two matrices. For example, $(G \odot H) = Q$

$$\begin{bmatrix} 3 & 5 & 7 \\ 4 & 9 & 8 \end{bmatrix} \odot \begin{bmatrix} 1 & 6 & 3 \\ 0 & 2 & 9 \end{bmatrix} = \begin{bmatrix} 3 \times 1 & 5 \times 6 & 7 \times 3 \\ 4 \times 0 & 9 \times 2 & 8 \times 9 \end{bmatrix} \quad [20]$$

Or, in the case of the row normalization:

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} \odot \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ 2 & 2 & 2 \\ 3 & 3 & 3 \\ 4 & 4 & 4 \\ 5 & 5 & 5 \\ 6 & 6 & 6 \end{bmatrix} \quad [21]$$

Thus, ${}_nA_{norm}$ is given by:

$${}_nA_{norm} = {}_nA_{r,c} \odot {}_nD \quad [22]$$

$${}_n A_{norm} = \begin{bmatrix} {}_n a_{1,1} & \cdots & {}_n a_{1,C} \\ \vdots & \ddots & \vdots \\ {}_n a_{R,1} & \cdots & {}_n a_{R,C} \end{bmatrix} \odot \begin{bmatrix} {}_n d_1 \\ \vdots \\ {}_n d_R \end{bmatrix} = \begin{bmatrix} {}_n a_{1,1} {}_n d_1 & \cdots & {}_n a_{1,C} {}_n d_1 & \cdots & {}_n a_{1,C} {}_n d_1 \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ {}_n a_{r,1} {}_n d_r & \cdots & {}_n a_{r,C} {}_n d_r & \cdots & {}_n a_{r,C} {}_n d_r \\ \vdots & \ddots & \vdots & \ddots & \vdots \\ {}_n a_{R,1} {}_n d_R & \cdots & {}_n a_{R,C} {}_n d_R & \cdots & {}_n a_{R,C} {}_n d_R \end{bmatrix} \quad [23]$$

This approach leads to two possible methods for multiple radiographic images. Method one, where each image is normalized before averaging them, and method two, where the images are averaged into one image, and then normalized.

Method one: normalized and then averaged (NTA), using **Equation 23**, is given by:

$$NTA = \frac{1}{N} \sum_{n=1}^N [{}_n A_{norm}] \quad [24]$$

Method two: averaged and then normalized (ATN), is given by:

$$ATN = \left[\frac{1}{N} \sum_{n=1}^N {}_n A_{r,c} \right] \odot (DN) \quad [25]$$

Where (DN) is calculated using the averaged matrix (AN)_{r,c}.

The difference between a FFC image and a normalized FFC image (using ATN) is illustrated below. The normalized image clearly reduces the black and white stripes from the row sequential readout of the detector.

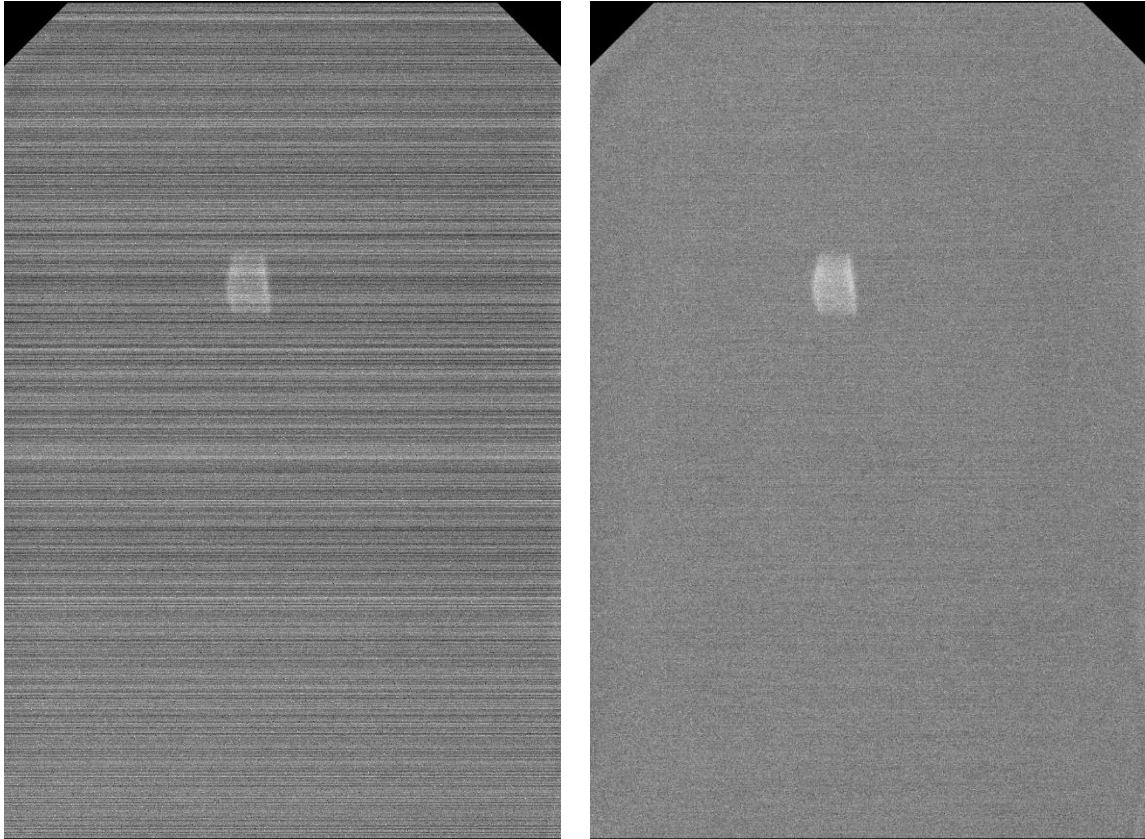


Figure 11 - Comparison of a FFC image (left), and the same image after using method two of normalization, averaging then normalizing (right). This uses the same image as Figure 10.

3.0 Methods and Materials

3.1 Experimental Set up

The detector used was the Hamamatsu S11684-12 complementary metal-oxide semiconductor (CMOS) area image sensor with a row-sequential readout. It consists of 1000×1506 pixels, each measuring $(20 \times 20)\mu m^2$, resulting in an effective image size of $(20 \times 30)mm^2$ when ignoring the shielded pixels. The shielded regions are divided into upper and lower sections: the lower section comprises the bottom three rows, each with 1000 columns, while the upper section includes the top three rows and two triangular areas extending 114 pixels down and 114 pixels inwards on each side due to the detector's shape. **Figure 12** illustrates the detector's dimensions. The sensor supports USB 2.0 connectivity and required a powered USB for operation.

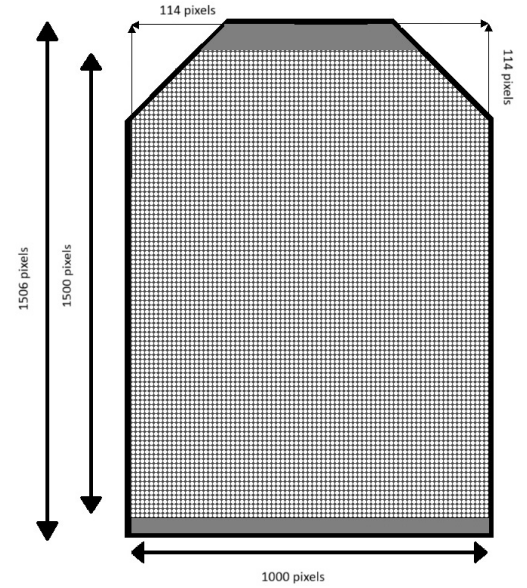


Figure 12 - Hamamatsu S11684-12 detector pixel layout.

The powered USB used during testing was the 5-meter StarTech USB 2.0 Active Extension Cable (model #USB2AAEXT5M). The powered USB connected the detector to the lab's Windows 10 computer, where the Hamamatsu S11684/S11685 Image Acquisition Application was used to capture images. This software is designed to evaluate the Hamamatsu S11684-12 detector and supports both manual and automatic capture modes. Due to the limited photon fluence from the pinhole, all images were captured using manual mode.

A custom case was 3D-printed by a colleague, Jack Rubio, to securely hold the detector. Its dimensions, shown in **Figure 13**, were designed to fit the detector snugly, relying on gravity to keep it

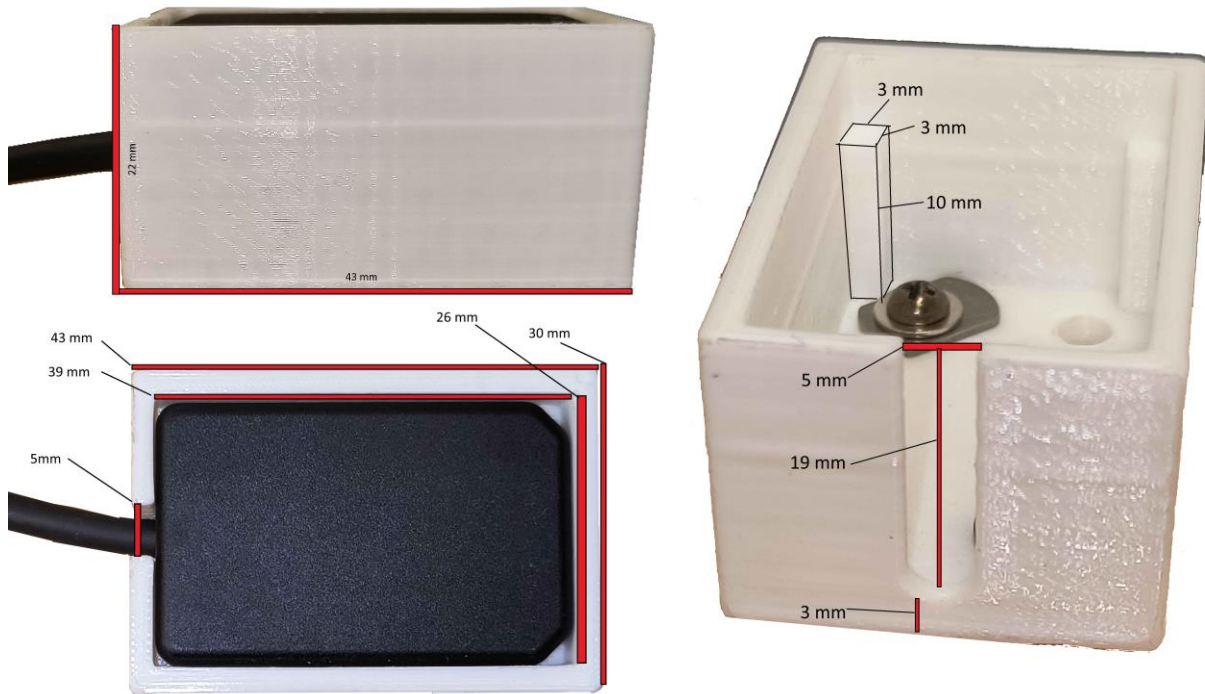


Figure 13 - Side, top, and front view of the 3D printed Hamamatsu S11648-12 detector case.

in place. Green painter's tape was applied to prevent any shifting within the case. Additionally, screw holes in the bottom of the case allowed it to be secured to a larger structure, preventing tipping.

The x-ray generator was the Picker GX550, a single-phase generator commonly used for medical imaging. It includes adjustable controls for exposure time, mA, and kVp. Paired with the generator was the Picker Dunlee PX 412A x-ray tube (insert model #DU304), with specifications shown in **Figure 14**. The x-ray tube has two focal spots: a small focal spot with a nominal value of 0.6 and a large focal spot with a nominal value of 1.2.

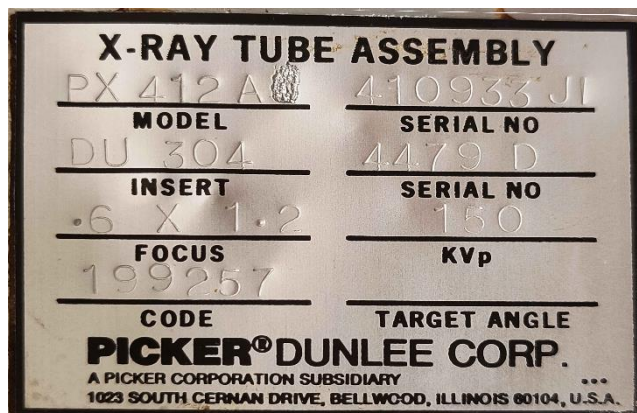


Figure 14 - X-ray Tube Serial Plate.

The imaging setup utilized RaySafe (formerly Fluke Biomedical) x-ray pinholes.¹³ Available were a 30 μm pinhole (model 07-613) and a 75 μm pinhole (model 07-617), both with diaphragms composed of a 90:10 gold-platinum alloy. A DVD stand, the Atlantic Onyx 28 DVDs Blurays Tower Wall Mounted Or Free Standing (Matte Black) 1331 was chosen by Eva Anderson to hold the pinhole for imaging.¹⁴ It has a height of 50.8 cm, a width of 21.59 cm, and a depth of 9.52 cm. The shelves are spaced roughly 19 mm apart, with slight variations between 18 – 20 mm. The shelves adequately accommodated the pinhole, which was affixed to a CD disk case using green painters tape, as illustrated in **Figure 15**.

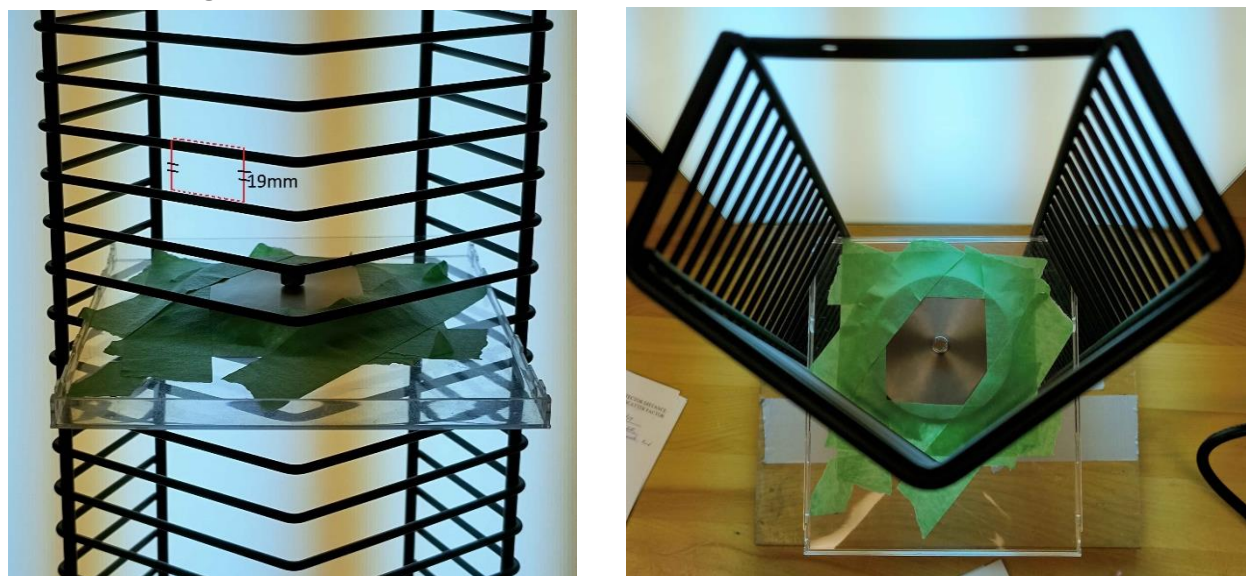


Figure 15 - Pinhole placement using the DVD shelving unit. Front view (left) and top view (right) show the pinhole being aligned with roughly the centre of the stand.

To ensure alignment with the central axis, Dr. Tong Xu provided a hollow metal pipe (**Figure 16**) that he modified by machining one end flat to allow it to stand upright. Radiographs were then captured with the pipe placed directly on the detector. To achieve proper alignment, the detector was adjusted until the radiographic image of the pipe displayed a perfect circle. If the image appeared thicker in one direction, the detector was repositioned toward the thicker side of the pipe's radiographic projection to correct the misalignment.

3.2 Capturing a Radiographic Pinhole Image

To capture a radiographic pinhole image using the Hamamatsu S11684-12 detector, the following procedure was performed:

Device Setup:

1. Connect all devices:
 - USB power cord
 - Detector to USB
 - USB to computer
2. Turn on the computer and open the image acquisition software when all devices are plugged in.

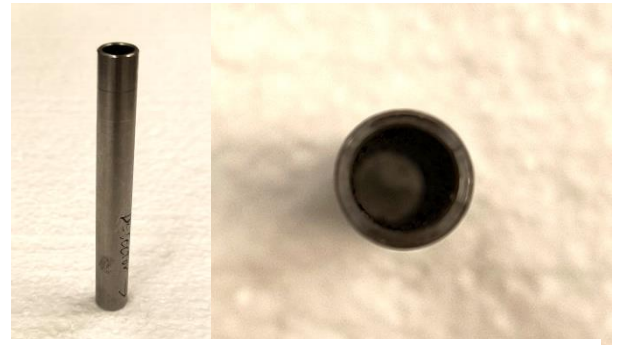


Figure 16 - Hollow pipe front view (left) and top view (right).

Software configuration: Navigate to *Settings > Image Acquisition*:

- I. Set *Integration mode* to “For X-ray image acquisition.”
- II. Deselect “Subtract the stored dark image for fast acquisition.”
- III. Select “Raw image” in the *Image type* dropdown.
- IV. Set *Integration time* to 1000 *ms* and click “OK.”

Detector Alignment:

1. Position the detector in the DVD stand at the preferred height (d_3).
2. Turn on the x-ray generator and set milliamperes, exposure time, and accelerating potential.
3. Align the detector with the central axis using the hollow metal pipe.
4. Secure the detector with tape and record the distance to the focal spot (d_3).

Pinhole Placement and Image Acquisition:

5. Place the pinhole at $d_2 \cong 10\text{ cm}$ from the detector to increase the brightness of the image for alignment.
6. Spin the rotor, then click “Acquire Image” and immediately generate x-rays within the set integration time. If the focal spot cannot be seen, check pinhole alignment.

Fine-Tune Pinhole Alignment:

7. Adjust the pinhole height (d_2) in small increments to keep the pinhole centered in the image until the desired enlargement is reached.
8. Measure its height relative to the detector (d_2), from the top of the pinhole to the top of the detector.
9. Reduce the *Integration time* down to 500 *ms* to minimize noise.

Capture and Save Images: Acquire and save the following as 16-bit TIFFs:

- 10 RAW pinhole images.
- 10 RAW dark images.
- 10 RAW gain images.

Shutdown: Turn off the computer and the x-ray generator, and unplug all devices.

3.3 Measuring Nominal Value

The focal spot measurement is performed using a custom MATLAB (R2024b) script (Nominal_Value_Finder.m, **Appendix A.1**) that applies FFC and row-sequential readout normalization to raw x-ray image data. The script outputs the focal spot dimensions (width and height) in both pixels and millimeters.

Image Input and Preprocessing: The script (**Appendix A.2**) accepts three sets of 16-bit *TIF* images:

- Raw x-ray images
- Raw dark images
- Raw gain images

Each image is converted into a 1000×1506 numerical array. A custom mask is applied to exclude shielded pixels from all subsequent calculations.

The script (**Appendix A.2.i**) averages the input arrays to produce:

- An averaged x-ray array
- An averaged dark array
- An averaged gain array

The ATN technique (**Equation 25**) is applied to these averaged arrays. (**Appendix B**)

Normalization and Output: A manual column selection function allows the user to exclude the focal spot region from row-wise normalization. This is performed by selecting a section from the averaged x-ray array and is then applied across the dark and gain arrays using the same selection. Flat field correction (**Equation 14**) is performed using the three averaged arrays, resulting in a corrected image array. This corrected array is saved as both a *TIF* image and a *MAT* data file to preserve data integrity.

The script (**Appendix A.3**) then automatically adjusts the window and level of the corrected array to produce a visually enhanced version (**Figure 17**).

Focal Spot Analysis: The user provides input parameters:

- Pinhole size

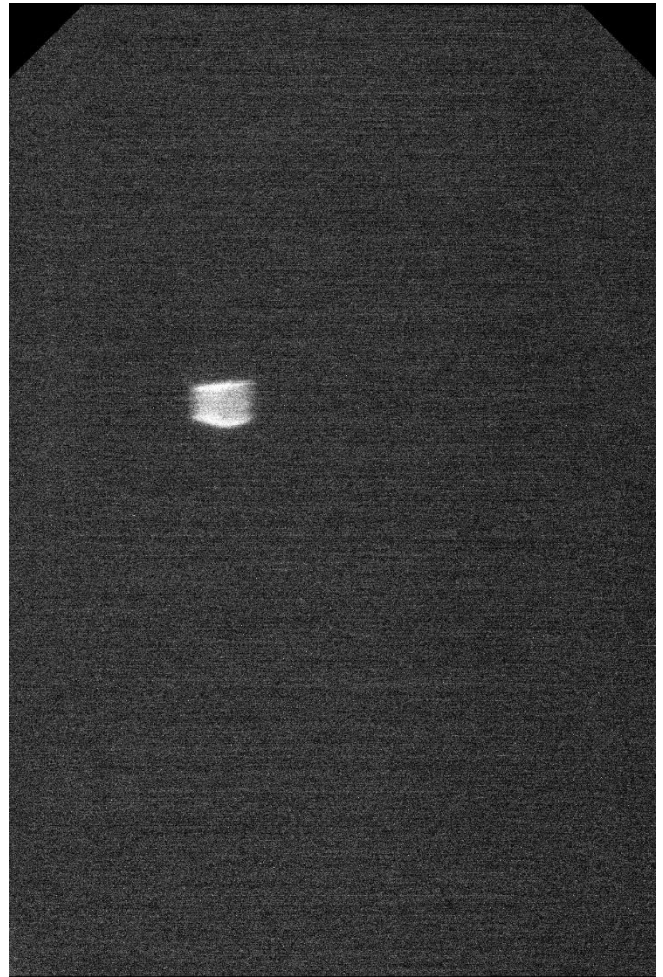


Figure 17 - A fully normalized and FFC image of the Large focal spot. This image is not rotated or cropped, but is corrected for window and level.

- Distances d_2 and d_3

Using the displayed image (**Figure 17**), the user selects a region of interest (*ROI*) containing:

- The focal spot (now visible after correction and adjustment)
- A background area big enough for comparison

This ROI is cropped, and the user aligns the focal spot horizontally and vertically to simplify the procedure and calculations. The script applies the MATLAB *imrotate* function (using "nearest" interpolation) to perform the alignment, where the output pixel takes the value of the nearest input pixel without considering surrounding values.

The user selects two additional ROIs (**Figure 18**) from the now cropped and rotated image:

1. Focal Spot ROI: Encompassing the entire focal spot without excess background.
2. Background ROI: Excluding the focal spot and occupying as large an area as possible.

The script projects the aligned focal spot into horizontal and vertical intensity vectors to calculate its dimensions in both pixels and millimeters. Enlargement (E) and total blur (B_{tot}) are computed and used to adjust these measurements.

Final Output: The script outputs the corrected focal spot width and height measurements and prompts the user to save the final .*MAT* data file, concluding the analysis. The calculated dimensions can subsequently be converted into a nominal focal spot value.

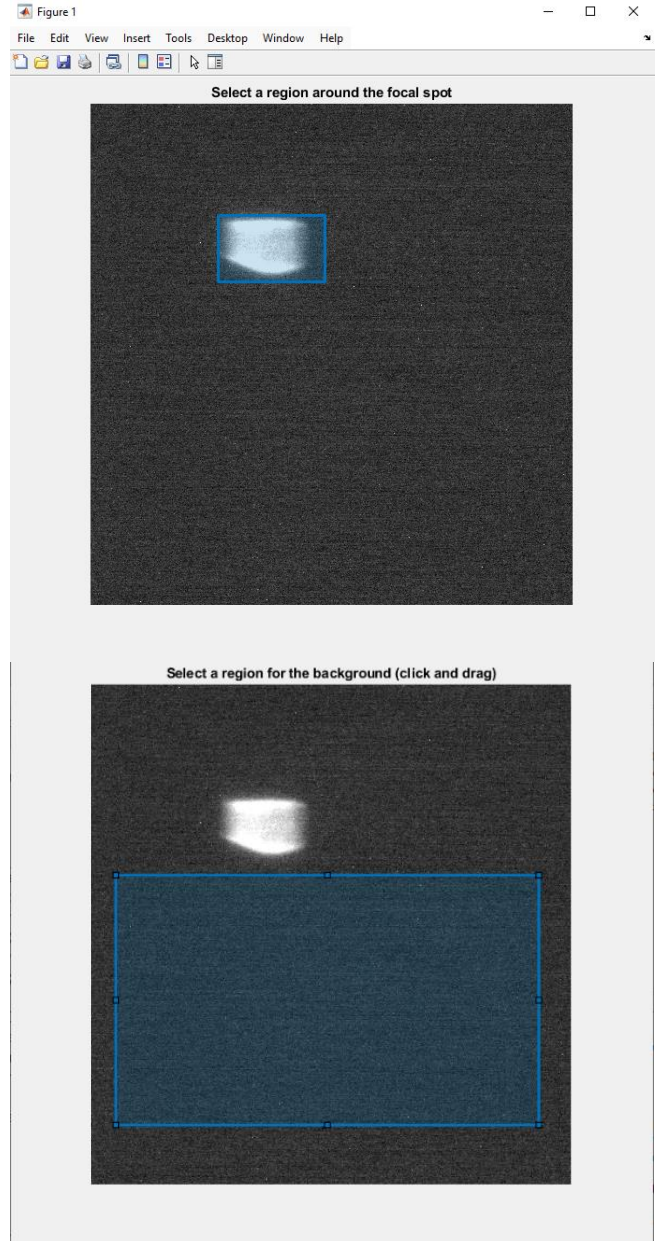


Figure 18 - Screenshot of the Focal Spot (top) and Background (bottom) ROI selection. These are the cropped and rotated versions of the image in Figure 17.

4.0 Analysis

4.1 Matrix Analysis

To compare the performance of the two normalization methods, NTA and ATN, a series of controlled tests were conducted. These tests were designed to assess each method's ability to handle small fluctuations in an otherwise constant dataset. The objectives of these tests were as follows:

- Minimizing row-wise deviations.
- Reducing the impact of the deviations on the rest of the matrix after normalization.
- Minimizing computation time.

For the simplest test, three initial matrices ${}_1A_{r,c}$, ${}_2A_{r,c}$, and ${}_3A_{r,c}$ were established. ${}_1A_{r,c}$ was given a single fluctuation, but ${}_2A_{r,c}$ and ${}_3A_{r,c}$ both contained only a constant value. The matrices were:

$${}_1A_{r,c} = \begin{bmatrix} \mathbf{11} & 10 & 10 & 10 & 10 \\ 10 & 10 & 10 & 10 & 10 \\ 10 & 10 & 10 & 10 & 10 \\ 10 & 10 & 10 & 10 & 10 \\ 10 & 10 & 10 & 10 & 10 \end{bmatrix} \quad [26]$$

$${}_2A_{r,c} = {}_3A_{r,c} = \begin{bmatrix} 10 & 10 & 10 & 10 & 10 \\ 10 & 10 & 10 & 10 & 10 \\ 10 & 10 & 10 & 10 & 10 \\ 10 & 10 & 10 & 10 & 10 \\ 10 & 10 & 10 & 10 & 10 \end{bmatrix} \quad [27]$$

The **bolded** number in ${}_1A$ is the fluctuation. With these three matrices, ATN and NTA will be shown below. To follow the fluctuation's effect on the rest of the matrix/matrices, if a number is affected by the fluctuation, the number will be in **bold**.

Averaging, Then Normalizing (ATN)

ATN can be calculated in five steps with each matrix having $r = 1 \rightarrow R = 5$ rows, $c = 1 \rightarrow C = 5$ columns, and $n = 1 \rightarrow N = 3$ matrices.

Step 1: Average over N , where $(AN)_{r,c}$ will denote that the matrix ${}_nA_{r,c}$ has been averaged over N :

$$(AN)_{r,c} = \frac{1}{N} \sum_{n=1}^N {}_nA_{r,c} \quad [28]$$

$$(AN)_{r,c} = \frac{1}{3} \begin{bmatrix} \mathbf{11} + 10 + 10 & 10 + 10 + 10 & 10 + 10 + 10 & 10 + 10 + 10 & 10 + 10 + 10 \\ 10 + 10 + 10 & 10 + 10 + 10 & 10 + 10 + 10 & 10 + 10 + 10 & 10 + 10 + 10 \\ 10 + 10 + 10 & 10 + 10 + 10 & 10 + 10 + 10 & 10 + 10 + 10 & 10 + 10 + 10 \\ 10 + 10 + 10 & 10 + 10 + 10 & 10 + 10 + 10 & 10 + 10 + 10 & 10 + 10 + 10 \\ 10 + 10 + 10 & 10 + 10 + 10 & 10 + 10 + 10 & 10 + 10 + 10 & 10 + 10 + 10 \end{bmatrix} \quad [29]$$

$$= \begin{bmatrix} \mathbf{10.33} & 10 & 10 & 10 & 10 \\ 10 & 10 & 10 & 10 & 10 \\ 10 & 10 & 10 & 10 & 10 \\ 10 & 10 & 10 & 10 & 10 \\ 10 & 10 & 10 & 10 & 10 \end{bmatrix} \quad [30]$$

Step 2: Average each row using **Equation 15**, where $(ANC)_r$ will denote that the matrix ${}_nA_{r,c}$ has been averaged over both N and C :

$$(ANC)_r = \frac{1}{C} \sum_{c=1}^C (AN)_{r,c} = \frac{1}{5} \begin{bmatrix} \mathbf{10.33} + (4 \times 10) \\ 5 \times 10 \\ 5 \times 10 \\ 5 \times 10 \\ 5 \times 10 \end{bmatrix} = \begin{bmatrix} \mathbf{10.066} \\ 10 \\ 10 \\ 10 \\ 10 \end{bmatrix} \quad [31]$$

Step 3: Average the columns of $(ANC)_r$ to determine the average pixel value, $\langle aN \rangle$, of the matrix $(AN)_{r,c}$:

$$\langle aN \rangle = \frac{1}{R} \sum_{r=1}^R (ANC)_r = \frac{1}{5} (\mathbf{10.066} + 10 + 10 + 10 + 10) = \mathbf{10.0133} \quad [32]$$

Step 4: Calculate the row-specific normalization constant $(dN)_r = \frac{\langle aN \rangle}{(ANC)_r}$, where $(dN)_r$ uses the matrix $(ANC)_r$ which was averaged over N :

$$(DN) = \begin{bmatrix} \frac{\mathbf{10.0133}}{\mathbf{10.066}} \\ \frac{\mathbf{10.0133}}{10} \\ \frac{\mathbf{10.0133}}{10} \\ \frac{\mathbf{10.0133}}{10} \\ \frac{\mathbf{10.0133}}{10} \end{bmatrix} = \begin{bmatrix} \mathbf{0.99470199} \\ \mathbf{1.00133} \\ \mathbf{1.00133} \\ \mathbf{1.00133} \\ \mathbf{1.00133} \end{bmatrix} \quad [33]$$

Step 5: ATN can be calculated with **Equation 25**:

$$ATN = (AN)_{r,c} \odot (DN) \quad [34]$$

$$ATN = \begin{bmatrix} \mathbf{10.33} & 10 & 10 & 10 & 10 \\ 10 & 10 & 10 & 10 & 10 \\ 10 & 10 & 10 & 10 & 10 \\ 10 & 10 & 10 & 10 & 10 \\ 10 & 10 & 10 & 10 & 10 \end{bmatrix} \odot \begin{bmatrix} \mathbf{0.99470199} \\ \mathbf{1.00133} \\ \mathbf{1.00133} \\ \mathbf{1.00133} \\ \mathbf{1.00133} \end{bmatrix} \quad [35]$$

$$ATN = \begin{bmatrix} \mathbf{10.2785872} & \mathbf{9.9470199} & \mathbf{9.9470199} & \mathbf{9.9470199} & \mathbf{9.9470199} \\ \mathbf{10.0133} & \mathbf{10.0133} & \mathbf{10.0133} & \mathbf{10.0133} & \mathbf{10.0133} \\ \mathbf{10.0133} & \mathbf{10.0133} & \mathbf{10.0133} & \mathbf{10.0133} & \mathbf{10.0133} \\ \mathbf{10.0133} & \mathbf{10.0133} & \mathbf{10.0133} & \mathbf{10.0133} & \mathbf{10.0133} \\ \mathbf{10.0133} & \mathbf{10.0133} & \mathbf{10.0133} & \mathbf{10.0133} & \mathbf{10.0133} \end{bmatrix} \quad [36]$$

Normalizing, Then Averaging (NTA)

NTA can be calculated using the same notation as above.

Step 1: Average over the rows of each matrix using **Equation 15**, where ${}_n(AC)_r$ will denote that the matrix ${}_nA_{r,c}$ has been averaged over C :

$${}_1(AC)_r = \begin{bmatrix} 10.2 \\ 10 \\ 10 \\ 10 \\ 10 \end{bmatrix} \quad [37]$$

$${}_2(AC)_r = {}_3(AC)_r = \begin{bmatrix} 10 \\ 10 \\ 10 \\ 10 \\ 10 \end{bmatrix} \quad [38]$$

Step 2: Average the columns of each matrix using **Equation 16** to determine the average pixel value, ${}_n\langle a \rangle$, of each matrix:

$${}_1\langle a \rangle = 10.04 \quad [39]$$

$${}_2\langle a \rangle = {}_3\langle a \rangle = 10 \quad [40]$$

Step 3: Calculate the row-specific normalization constant ${}_nd_r = \frac{{}_n\langle a \rangle}{{}_n(AC)_r}$, using **Equation 17**:

$${}_1D = \begin{bmatrix} \frac{10.04}{10.2} \\ \frac{10.04}{10} \\ \frac{10.04}{10} \\ \frac{10.04}{10} \\ \frac{10.04}{10} \end{bmatrix} = \begin{bmatrix} 0.98431372 \\ 1.04 \\ 1.04 \\ 1.04 \\ 1.04 \end{bmatrix} \quad [41]$$

$${}_2D = {}_3D = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} \quad [42]$$

Step 4: Normalize the individual matrices using **Equation 22**:

$${}_1A_{norm} = \begin{bmatrix} 11 & 10 & 10 & 10 & 10 \\ 10 & 10 & 10 & 10 & 10 \\ 10 & 10 & 10 & 10 & 10 \\ 10 & 10 & 10 & 10 & 10 \\ 10 & 10 & 10 & 10 & 10 \end{bmatrix} \odot \begin{bmatrix} 0.98431372 \\ 1.04 \\ 1.04 \\ 1.04 \\ 1.04 \end{bmatrix} \quad [43]$$

$${}_1A_{norm} = \begin{bmatrix} 10.827451 & 9.8431372 & 9.8431372 & 9.8431372 & 9.8431372 \\ 10.04 & 10.04 & 10.04 & 10.04 & 10.04 \\ 10.04 & 10.04 & 10.04 & 10.04 & 10.04 \\ 10.04 & 10.04 & 10.04 & 10.04 & 10.04 \\ 10.04 & 10.04 & 10.04 & 10.04 & 10.04 \end{bmatrix} \quad [44]$$

$${}_2A_{norm} = {}_3A_{norm} = \begin{bmatrix} 10 & 10 & 10 & 10 & 10 \\ 10 & 10 & 10 & 10 & 10 \\ 10 & 10 & 10 & 10 & 10 \\ 10 & 10 & 10 & 10 & 10 \\ 10 & 10 & 10 & 10 & 10 \end{bmatrix} \odot \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 10 & 10 & 10 & 10 & 10 \\ 10 & 10 & 10 & 10 & 10 \\ 10 & 10 & 10 & 10 & 10 \\ 10 & 10 & 10 & 10 & 10 \\ 10 & 10 & 10 & 10 & 10 \end{bmatrix} \quad [45]$$

Step 5: NTA can be calculated using **Equation 24**:

$$NTA = \frac{1}{N} \sum_{n=1}^N [{}_nA_{norm}] = \begin{bmatrix} 10.275817 & 9.947712 & 9.947712 & 9.947712 & 9.947712 \\ 10.0133 & 10.0133 & 10.0133 & 10.0133 & 10.0133 \\ 10.0133 & 10.0133 & 10.0133 & 10.0133 & 10.0133 \\ 10.0133 & 10.0133 & 10.0133 & 10.0133 & 10.0133 \\ 10.0133 & 10.0133 & 10.0133 & 10.0133 & 10.0133 \end{bmatrix} \quad [46]$$

Both methods produced similar results. Comparing **Equations 36 and 46** by analyzing the specific fluctuation point $r, c = 1, 1$:

The difference $\Delta\gamma_{1,1}$ between the fluctuation and the normalized arrays are:

$$\Delta\gamma_{1,1,ATN} = 11 - 10.275817 = 0.724183 \quad [47]$$

$$\Delta\gamma_{1,1,NTA} = 11 - 10.275817 = 0.724183 \quad [48]$$

where, since $\Delta\gamma_{1,1,NTA} > \Delta\gamma_{1,1,ATN}$, NTA better minimizes the fluctuation at the deviation point. Similarly, analyzing the row of the fluctuation ($r = 1$) for values where $c \neq 1$, finds:

$$\Delta\gamma_{1,c \neq 1,ATN} = 10 - 9.9470199 = 0.0529801 \quad [49]$$

$$\Delta\gamma_{1,c \neq 1,NTA} = 10 - 9.9477124 = 0.0522876 \quad [50]$$

Here, $\Delta\gamma_{1,c \neq 1,NTA} < \Delta\gamma_{1,c \neq 1,ATN}$, indicating that NTA reduces the impact of the fluctuation on neighbouring values in the same row compared to ATN.

As shown between **Equations 36 and 46**, non-deviated rows are affected by the same amount no matter the condition performed.

The computational demands of NTA are higher than those of ATN, particularly as the number of input matrices increases. For cases with low N , NTA is preferred due to its superior ability to limit the propagation of fluctuations. The number of matrices involved in calculations for the purpose of nominal value measurements will be low, thus NTA is the preferred method.

4.1 Analyzing the Effective Focal Spot Data

As outlined in *Section 1.2*, the goal of this research is to measure the nominal value of the focal spots using the Hamamatsu detector in compliance with IEC requirements. The relevant IEC criteria (*Section 6.3.2*) include:

- The number of pixels over the 15% width of the line spread function is at least 60.
- The signal to noise ratio (SNR) must be greater than 100.
- The number of levels between the background level and maximum signal of the focal spot must be 200 or more.
- The image must be aligned within 1° of the reference axis.

Analysis of the images revealed several challenges:

- The SNR consistently fell below 100, failing to meet IEC compliance.
- Aligning the image within 1° of the reference axis was difficult, as the tube is enclosed in a metal case and not visible for exact measurements.

Thus, only the tests that met both criteria (a) and (c) were selected for detailed examination. Tests that failed to meet these requirements, such as all the *small focal spot* tests using the 30 μm pinhole, which all failed due to the signal level being below 200, were excluded from further analysis.

To ensure consistency in testing and measurements, all pixel values from the passing tests were rescaled to a range of 0 to 65,000 (approximately 16-bits). This rescaling was performed only after verifying that the signal difference between the maximum intensity and the background met the IEC's required 200 level difference. Line spread functions were then projected from these rescaled values and all evaluations are performed below. The passing test's conditions are summarized in *Table 1*.

Table 1 - Test Conditions and Enlargement Measurements of both the large focal spot (Tests A to F) and small focal spot (Tests G to J).

Test	Focal Spot	$(d_3 \pm 10)$ mm	$(d_2 \pm 10)$ mm	Enlargement (E)	Pinhole Size ± 0.005 mm	Tube Current (mA)	Exposure Time (ms)	Accelerating Potential (kVp)
A	Large	661	268	0.69 ± 0.024	0.030	200	100	60
B	Large	661	283	0.75 ± 0.028	0.030	200	100	60
C	Large	661	298	0.82 ± 0.032	0.030	200	100	60
D	Large	681	294	0.76 ± 0.028	0.075	200	100	60
E	Large	623	256	0.70 ± 0.027	0.030	200	200	60
F	Large	623	298	0.92 ± 0.040	0.030	200	200	60
G	Small	681	322	0.90 ± 0.035	0.075	50	200	60
H	Small	681	367	1.17 ± 0.053	0.075	100	100	60
I	Small	681	379	1.26 ± 0.059	0.075	50	200	60
J	Small	681	379	1.26 ± 0.059	0.075	100	200	60

The enlargement factor (E) is calculated using **Equation 4**. With its uncertainty propagated as follows:

$$\sigma_E = \sqrt{\left(\frac{\partial E}{\partial d_3} \sigma_{d_3}\right)^2 + \left(\frac{\partial E}{\partial d_2} \sigma_{d_2}\right)^2} \quad [51]$$

$$\sigma_E = \sqrt{\left(\frac{-d_2 \sigma_{d_3}}{(d_3 - d_2)^2}\right)^2 + \left(\frac{d_3 \sigma_{d_2}}{(d_2 - d_3)^2}\right)^2} \quad [52]$$

where σ_{d_3} and σ_{d_2} represent the uncertainties in d_3 and d_2 respectively.

Using the MATLAB scripts mentioned in *Section 3.3*, the tests from *Table 1* output the corrected focal spot height and width measurements by calculating the total blur (B_{tot}) in the source plane from **Equation 9**. The detector blur α is related to the spatial resolution R in lp/mm by:

$$\alpha = \frac{1}{2} \times \frac{1}{R} \quad [53]$$

where the factor of $\frac{1}{2}$ accounts for halving the line pairs, ensuring α is expressed in millimeters. From **Figure 9**, the resolution is approximately $(18 \pm 1) \frac{lp}{mm}$, so the intrinsic blur of the detector can be calculated as:

$$\alpha \pm \sigma_\alpha = \frac{1}{2 \cdot 18} \pm \sqrt{\left(\frac{-1}{2(18)^2}\right)^2} = (0.028 \pm 0.0015) \text{ mm} \quad [54]$$

Table 2 provides pinhole dimensions and their uncertainties from RaySafe, which are incorporated into total blur calculations.

Table 2 - Pinhole dimensions and their uncertainties.

Model	Diaphragm Dimensions (mm)
07-613	0.030 ± 0.005
07-619	0.075 ± 0.005

Propagating **Equation 9** with the errors calculated above, the error on the total blur $\sigma_{B_{tot}}$ is calculated as:

$$\sigma_{B_{tot}} = \sqrt{\left(\frac{\alpha \sigma_{\alpha}}{E \sqrt{a^2 + p^2(E+1)^2}}\right)^2 + \left(\frac{(-\alpha^2 - p^2(E+1))\sigma_E}{E^2 \sqrt{a^2 + p^2(E+1)^2}}\right)^2 + \left(\frac{p(E+1)^2 \sigma_p}{E \sqrt{a^2 + p^2(E+1)^2}}\right)^2} \quad [55]$$

where the total blur for each test is shown in *Table 3*.

Table 3 – Calculated Total Blur and Uncertainty for each Test.

Test	Enlargement (E)	Pinhole Diameter (± 0.005) mm	Total Blur (mm)
A	0.69 ± 0.046	0.030	0.083 ± 0.0109
B	0.75 ± 0.050	0.030	0.079 ± 0.0105
C	0.82 ± 0.055	0.030	0.074 ± 0.0101
D	0.76 ± 0.050	0.075	0.175 ± 0.0128
E	0.70 ± 0.050	0.030	0.082 ± 0.0109
F	0.92 ± 0.065	0.030	0.069 ± 0.0096
G	0.90 ± 0.058	0.075	0.161 ± 0.0115
H	1.17 ± 0.078	0.075	0.141 ± 0.0100
I	1.26 ± 0.085	0.075	0.136 ± 0.0096
J	1.26 ± 0.085	0.075	0.136 ± 0.0096

With the total blur and enlargement calculated for each test, the width (W) and height (H) of the focal spot can now be determined, including uncertainties. Using the MATLAB scripts (Section 3.3), the effective focal spot dimensions were determined by projecting intensity values along the width and height directions of the focal spot region to generate one-dimensional line spread functions (LSF). Some comparison LSFs are shown in **Figures 19 and 20**.

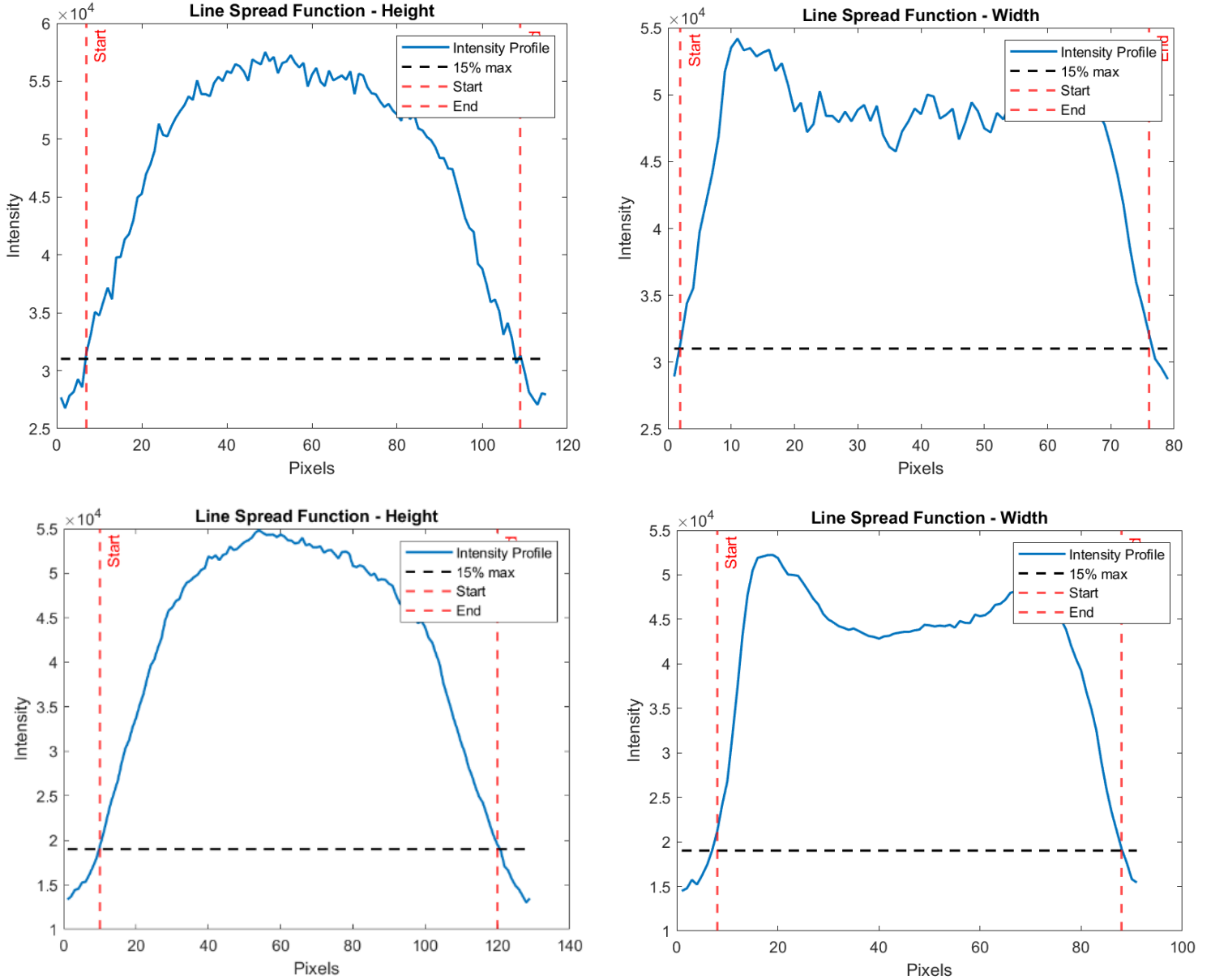


Figure 19 – **Large Focal Spot: Test B (Top) and Test D (Bottom)**

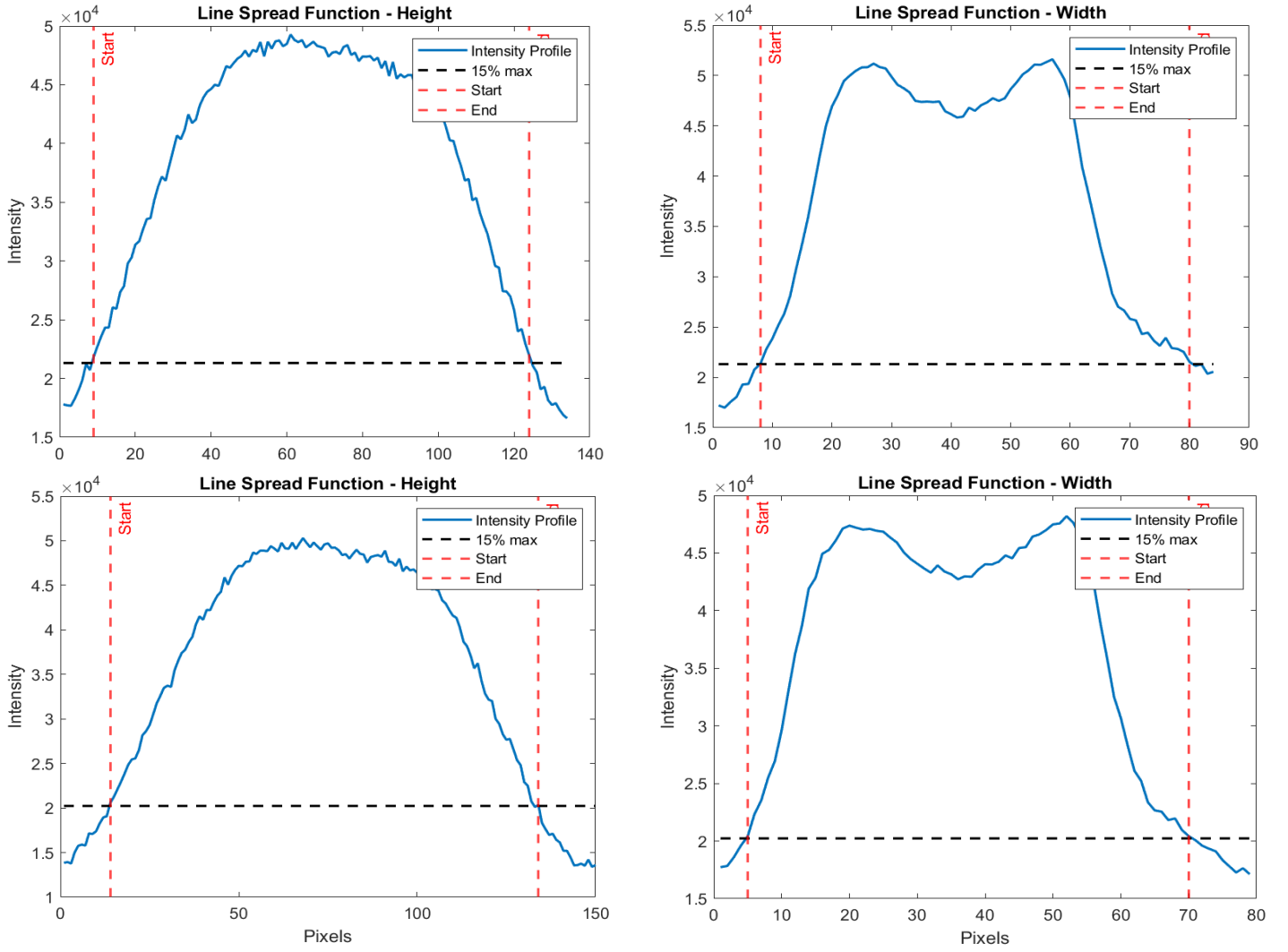


Figure 20 – **Small Focal Spot**: Test I (Top) and Test J (Bottom) comparison.

The number of pixels exceeding 15% of the maximum intensity was converted into physical dimensions using:

$$W_{eff} = N_W \times 0.02 \text{ mm} \quad [56]$$

$$H_{eff} = N_H \times 0.02 \text{ mm} \quad [57]$$

where N_W and N_H represent the number of pixels covering the width and height of the focal spot, respectively. Since the detector has no explicit uncertainty on pixel size, it is reasonable to assume a margin of error of $\pm 0.5 \mu\text{m}$ in each pixel. Adding an uncertainty onto the MATLAB script of $\pm 2 \text{ pixels}$, the effective width and height uncertainties are calculated as follows:

$$\sigma_{W_{eff}} = \sqrt{(0.005 * \sigma_{N_W})^2 + (N_W * \sigma_{pixel \text{ size}})^2} \quad [58]$$

$$= \sqrt{0.0001 + (N_W * 0.002)^2} \quad [59]$$

$$\sigma_{H_{eff}} = \sqrt{0.0001 + (N_H * 0.002)^2} \quad [60]$$

Table 4 - Pixel Coverage of the Focal Spot, Calculated in MATLAB.

Test	$N_W \pm 2 \text{ pixels}$	$N_H \pm 2 \text{ pixels}$	$W_{eff}(\text{mm})$	$H_{eff}(\text{mm})$
A	70	92	1.40 ± 0.14	1.84 ± 0.18
B	75	101	1.50 ± 0.15	2.02 ± 0.20
C	83	108	1.66 ± 0.17	2.16 ± 0.22
D	82	111	1.64 ± 0.16	2.22 ± 0.22
E	65	89	1.30 ± 0.13	1.78 ± 0.18
F	82	107	1.64 ± 0.16	2.14 ± 0.21
G	55	82	1.10 ± 0.11	1.64 ± 0.16
H	64	96	1.28 ± 0.13	1.92 ± 0.19
I	71	113	1.42 ± 0.14	2.26 ± 0.23
J	69	119	1.38 ± 0.14	2.38 ± 0.24

Equation 9 gives the blur in the focal spot plane, so the height and width will be calculated to the focal spot plane as well. Both the blur and enlargement can be calculated linearly in the focal spot plane as follows:

$$W_{lin} = \frac{W_{eff}}{E} - B_{tot} \quad [61]$$

$$H_{lin} = \frac{H_{eff}}{E} - B_{tot} \quad [62]$$

Alternatively, quadrature corrections were applied:

$$W_{quad} = \sqrt{\left(\frac{W_{eff}}{E}\right)^2 - B_{tot}^2} \quad [63]$$

$$H_{quad} = \sqrt{\left(\frac{H_{eff}}{E}\right)^2 - B_{tot}^2} \quad [64]$$

Given two separate ways to calculate the final value, the average of both corrections will be taken.

Finally, the uncertainties on the linear width and height are:

$$\sigma_{W_{lin}} = \sqrt{\left(\frac{\sigma_{W_{eff}}}{E}\right)^2 + (-\sigma_{B_{tot}})^2 + \left(-\frac{W_{eff} * \sigma_E}{E^2}\right)^2} \quad [65]$$

$$\sigma_{H_{lin}} = \sqrt{\left(\frac{\sigma_{H_{eff}}}{E}\right)^2 + (-\sigma_{B_{tot}})^2 + \left(-\frac{H_{eff} * \sigma_E}{E^2}\right)^2} \quad [66]$$

And the uncertainties on the quadrature width and height are:

$$\sigma_{W_{quad}} = \sqrt{\left(\frac{W_{eff} * \sigma_{W_{eff}}}{E^2 \sqrt{\left(\frac{W_{eff}}{E}\right)^2 - B_{tot}^2}}\right)^2 + \left(-\frac{B_{tot} * \sigma_{B_{tot}}}{\sqrt{\left(\frac{W_{eff}}{E}\right)^2 - B_{tot}^2}}\right)^2 + \left(-\frac{W_{eff} * \sigma_E}{E^3 \sqrt{\left(\frac{W_{eff}}{E}\right)^2 - B_{tot}^2}}\right)^2} \quad [67]$$

$$\sigma_{H_{quad}} = \sqrt{\left(\frac{H_{eff} * \sigma_{H_{eff}}}{E^2 \sqrt{\left(\frac{H_{eff}}{E}\right)^2 - B_{tot}^2}}\right)^2 + \left(-\frac{B_{tot} * \sigma_{B_{tot}}}{\sqrt{\left(\frac{H_{eff}}{E}\right)^2 - B_{tot}^2}}\right)^2 + \left(-\frac{H_{eff} * \sigma_E}{E^3 \sqrt{\left(\frac{H_{eff}}{E}\right)^2 - B_{tot}^2}}\right)^2} \quad [68]$$

These final widths and heights provide a corrected calculation of the effective focal spot size, accounting for blur and enlargement. The results are shown in *Table 5*.

Table 5 - Corrected width and height measurements of the large and small focal spots.

Test	$W_{eff}(mm)$	$H_{eff}(mm)$	$W_{lin}(mm)$	$H_{lin}(mm)$	$W_{quad}(mm)$	$H_{quad}(mm)$
A	1.40 ± 0.14	1.84 ± 0.18	1.95 ± 0.24	2.58 ± 0.32	2.03 ± 0.23	2.67 ± 0.28
B	1.50 ± 0.15	2.02 ± 0.20	1.92 ± 0.24	2.61 ± 0.32	2.00 ± 0.22	2.69 ± 0.28
C	1.66 ± 0.17	2.16 ± 0.22	1.95 ± 0.24	2.56 ± 0.32	2.02 ± 0.22	2.63 ± 0.28
D	1.64 ± 0.16	2.22 ± 0.22	1.98 ± 0.26	2.74 ± 0.35	2.15 ± 0.23	2.92 ± 0.31
E	1.30 ± 0.13	1.78 ± 0.18	1.77 ± 0.23	2.46 ± 0.31	1.86 ± 0.21	2.54 ± 0.27
F	1.64 ± 0.16	2.14 ± 0.21	1.72 ± 0.22	2.26 ± 0.29	1.79 ± 0.20	2.33 ± 0.25
G	1.10 ± 0.11	1.64 ± 0.16	1.06 ± 0.15	1.66 ± 0.22	1.21 ± 0.14	1.82 ± 0.20
H	1.28 ± 0.13	1.92 ± 0.19	0.95 ± 0.13	1.50 ± 0.20	1.08 ± 0.12	1.63 ± 0.17
I	1.42 ± 0.14	2.26 ± 0.23	0.99 ± 0.14	1.66 ± 0.22	1.12 ± 0.13	1.79 ± 0.19
J	1.38 ± 0.14	2.38 ± 0.24	0.96 ± 0.13	1.75 ± 0.23	1.09 ± 0.12	1.88 ± 0.19

The signal-to-noise ratio (SNR) is calculated using the focal spot mean $\langle FS \rangle$ and the standard deviation of the background region, σ_{BG} :

$$SNR = \frac{\langle FS \rangle}{\sigma_{BG}} \pm \sqrt{\left(\frac{[STDM]_{FS}}{\sigma_{BG}}\right)^2 + \left(-\frac{\langle FS \rangle}{\sigma_{BG}^2} [SE]_{BG}\right)^2} \quad [69]$$

where the uncertainty on the mean focal spot value is $[STDM]_{FS}$, calculated using the *standard deviation of the mean* formula:

$$[STDM]_{FS} = \frac{\langle FS \rangle_{std}}{\sqrt{N}} \quad [70]$$

with N being the sample size. The uncertainty on the standard deviation of the background, $[SE]_{BG}$, is calculated using the *standard error of the standard deviation* formula:

$$[SE]_{BG} = \frac{\sigma_{BG}}{\sqrt{2N-2}} \quad [71]$$

where N is the sample size. The SNR values are shown in *Table 6*.

Table 6 - Calculated Signal-to-Noise Ratio values.

Test	$\langle BG \rangle \pm \sigma_{BG}$	$[SE]_{BG}$	$\langle FS \rangle \pm [STDM]_{FS}$	SNR
A	$(24.0 \pm 6.9) \times 10^3$	$\pm 0.02 \times 10^3$	$(49.7 \pm 0.2) \times 10^3$	7.18 ± 0.03
B	$(24.9 \pm 7.2) \times 10^3$	$\pm 0.02 \times 10^3$	$(46.2 \pm 0.2) \times 10^3$	6.41 ± 0.03
C	$(24.9 \pm 7.3) \times 10^3$	$\pm 0.03 \times 10^3$	$(47.3 \pm 0.1) \times 10^3$	6.47 ± 0.03
D	$(10.8 \pm 2.7) \times 10^3$	$\pm 0.01 \times 10^3$	$(42.4 \pm 0.2) \times 10^3$	15.7 ± 0.07
E	$(13.4 \pm 3.8) \times 10^3$	$\pm 0.01 \times 10^3$	$(39.7 \pm 0.2) \times 10^3$	10.57 ± 0.06
F	$(13.9 \pm 4.0) \times 10^3$	$\pm 0.01 \times 10^3$	$(42.0 \pm 0.2) \times 10^3$	10.44 ± 0.06
G	$(12.8 \pm 4.7) \times 10^3$	$\pm 0.01 \times 10^3$	$(37.8 \pm 0.2) \times 10^3$	8.02 ± 0.06
H	$(14.0 \pm 4.2) \times 10^3$	$\pm 0.02 \times 10^3$	$(38.0 \pm 0.2) \times 10^3$	8.95 ± 0.06
I	$(13.5 \pm 4.0) \times 10^3$	$\pm 0.02 \times 10^3$	$(38.0 \pm 0.2) \times 10^3$	9.60 ± 0.06
J	$(12.2 \pm 4.2) \times 10^3$	$\pm 0.01 \times 10^3$	$(37.8 \pm 0.2) \times 10^3$	8.89 ± 0.05

The calculations and analysis presented throughout this section allow for a detailed analysis on each changing variable:

- Pinhole Size
- Enlargement
- Exposure Time
- Tube Current

Pinhole Size

From **Figure 19**, the primary variable between Test B and Test D was the pinhole diameter. The increased smoothness observed in the LSF of Test D may be attributed to this difference. The larger $75 \mu\text{m}$ pinhole used in Test D inherently produces more geometric blur, which smooths out fine details and results in a smoother LSF. This larger aperture also allows more x-rays to pass through, improving the SNR. In contrast, the $30 \mu\text{m}$ pinhole used in Test B reduces blur and provides a sharper, seemingly more accurate depiction of the focal spot's spatial distribution. However, the smaller aperture also restricts photon throughput, decreasing the SNR. The SNR of Test D is 15.7 ± 0.07 , while the smaller pinhole reduced the SNR of Test B to only 6.41 ± 0.03 .

Although these values fall well below the IEC requirement of an $SNR \geq 100$, they remain sufficient for spatial analysis. Therefore, the smaller pinhole is preferable when higher spatial resolution is required, provided the resulting images still satisfy the IEC criteria of at least 200 signals between the maximum and background, as well as a width at 15% maximum intensity covering at least 60 pixels.

Exposure Time

The effect of exposure time is demonstrated by comparing Test A (100 ms) and Test E (200 ms). As expected, the longer exposure time resulted in an improved SNR, increasing from 5.28 ± 0.03 in Test A to 7.35 ± 0.04 in Test E, likely due to greater photon accumulation. This SNR increase, by a factor of approximately 1.39, closely matches the theoretical prediction of $\sqrt{2} \cong 1.41$ for Poisson statistics, where signal increases linearly with exposure time but noise increases with the square root of signal. This supports the expectation that longer integration times improve image quality by increasing the dominance of true signal over random fluctuations.

It is worth noting that the highest SNR was observed in Test D (15.7 ± 0.07), which used a larger pinhole, emphasizing that both exposure time and pinhole size influence signal strength. In contrast, Test B, changing only the pinhole size, had the lowest SNR at 6.41 ± 0.03 . Comparisons between Test A and Test E, and Test B and Test D, suggest that while exposure time does improve both spatial resolution and SNR, the pinhole size has a more significant impact on both SNR and spatial dimensions than exposure time alone.

Enlargement

Tests A, B, and C varied only by the enlargement factor, with values of 0.69 ± 0.02 , 0.75 ± 0.03 , and 0.82 ± 0.03 , respectively. From Table 3 and Figure 21, as expected, greater enlargement reduced the total blur. A similar trend was observed between Tests G and I. Test G, with an enlargement of 0.90 ± 0.06 , had a total blur of $(0.161 \pm 0.012) \text{ mm}$, while Test I, with an enlargement of 1.26 ± 0.06 , showed a total blur of $(0.126 \pm 0.01) \text{ mm}$.

The final corrected measurements of the focal spot's width and height remained consistent across varying enlargement levels (Table 5), suggesting that enlargement has a minimal influence on the corrected focal spot dimensions compared to other varying factors.

Tube Current

Tests I and J differed only in tube current, with Test I conducted at 50 mA and Test J at 100 mA . Theoretically, increasing the tube current should slightly increase the focal spot size due to electrostatic repulsion among the increased electron population. However, the results from Tests I and J do not show conclusive evidence of this relationship, as Test I has a larger width but a smaller height than Test J. This variation is most likely due to statistical fluctuations rather than experimental factors such as detector or pinhole misalignment or variations in filament temperature. Consequently, the effect of the tube current on the focal spot size remains inconclusive and warrants further investigation.

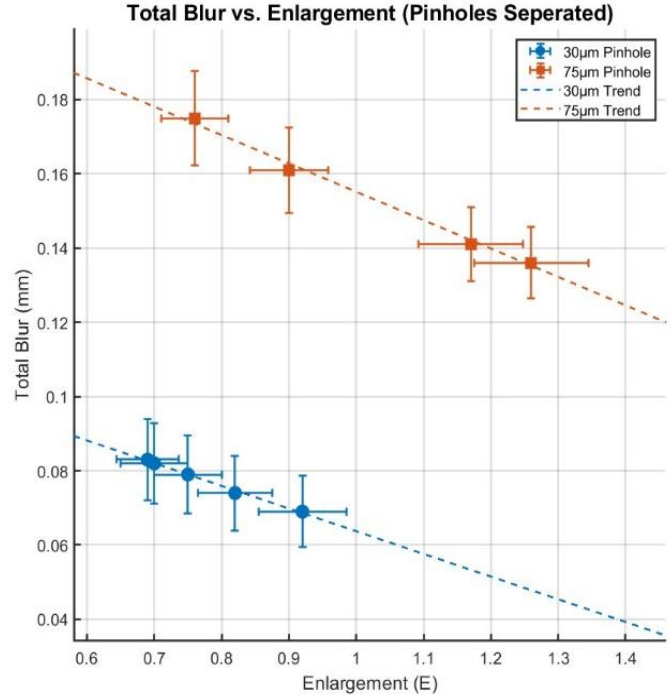


Figure 21 - Total Blur as a function of Enlargement. The orange points corresponding to the 75-micron pinhole, and the blue points correspond to the 30-micron pinhole. The blur decreases as enlargement increases.

4.3 Nominal Value Analysis

The IEC provides a table (Section 7.3.2, Table 3) titled “Maximum permissible values of focal spot dimensions for nominal focal spot values”, where the nominal focal spot value is assigned by using the width and length of the measured focal spot in millimeters. Values from the table are reproduced in Table 7.

Table 7 - IEC table for Maximum permissible values of focal spot dimensions for nominal focal spot values.

Nominal Focal Spot Value	Focal Spot dimensions, Maximum permissible values <i>mm</i>	
f	Width	Length
0.5	0.75	1.10
0.6	0.90	1.30
0.7	1.10	1.50
0.8	1.20	1.60
0.9	1.30	1.80
1.0	1.40	2.00
1.1	1.50	2.20
1.2	1.70	2.40
1.3	1.80	2.60
1.4	1.90	2.80
1.5	2.00	3.00
1.6	2.10	3.10
1.7	2.20	3.20

Using Table 7, the nominal focal spot values are found using the widths and heights from Table 5.

Table 8 - Nominal Focal Spot Values consistent with measured dimensions.

Test	$W_{lin}(mm)$	$H_{lin}(mm)$	$f_{W_{lin}}$	$f_{H_{lin}}$	$W_{quad}(mm)$	$H_{quad}(mm)$	$f_{W_{quad}}$	$f_{H_{quad}}$
A	1.95 ± 0.24	2.58 ± 0.32	1.5	1.3	2.03 ± 0.23	2.67 ± 0.28	1.6	1.4
B	1.92 ± 0.24	2.61 ± 0.32	1.5	1.4	2.00 ± 0.22	2.69 ± 0.28	1.5	1.4
C	1.95 ± 0.24	2.56 ± 0.32	1.5	1.3	2.02 ± 0.22	2.63 ± 0.28	1.4	1.4
D	1.98 ± 0.26	2.74 ± 0.35	1.5	1.4	2.15 ± 0.23	2.92 ± 0.31	1.7	1.5
E	1.77 ± 0.23	2.46 ± 0.31	1.3	1.2	1.86 ± 0.21	2.54 ± 0.27	1.4	1.3
F	1.72 ± 0.22	2.26 ± 0.29	1.3	1.2	1.79 ± 0.20	2.33 ± 0.25	1.3	1.2
G	1.06 ± 0.15	1.66 ± 0.22	0.7	0.9	1.21 ± 0.14	1.82 ± 0.20	0.9	1.0
H	0.95 ± 0.13	1.50 ± 0.20	0.7	0.7	1.08 ± 0.12	1.63 ± 0.17	0.7	0.9
I	0.99 ± 0.14	1.65 ± 0.22	0.7	0.9	1.12 ± 0.13	1.79 ± 0.19	0.8	1.0
J	0.96 ± 0.13	1.75 ± 0.23	0.7	0.9	1.09 ± 0.12	1.88 ± 0.19	0.7	1.0

The averages of the linear and quadrature width and height measurements for both the large and small focal spots are presented in Table 9. Based on these averaged dimensions, a nominal focal spot value was assigned using the values from Table 7.

Table 9 - Averaged Large and Small Focal Spot Nominal Values.

Focal Spot	$W_{lin}(mm)$	$H_{lin}(mm)$	$f_{W_{lin}}$	$f_{H_{lin}}$	$W_{quad}(mm)$	$H_{quad}(mm)$	$f_{W_{quad}}$	$f_{H_{quad}}$
Large Average	1.88 ± 0.24	2.54 ± 0.32	1,4	1,3	1.97 ± 0.22	2.63 ± 0.28	1,5	1,4
Small Average	1.01 ± 0.14	1.65 ± 0.22	0,7	0,9	1.14 ± 0.13	1.79 ± 0.19	0,8	0,9

Finally, the linear and quadrature values were averaged together for both width and height, and large and small focal spots. The resulting corrected averages were then used to assign final nominal focal spot values using Table 7. These are shown in Table 10.

Table 10 - Corrected and averaged focal spot nominal values for both the large and small focal spots.

Focal Spot	$W (mm)$	$H (mm)$	f_W	f_H
Large Average	1.93 ± 0.26	2.58 ± 0.31	1.5	1.3
Small Average	1.07 ± 0.14	1.72 ± 0.21	0.7	0.9

Figure 22 illustrates how the measured focal spot height for a test point (*Test F*) is interpreted relative to the IEC's nominal focal spot values. The blue line represents the IEC's relationship between nominal focal spot values and their corresponding maximum permissible values, as outlined in Table 7. This relationship is nonlinear due to the uneven spacing of maximum permissible values.

Because the IEC provides no uncertainty for nominal values, plotting physical measurements directly against nominal values can be misleading. To resolve this, the measurement (black dot) and its uncertainty (magenta error bar) are projected onto the blue IEC curve. The corresponding y-value is the nominal focal spot value associated with the physical measurement. This effectively translates the physical measurement and its uncertainty (both only in the x-direction) into a nominal focal spot value with an implied uncertainty in the y-direction, where the uncertainty is represented by the shaded pink region. The dashed black lines denote the maximum permissible heights for nominal values 1.1 and 1.2, bracketing the zone in which the large focal spot is expected to lie (**Figure 14**). This enables a more visual comparison between measured data and IEC-defined tolerances.

Figures 23–27 apply the same method to all focal spot measurements, with darker shaded areas being where uncertainties overlap.

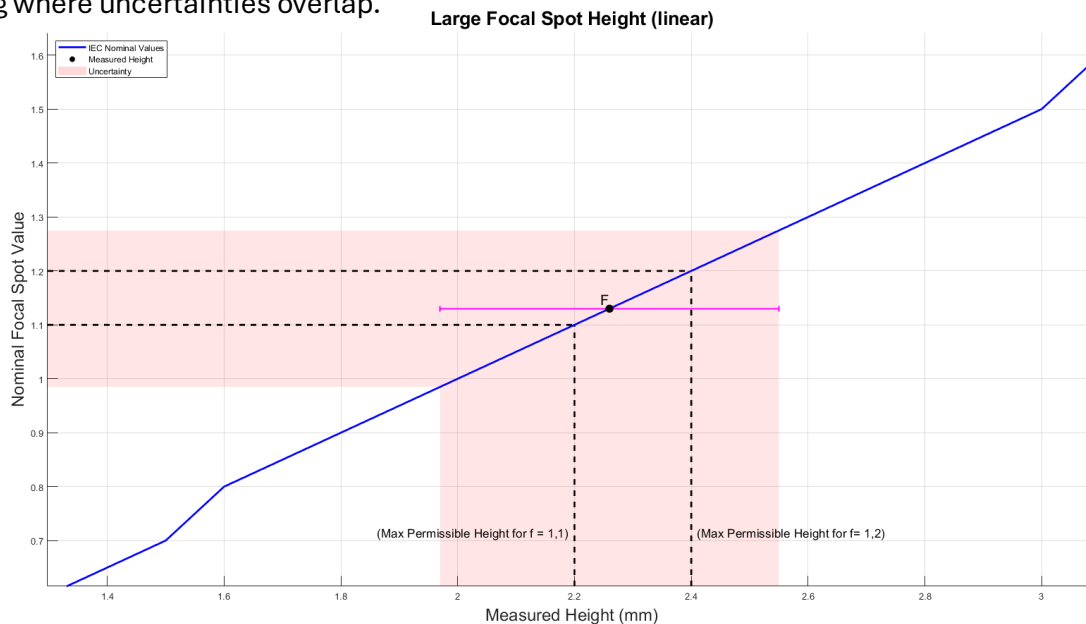


Figure 22 - Example graph of a measured point (*Test F*) corresponding to the IEC's nominal values. This is for the large focal spot's height, measured linearly.

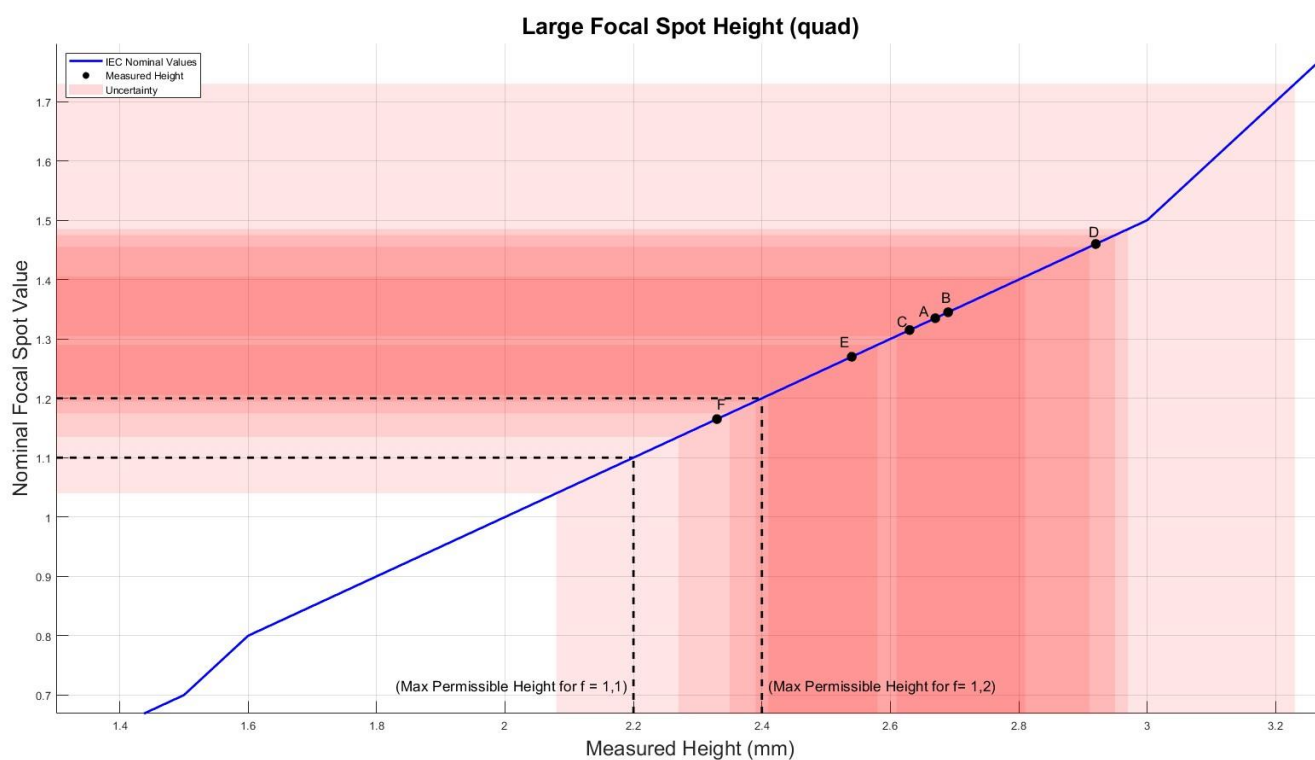
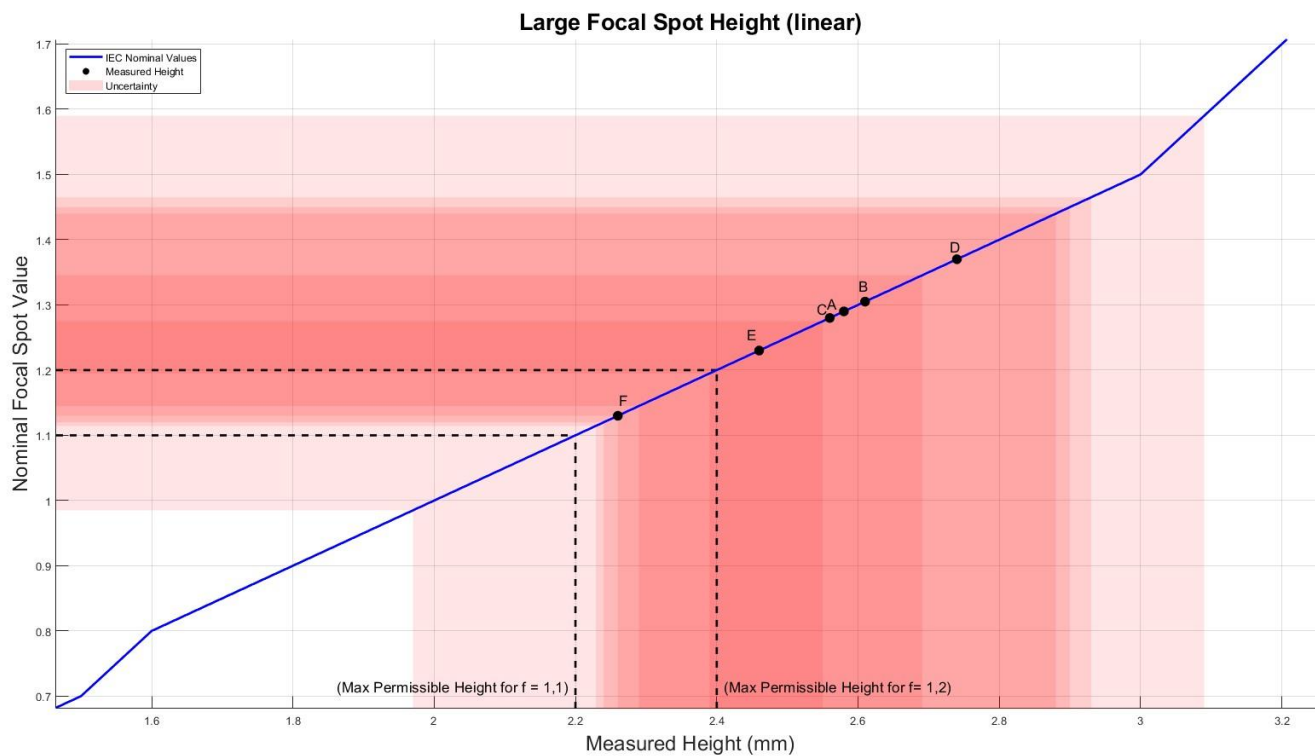


Figure 23 - Large Focal Spot Corrected Height Measurements.

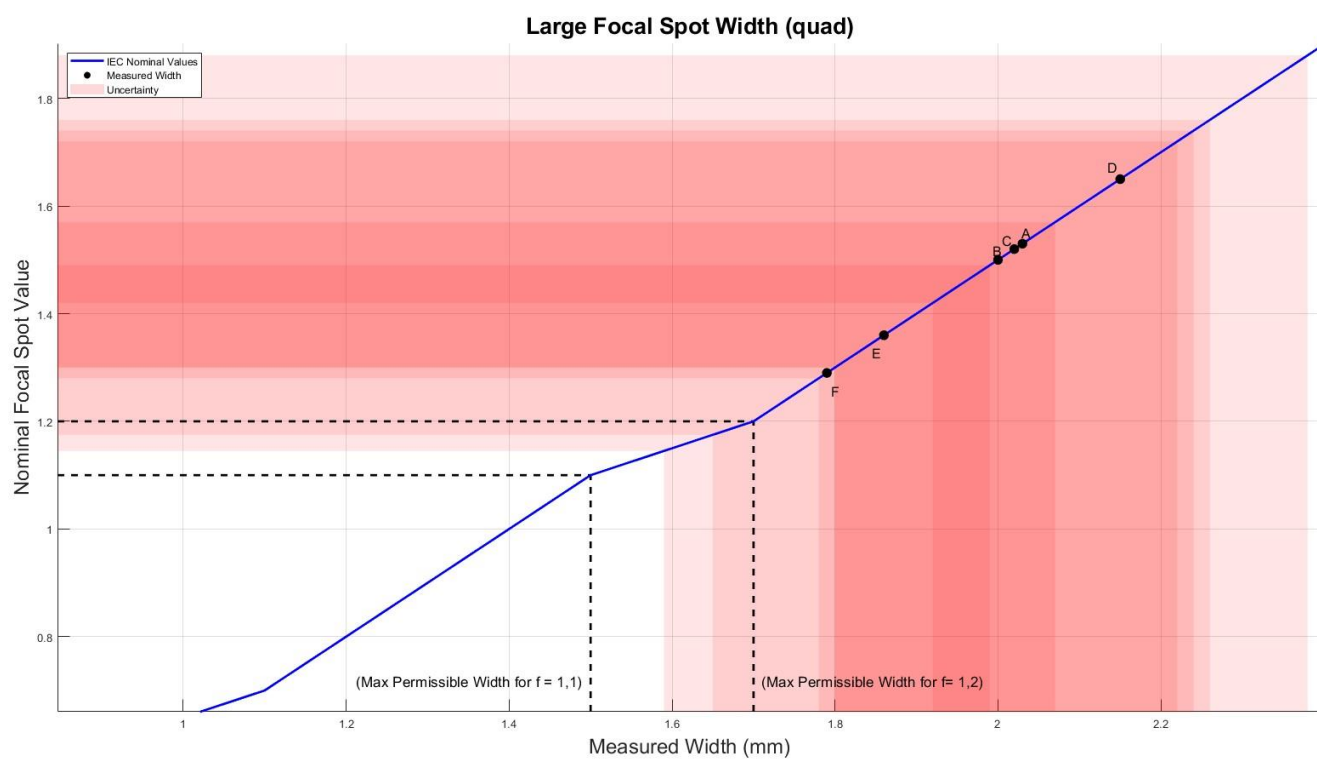
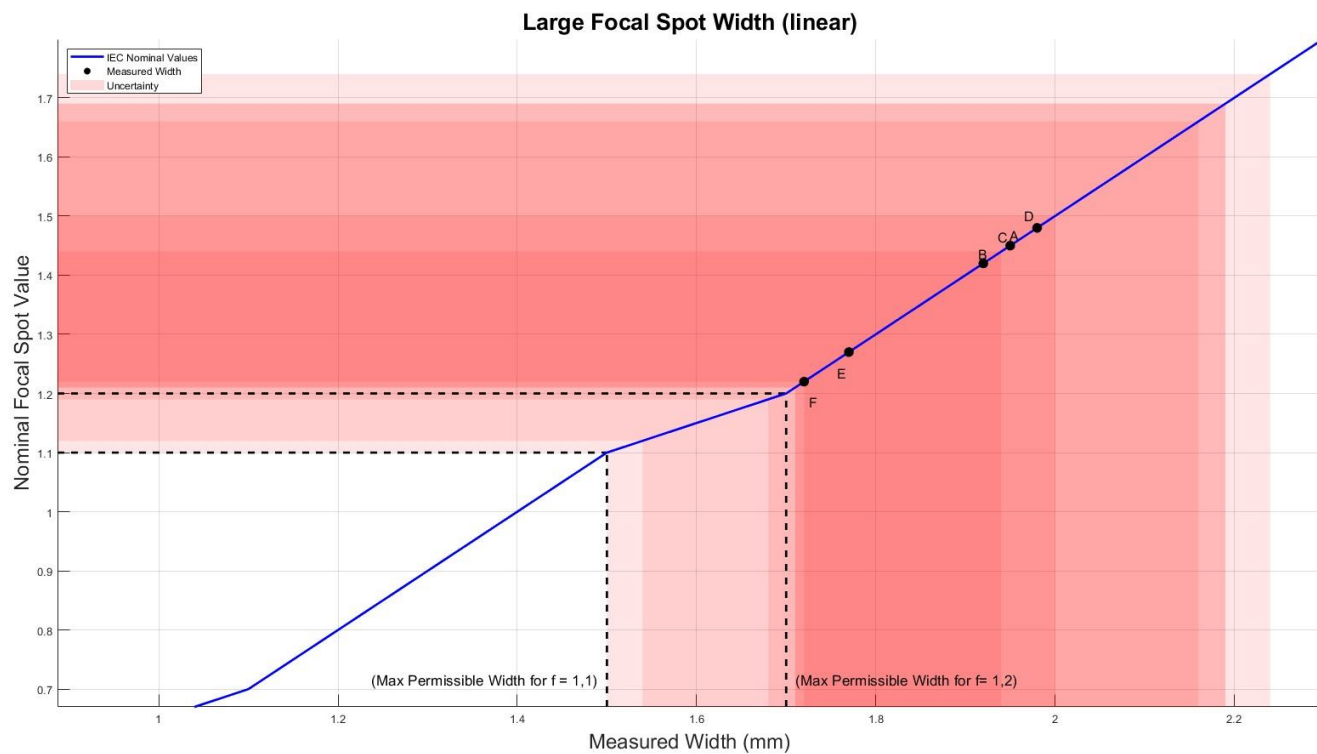


Figure 24 - Large Focal Spot Corrected Width Measurements.

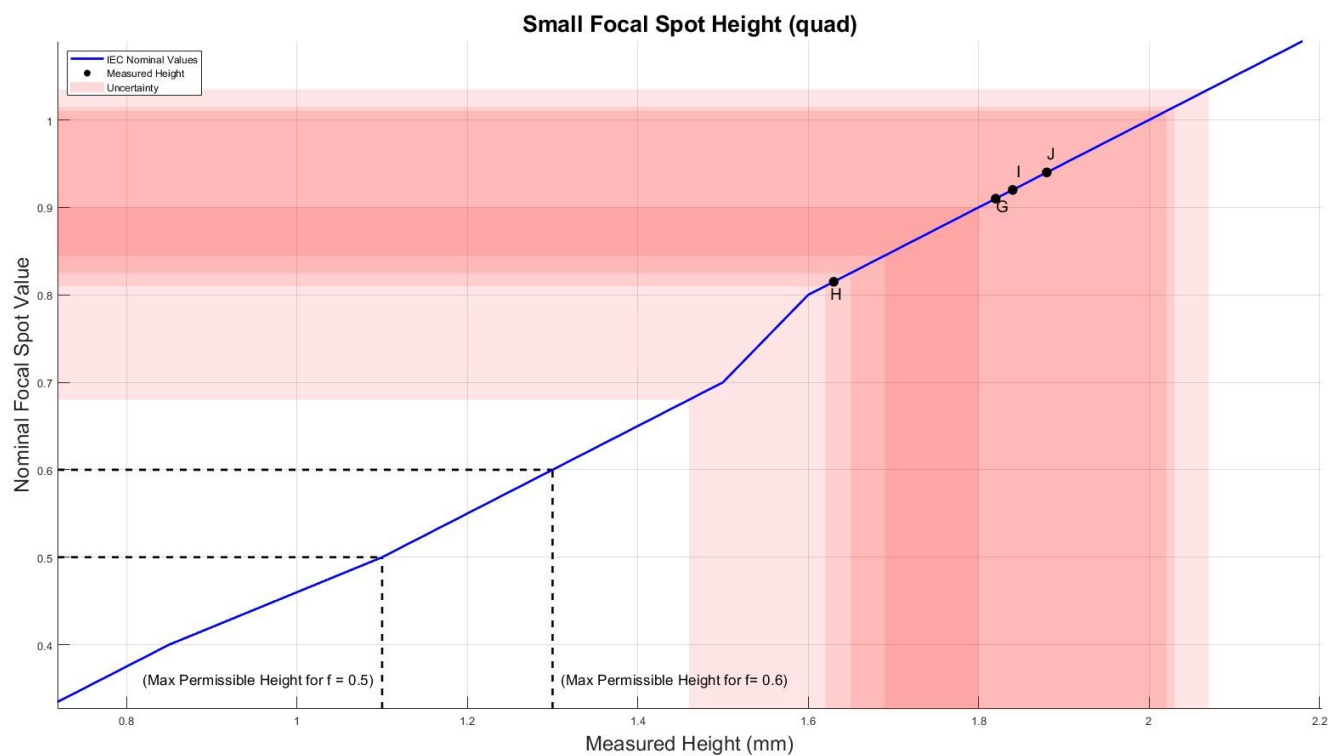
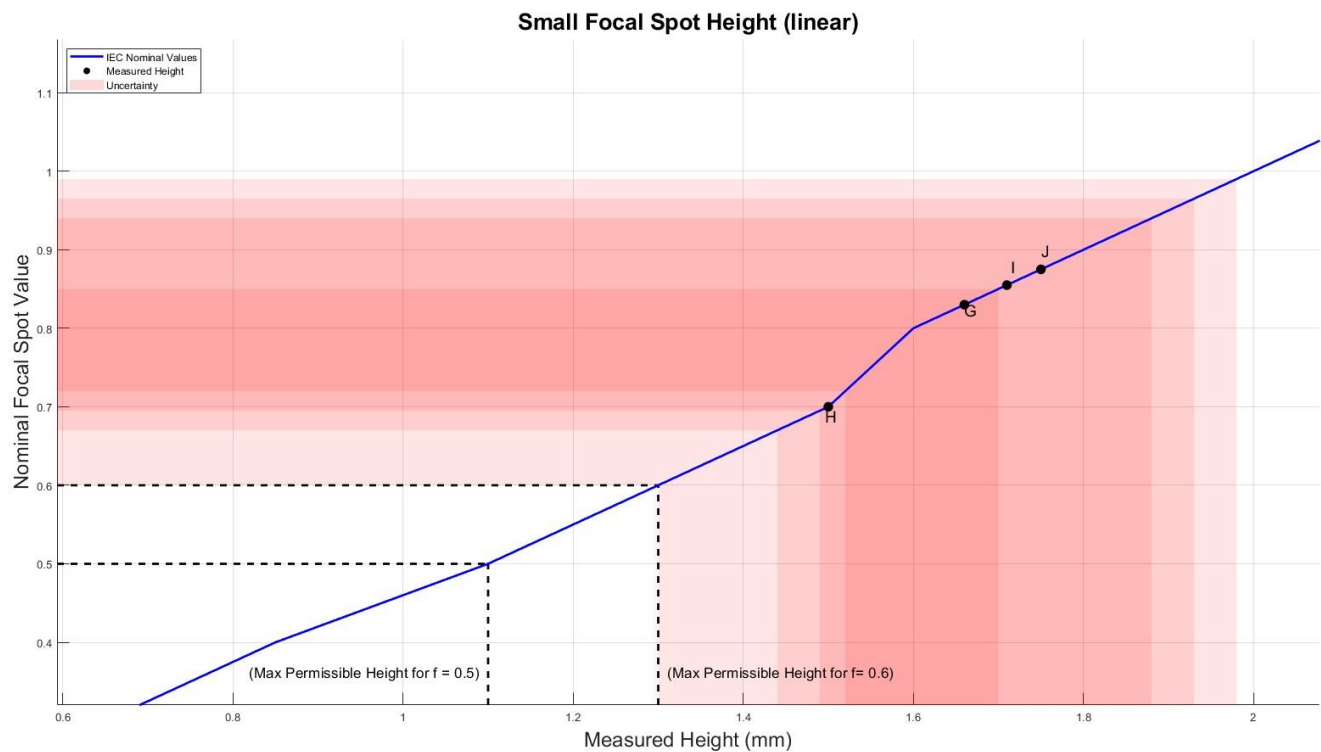


Figure 25 - Small Focal Spot Corrected Height Measurements.

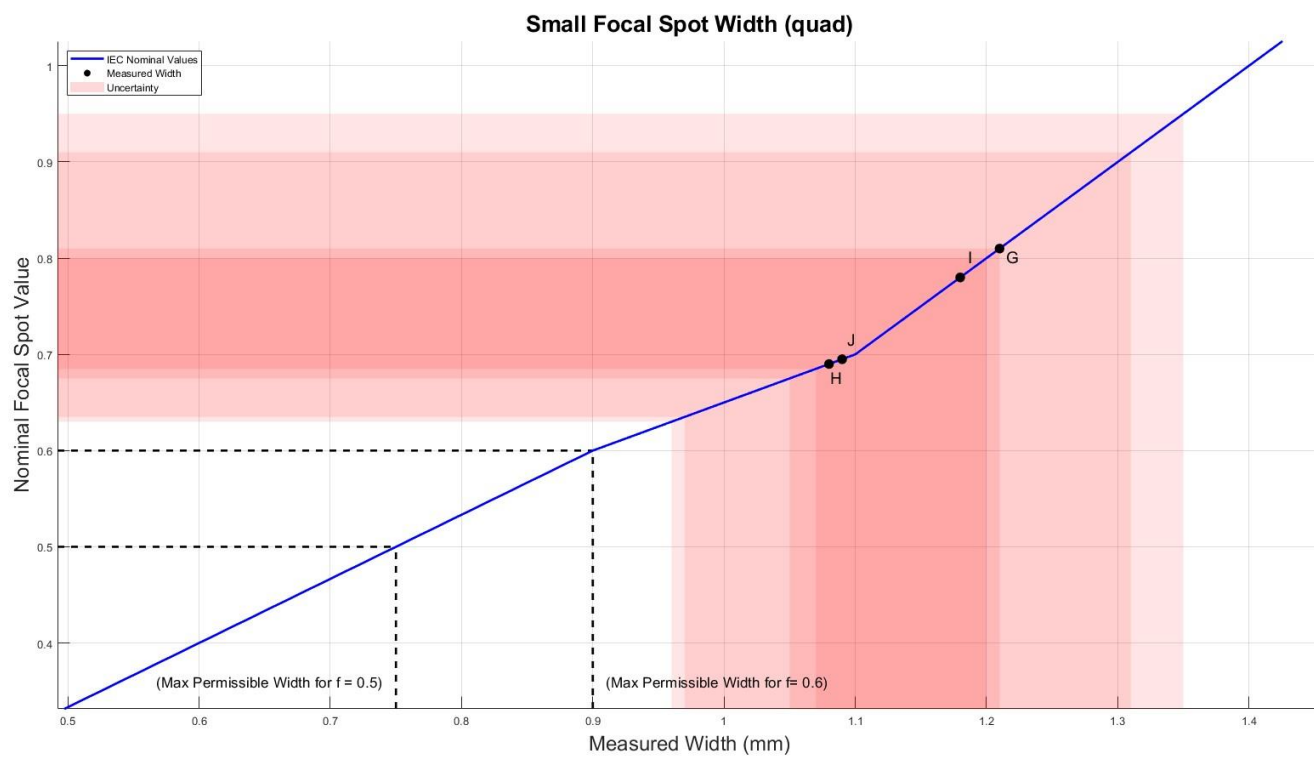
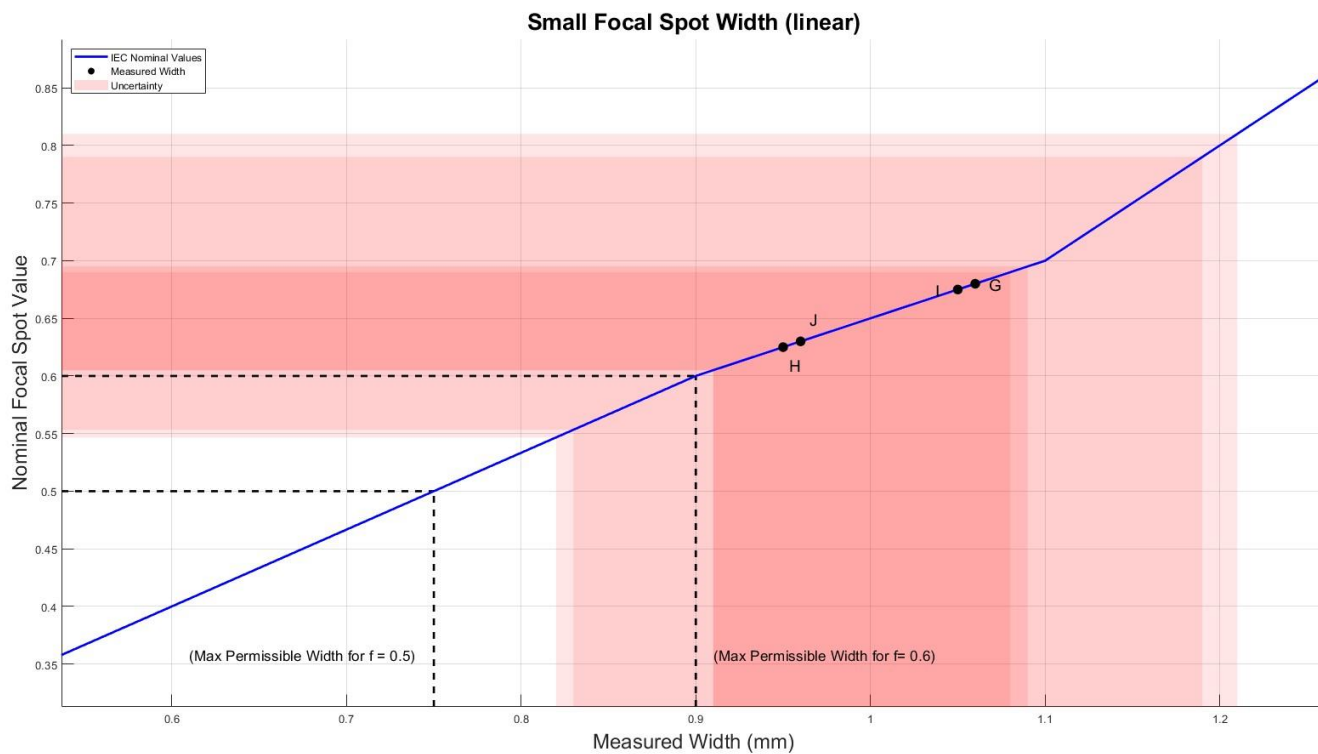


Figure 26 - Small Focal Spot Corrected Width Measurements.

5.0 Results and Discussion

5.1 Results

The Hamamatsu S11684-12 CMOS detector demonstrated sufficient resolution to measure the nominal focal spot size, achieving a resolution of $(18 \pm 1) \frac{lp}{mm}$, consistent with the manufacturer's specifications. The sharpest images were obtained when the line pairs were aligned vertically along the detector, as shown in **Figure 9**. In contrast, the lowest resolution was observed with the diagonal alignment. This was likely because the diagonal orientation projected the line pairs across both pixel dimensions (horizontal *and* vertical), effectively reducing the sampling density and making the line pairs less distinguishable from the pixel structure.

Equation 12 showed that total blur in the object plane is minimized by maximizing the enlargement. While direct measurement of blur was not performed, this theoretical prediction is supported qualitatively by the observed improvements in image sharpness and nominal measurement accuracy as enlargement increased, as shown in **Figure 21**, as well as **Figures 23 – 27**.

The matrix analysis found that both ATN and NTA were effective at normalizing fluctuations caused by the detector's row-sequential readout. Although NTA slightly outperformed ATN in limiting the propagation of fluctuations to neighboring pixels, the difference between the two methods was minimal. The key observation is that both methods significantly outperformed the manufacturer's flat field correction, as shown in **Figure 11**. In this experiment, only a subset of the 1500×1000 image was of interest, allowing specific regions to be allocated for row normalization. This is something that is not feasible with the FFC approach, which attempts to correct the entire image uniformly.

Table 11 – Comparing the effects of the fluctuations on the specific deviation point, as well as how it affects the row its in.

Method	Effect on Fluctuation $\Delta\gamma_{1,1}$	Effect on Row $\Delta\gamma_{1,c \neq 1}$
ATN	0.721	0.0529
NTA	0.724	0.0522

Of all variables tested, pinhole size, exposure time, and enlargement had the most significant impact on measurement precision. Increasing the enlargement consistently reduced image blur, while using a smaller pinhole and longer exposure time yielded the most accurate focal spot measurements.

The averaged measured nominal values for the large focal spot were $f_W = 1.5$ and $f_H = 1.3$. The averaged measured nominal values for the small focal spot were $f_W = 0.7$ and $f_H = 0.9$. These results deviate from the manufacturer's reported values of $f_{(large)} = 1.2$ and $f_{(small)} = 0.6$.

Based on the above findings, the most accurate measurements were likely those using the smallest pinhole, longest exposure time, and greatest enlargement. These were Test F for the large focal spot, and Tests I and J for the small focal spot:

Table 12 – Most accurate test conditions and corresponding parameters.

Test	Focal Spot	$(d_3 \pm 10)$ mm	$(d_2 \pm 10)$ mm	Enlargement (E)	Pinhole Size ± 0.005 mm	Tube Current (mA)	Exposure Time (ms)	Accelerating Potential (kVp)
F	Large	623	298	0.92 ± 0.040	0.030	200	200	60
I	Small	681	379	1.26 ± 0.059	0.075	50	200	60
J	Small	681	379	1.26 ± 0.059	0.075	100	200	60

Table 13 – Corresponding nominal values derived from Table 12 measurements.

Test	$W_{lin}(mm)$	$H_{lin}(mm)$	$f_{W_{lin}}$	$f_{H_{lin}}$	$W_{quad}(mm)$	$H_{quad}(mm)$	$f_{W_{quad}}$	$f_{H_{quad}}$
F	1.72 ± 0.22	2.26 ± 0.29	1.3	1.2	1.79 ± 0.20	2.33 ± 0.25	1.3	1.2
I	0.99 ± 0.14	1.65 ± 0.22	0.7	0.9	1.12 ± 0.13	1.79 ± 0.19	0.8	1.0
J	0.96 ± 0.13	1.75 ± 0.23	0.7	0.9	1.09 ± 0.12	1.88 ± 0.19	0.7	1.0

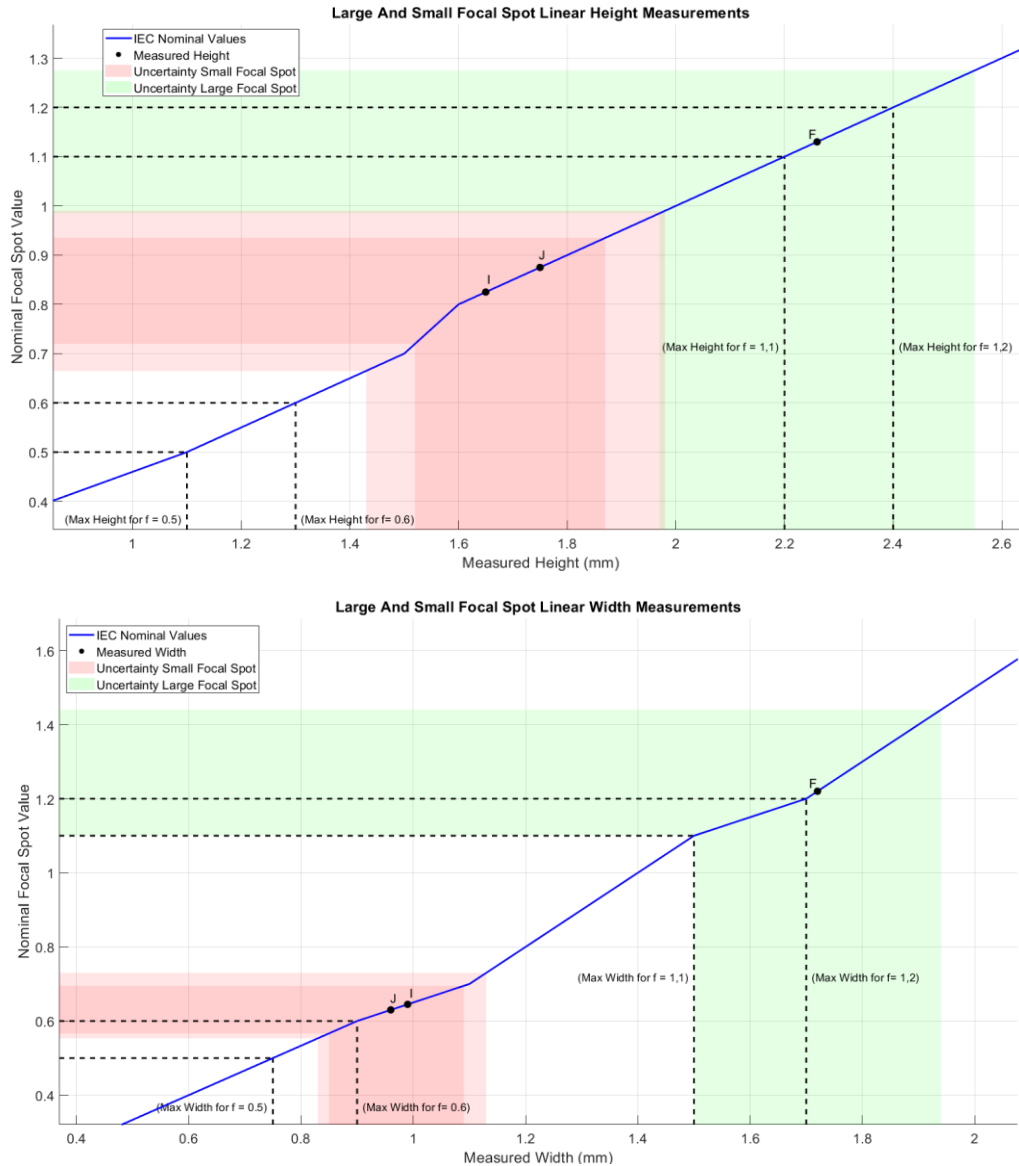


Figure 27 - Linear measurements corresponding to their nominal focal spot dimensions based on Table 13. The shaded regions show how measurement uncertainty propagated into nominal value uncertainty.

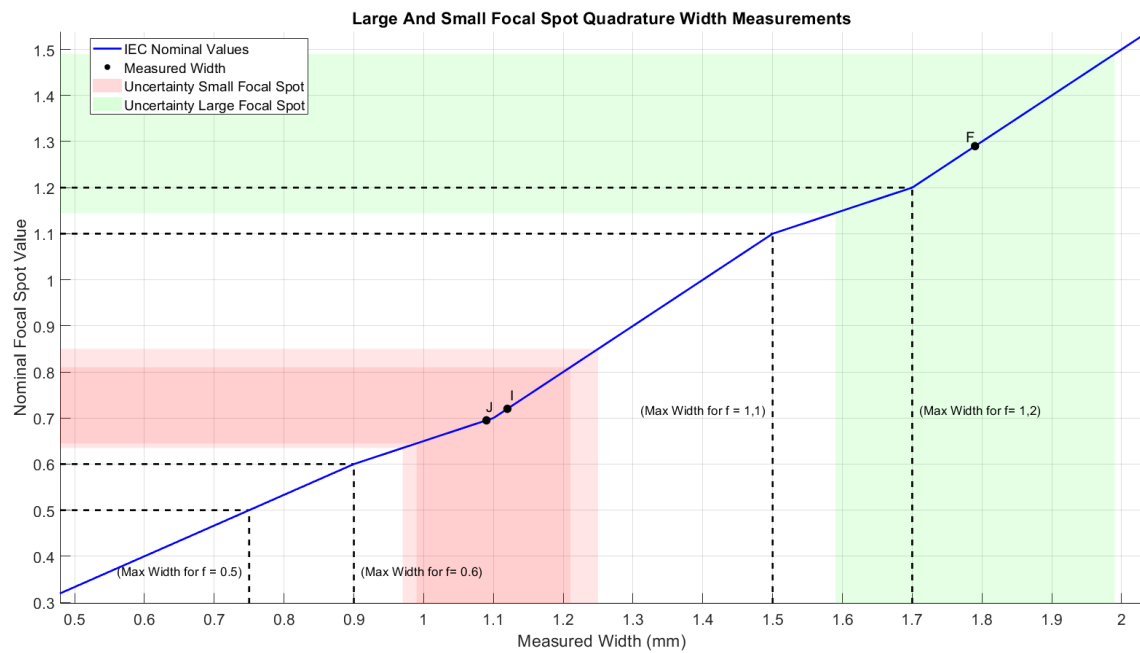
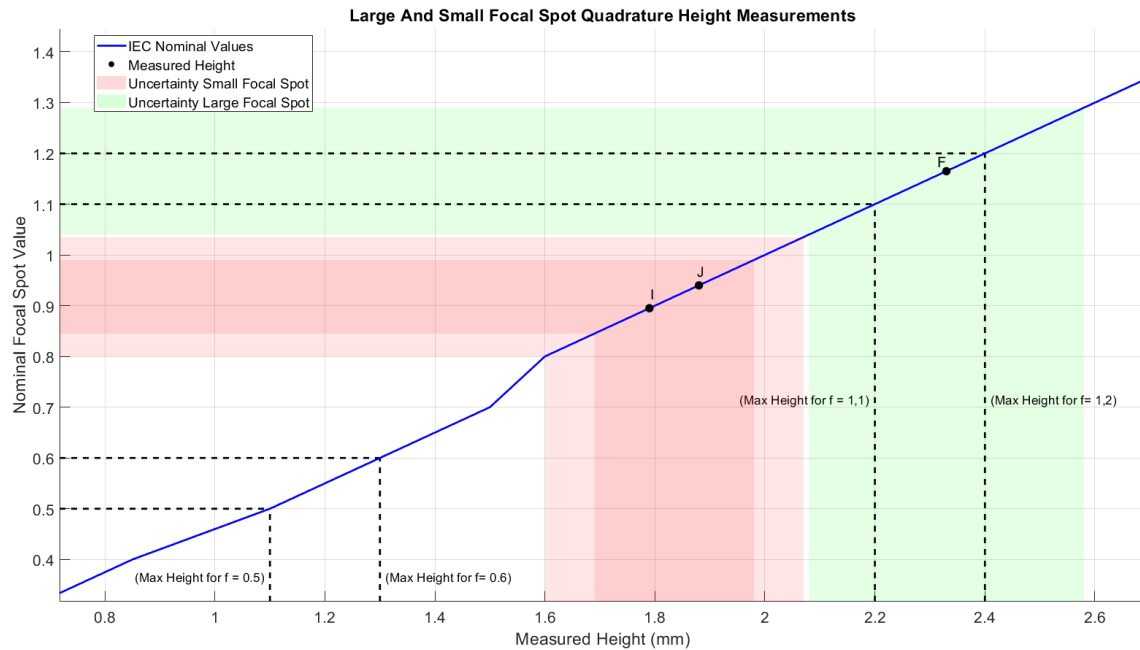


Figure 28 - Quadrature measurements used for the same purpose, with corresponding data from Table 13.

5.2 Discussion

The NTA (normalizing-then-averaging) method proved more effective than the ATN (averaging-then-normalizing) method in reducing row-specific noise caused by the detector's row-sequential readout. This reduction is crucial for focal spot measurements, as reliable pixel values are essential for accurate LSF projections. By normalizing each matrix before averaging, NTA minimizes structured noise and produces more consistent results. That said, both methods perform similarly overall and are significantly better than the manufacturer's FFC. A key advantage of NTA and ATN is that the normalization factors are calculated using selected regions on either side of the focal spot, extending from the top to the bottom of the image, ensuring accurate background correction without relying on the focal spot itself. This targeted approach allows for more precise normalization of row-dependent fluctuations, unlike the manufacturer's FFC, which applies a uniform correction across the entire image and does not account for localized image structure.

Despite this improvement, noise from the detector that is independent of row readout - such as electronic noise, fixed pattern noise, scattered radiation, and scintillator light spread - remains a challenge. These types of noise are intrinsic to the detector's design and operation and are difficult to eliminate entirely. The most effective way to mitigate such noise is through averaging over multiple images, as this reduces random fluctuations and enhances signal consistency.

Future work should explore the effect of processing larger numbers of raw, dark, and gain images after employing the NTA method to the Hamamatsu S11684-12 detector. Testing with varying numbers of averaged images (e.g., 10, 20, or 30) would help determine the number of images needed to achieve negligible background noise differences. This approach could potentially address the SNR issue, as obtaining a pinhole image with an SNR greater than 100 is challenging, if not impossible.

The custom detector case built for the Hamamatsu S11684-12 detector proved reliable and facilitated easier measurements. A combination of Jack Rubio's detector case (**Figure 13**), Eva Anderson's DVD stand (**Figure 15**), and Tong Xu's hollow pipe (**Figure 16**) allowed the detector to be aligned to within a small angle relative to the reference axis. However, the exact degree of alignment could not be confirmed, and potential misalignment of the x-ray tube might have introduced inaccuracies in some measurements.

Challenges in maintaining alignment were noted. Attaching the case to the bottom of the DVD stand enabled movement of the entire stand for alignment, rather than just adjusting the detector itself. Aligning the pinhole and detector gradually as d_2 changed was essential. Starting with the pinhole at $d_2 \cong 10\text{ cm}$, the image of the focal spot was centered on the detector, but at $d_2 \cong 20\text{ cm}$, the image shifted toward the edge. Around $d_2 \cong 30\text{ cm}$, the image no longer appeared on the detector. Repeated attempts to realign the pinhole with the reference axis revealed that achieving proper alignment required moving the detector after positioning the pinhole at greater distances. This suggests that the reference axis was not truly vertical. Future work should prioritize improved methods to identify and align with the reference axis to ensure proper x-ray beam geometry.

Two IEC requirements were consistently able to be met during testing:

- At least 60 pixels spanned the 15% width of the LSF.
- The pixel value range between the background and focal spot maximum was ≥ 200 .

These metrics validate measurement reliability and serve as benchmarks for future experiments. Determining a minimum d_2 based on enlargement **equations (3 and 4)** ensured both IEC criteria were satisfied.

While increasing enlargement reduces blur and improves measurement accuracy, excessive enlargement can cause signal loss, risking IEC violations. Identifying a practical upper bound for enlargement that balances precision and signal strength is a logical next step for improving nominal value measurements.

Smaller pinholes and longer exposure times also improved spatial measurement accuracy. Optimizing these parameters in tandem with enlargement boundaries will streamline testing and reduce guesswork.

In summary, this study highlights practical strategies for improving focal spot measurements when using the Hamamatsu S11684-12 CMOS detector, while also adhering to the IEC's standards. By refining alignment techniques, mitigating noise through averaging methods, and optimizing key parameters such as pinhole size and exposure time, future research can further enhance measurement accuracy and reproducibility.

5.3 Acknowledgements

I would like to thank Dr. Paul Johns, my supervising professor, for his consistent support and guidance throughout this project. With his help, I've learned how to navigate the research process and understand the underlying math and physics of x-ray imaging. Beyond academics, Dr. Johns also took the time to help me with my resume, which ultimately helped me land my first position in the field. His support speaks to his character and his dedication to his students. Thanks, Paul.

I also want to acknowledge the work of Eva Anderson and André Ramos Moreno; prior students whose contributions laid the groundwork for this project. Their efforts made many aspects of this research possible.

Special thanks to Jack Rubio for helping design and 3D print the detector case. You made my life a lot easier. I'd also like to thank Dr. Tong Xu for machining the hollow metal pipe that made the detector alignment process significantly smoother.

And finally, to my girlfriend Peyton Horning; thank you for being my biggest supporter through every challenge and triumph of this experience (especially when my laptop crashed after putting 20 hours into my midterm report). Your encouragement and positivity kept me going and reminded me why I care about this work. I wouldn't be half as driven or fulfilled without you by my side.

6.2 References

- [1] A. B. Wolbarst and G. Cook, *Physics of Radiology*. London: Medical Physics Publishing, 2012.
- [2] “Bremsstrahlung,” Wikipedia, <https://en.wikipedia.org/wiki/Bremsstrahlung> (accessed Mar. 17, 2025).
- [3] E. Anderson, *Evaluation of Miniature X-ray Imaging Detector and Its Application to Focal Spot Imaging*. Ottawa: Carleton University, Physics 4909 Honours Report, Apr. 2022.
- [4] E.T.E. Anderson and P.C. Johns, *Evaluation of miniature x-ray imaging detector and its application to focal spot assessment*, Poster # 2 at the 68th annual scientific meeting of the Canadian Organization of Medical Physicists, Quebec City (2022 June 22-25). [Abstract: *Medical Physics* 49 (8), 5645-5645 (2022), <https://dx.doi.org/10.1002/mp.15896>].
- [5] A. Ramos Moreno, *Design and Setup of an Assembly for Evaluating an X-ray Tube Focal Spot*. Ottawa: Mitacs Globalink internship report with Carleton University, Aug. 2024.
- [6] *Focal spot dimensions and related characteristics IEC 60336 ed. 5.0 B:2020*. International Electrotechnical Commission, Dec. 2020.
- [7] U. Themes, “1. plain radiographic imaging,” *Radiology Key*, <https://radiologykey.com/1-plain-radiographic-imaging-2/> (accessed Mar. 17, 2025).
- [8] Hamamatsu Photonics K.K., *CMOS Area Image Sensors for X-ray Imaging (USB Interface Type) S11684-12/-62*, Solid State Division, Hamamatsu City, Japan, Nov. 2016.
- [9] Hamamatsu Photonics K.K., *X-ray Detectors Technical Note*, Oct. 2023. [Online]. Available: https://www.hamamatsu.com/resources/pdf/ssd/x-ray_kmpd9016e.pdf [Accessed: Mar. 18, 2025].
- [10] F. A. A. Saoudi, “Comparative Study of Scintillator Materials for High Energy Imaging,” *Open Journal of Medical Imaging*, vol. 8, no. 2, pp. 29–37, 2018. [Online]. Available: <https://www.scirp.org/journal/paperinformation?paperid=82312>.
- [11] Wikipedia contributors, “Flat-field correction,” Wikipedia, *The Free Encyclopedia*. [Online]. Available: https://en.wikipedia.org/wiki/Flat-field_correction. [Accessed: Mar. 9, 2025].
- [12] Wikipedia contributors, “Hadamard product (matrices),” Wikipedia, *The Free Encyclopedia*. [Online]. Available: [https://en.wikipedia.org/wiki/Hadamard_product_\(matrices\)](https://en.wikipedia.org/wiki/Hadamard_product_(matrices)). [Accessed: Mar. 15, 2025].
- [13] RaySafe, “X-ray Pinhole Assemblies,” RaySafe, 2021. [Online]. Available: https://www.raysafe.com/sites/default/files/2021-07/g21222_pinhole_leaflet_eng_reva.pdf. [Accessed: Apr. 8, 2025].
- [14] Atlantic, “Onyx 28 DVDs/Blu-rays Tower,” Atlantic, 2021. [Online]. Available: <https://www.standsandmounts.com>. [Accessed: Apr. 8, 2025].

Appendix A

A.1 Nominal_Value_Finder.m

This script uses functions “length_average.m” (Appendix A.1.i) and “width_average.m” (Appendix A.1.ii).

Code Description

Focal Spot Measurement Script

This script is designed to measure the focal spot width and height using data from the Hamamatsu S11684-12 detector. It analyzes an image array captured in the detector plane, transforms the dimensions to the source plane, and accounts for optical blur. The script also evaluates signal quality metrics such as Signal-to-Noise Ratio (SNR) and calculates the number of unique signal levels.

Key Features:

1. Interactive selection of background and focal spot regions.
2. Calculation of focal spot dimensions in pixels and millimeters.
3. Optical blur correction for accurate measurements.
4. Signal-to-noise ratio computation and uncertainty estimation.
5. Visualization of line spread functions for width and height.

Inputs:

- A '.mat' file containing the image array from the Hamamatsu detector.
- User-defined optical parameters: d3, d2, pinhole size.

Outputs:

- Focal spot dimensions (pixels and millimeters).
- Blur-corrected dimensions.
- Signal-to-noise ratio and uncertainties.
- Number of unique signal levels.

Uses functions:

- length_average.m
- width_average.m

Author: Grant Budge

```
clc;  
clear;  
close all;
```

Select array and gather info from the array

```
% Prompts select a .mat file containing the image array data.  
disp("Select your array.");  
[file, path] = uigetfile('*.mat', 'Select the saved Adjusted array');
```

```

% Check if the user selected a file or canceled the operation.
if isequal(file, 0)
    disp('No file selected. Exiting script.');
```

return; % Exit the script if no file is selected.

```
else
    % Load the selected .mat file and extract its contents.
    loadedData = load(fullfile(path, file));
    varNames = fieldnames(loadedData); % Get variable names in the .mat file.

    % Check if the .mat file contains any variables, just for error
    % handling
    if isempty(varNames)
        error('The selected .mat file contains no variables.');
```

end

% Extract the first variable in the .mat file as the image array.

```
    image_array = loadedData.(varNames{1});
end

% Display the name of the selected file for confirmation, this lets the
% user see what variables they used. Users should name their array
% something like "d3_661_d2_298_pinholesize_30_exposuretime_100ms_..."
disp(['The name of your file is: ', file]);

% Prompts to input optical parameters (d3, d2, pinhole size).
while true
    d3 = input("Enter your d3 value in millimeters: ");
    d2 = input("Enter your d2 value in millimeters: ");
    pinhole_size = input("Enter your pinhole in micrometers: ");
    pinhole_size = pinhole_size/1000; % Convert micrometers to millimeters.

    % Define detector maximum blur (fixed value based on system calibration).
    detector_max_blur = 0.0277777; % in millimeters

    % Ensure that d3 is greater than d2; otherwise, prompt again.
    if d3 > d2
        break;
    else
        disp('Error: d3 must be greater than d2. Please try again.');
```

end

```

end

% Calculate the enlargement factor based on optical geometry:
Enlargement = d2 / (d3 - d2);
disp(['Enlargement factor: ', num2str(Enlargement)]);

% Find the maximum intensity value in the image array for thresholding later.
max_intensity = max(image_array(:));

```

Select background

```

% Prompts to select background region. Select the biggest region possible
% for higher accuracy.
while true
    figure, imshow(image_array, []);
    title('Select a region for the background (click and drag)');
    bg_selection = drawrectangle;
    wait(bg_selection);
    position = bg_selection.Position;
    background_region = image_array(round(position(2)):(round(position(2)) +
round(position(4))), round(position(1)):(round(position(1)) + round(position(3))));
    background_mean = mean(background_region(:));
    disp(['Background mean intensity: ', num2str(background_mean)]);
    if strcmp(questdlg('Are you satisfied with this selection?', 'Confirm Selection',
'Yes', 'No', 'Yes'), 'Yes')
        break;
    else
        close;
    end
end
end

```

Select focal spot region

Prompts to select the focal spot region. Make sure to select as little background as possible for highest accuracy. Include the entire focal spot in selection.

```

close all;
while true
    figure, imshow(image_array, []);
    title('Select a region around the focal spot');
    focal_region = drawrectangle;

```



```

    position = focal_region.Position;
    x_start = round(position(1));
    y_start = round(position(2));
    width_region = round(position(3));
    height_region = round(position(4));
    break;
end
% focal spot full region
focal_spot_region = image_array(y_start:(y_start + height_region - 1),
x_start:(x_start + width_region - 1));

```

Find the width and length

```

% - width_array: Vertical profile
% - height_array: Horizontal profile (yes, height seems like a backwards
%                               term when doing it this way)
width_array = width_average(image_array, x_start, y_start, width_region,
height_region);
height_array = length_average(image_array, x_start, y_start, width_region,
height_region);

threshold_intensity = background_mean + 0.15*(max_intensity - background_mean);
disp(['Threshold intensity (15% max): ', num2str(threshold_intensity)]);

```

Calculate the number of consecutive values > threshold.

```

% Function to find the longest sequence of consecutive values above a given threshold
function [max_consecutive_count, start_index, end_index] =
find_max_consecutive_above_threshold(array, threshold)
    count = 0; % Counter for consecutive values above threshold
    max_count = 0; % Maximum count of consecutive values
    start_idx = 0; % Temporary start index for current sequence
    max_start_idx = 0; % Start index of the longest sequence
    max_end_idx = 0; % End index of the longest sequence

    % Loop through the array to check each value against the threshold
    for i = 1:length(array)
        if array(i) > threshold
            if count == 0

```

```

        start_idx = i; % Start a new sequence
    end
    count = count + 1; % Increment count for consecutive values

    % Update maximum sequence if current sequence is longer
    if count > max_count
        max_count = count;
        max_start_idx = start_idx;
        max_end_idx = i;
    end
else
    count = 0; % Reset counter if value drops below threshold
    start_idx = 0; % Reset starting point
end
end

% Return the maximum count and indices of the longest sequence
max_consecutive_count = max_count;
start_index = max_start_idx;
end_index = max_end_idx;
end

% Find the maximum consecutive counts above the threshold for width and height arrays
Pixel_width = find_max_consecutive_above_threshold(width_array, threshold_intensity);
Pixel_height = find_max_consecutive_above_threshold(height_array,
threshold_intensity);

% Display the focal spot dimensions in pixels based on the calculated sequences
disp(['The focal spot is: ', num2str(Pixel_width), ' pixels wide']);
disp(['The focal spot is: ', num2str(Pixel_height), ' pixels long']);

% clean up workspace, you may have multiple images open from the selection,
% so we'll close those to save some space
close all;

```

Find intersection points for width and height

```

% Find the indices where the intensity profile crosses the threshold

```

```

width_crossings = find(width_array > threshold_intensity); % Indices where width
profile is above 15% max
height_crossings = find(height_array > threshold_intensity); % Indices where height
profile is above 15% max

% Extract the first and last crossing points for the width profile
width_start = width_crossings(1);
width_end = width_crossings(end);

% Extract the first and last crossing points for the height profile
height_start = height_crossings(1);
height_end = height_crossings(end);

```

Graphing the line spread function for width

Create a plot of the intensity profile for width

```

figure;
plot(width_array,"LineWidth",1.5);
title('Line Spread Function - Width');
xlabel('Pixels');
ylabel('Intensity');
hold on;

% Add a horizontal line at the threshold intensity
plot([1, length(width_array)], [threshold_intensity, threshold_intensity], '--
k',"LineWidth",1.5); %threshold line

% Mark the start and end points of the region above threshold with vertical lines
xline(width_start, '--r', 'Start',"LineWidth",1.5); % vertical
xline(width_end, '--r', 'End',"LineWidth",1.5); % vertical
legend('Intensity Profile', '15% max', 'Start', 'End');

```

Graphing the line spread function for height

Create a plot of the intensity profile for height

```

figure;
plot(height_array,"LineWidth",1.5);
title('Line Spread Function - Height');
xlabel('Pixels');
ylabel('Intensity');
hold on;

```

```

% Add a horizontal line at the threshold intensity
plot([1, length(height_array)], [threshold_intensity, threshold_intensity], '--
k',"LineWidth",1.5); % threshold line

% Mark the start and end points of the region above threshold with vertical lines
xline(height_start, '--r', 'Start',"LineWidth",1.5); % vertical
xline(height_end, '--r', 'End',"LineWidth",1.5); % vertical
legend('Intensity Profile', '15% max', 'Start', 'End');

```

Change from pixels to distance

and take into account enlargement

```

% pixel size in millimeters
pixel_size = 0.02;

% Convert focal spot dimensions from pixels to millimeters, accounting for
enlargement
focal_spot_width_mm = (Pixel_width * pixel_size) / Enlargement;
focal_spot_height_mm = (Pixel_height * pixel_size)/ Enlargement;

% Display the focal spot dimensions in millimeters
disp(['The focal spot width is: ', num2str(focal_spot_width_mm), ' mm']);
disp(['The focal spot height is: ', num2str(focal_spot_height_mm), ' mm']);

```

Account for blur Width and Height

```

% Calculate the total blur using detector blur and pinhole size
Blur = sqrt(detector_max_blur.^2 + pinhole_size.^2 * (Enlargement + 1).^2) /
Enlargement;

% Linear Corrections
quad_no_blur_width = (focal_spot_width_mm / Enlargement) - Blur;
quad_no_blur_height = (focal_spot_height_mm / Enlargement) - Blur;

% Quadrature Corrections
% no_blur_height = sqrt((focal_spot_width_mm / E)^2 - B_tot^2);
% no_blur_height = sqrt((focal_spot_height_mm/ E)^2 - B_tot^2);

```

```

disp(['The focal spot width after accounting for blur linearly is: ',
num2str(no_blur_width), ' mm']);
disp(['The focal spot height after accounting for blur linearly is: ',
num2str(no_blur_height), ' mm']);
disp(['The focal spot width after accounting for blur quadrature is: ',
num2str(quad_no_blur_width), ' mm']);
disp(['The focal spot height after accounting for blur quadrature is: ',
num2str(quad_no_blur_height), ' mm']);

```

Signal to noise ratio

```

% Calculate the mean signal intensity within the focal spot region
signal = mean(focal_spot_region(:));

% Calculate the standard deviation of the background region (noise)
noise = std(background_region(:),1,"all");

% Calculate the standard deviation of the focal spot region
fs_std = std(focal_spot_region(:),1,"all");

% Compute the signal-to-noise ratio (SNR)
SNR = signal./noise;

% Calculate uncertainties using standard error formulas
n= numel(focal_spot_region) % Number of pixels in the focal spot region
fs_error = fs_std / sqrt(n); % Uncertainty in focal spot mean
m = numel(background_region) % Number of pixels in the background region
BG_std_error = noise / sqrt(m); % Uncertainty in background noise
disp(['The signal to noise ratio is: ', num2str(SNR)]);
disp(['The focal spot mean is ', num2str(signal)]);
disp(['The uncertainty on Focal Spot Mean is ', num2str(fs_error)]);
disp(['The background std is ', num2str(noise)]);
disp(['The uncertainty on Background standard deviation is ',
num2str(BG_std_error)]);

```

A.1.i length_average.m

```
function row_average = length_average(image_array, x, y, width, height)
    % region of interest is the focal spot rectangle
    roi = image_array(y:y+height-1, x:x+width-1);

    % Compute the mean along each column to get a row
    row_average = mean(roi,1);
end
```

Published with MATLAB® R2024b

A.1.ii width_average.m

```
function column_average = width_average(image_array, x, y, width, height)
    % region of interest is the focal spot rectangle
    roi = image_array(y:y+height-1, x:x+width-1);

    % Compute the mean along each row to get a column vector
    column_average = mean(roi, 2);
end
```

Published with MATLAB® R2024b

A.2 GB_FFC.m

```
function FFC = GB_FFC()
```

Code Description

This script performs flat-field correction (FFC) and row normalization for a set of images taken by the Hamamatsu S11684-12 detector (Dark, X-ray, and Gain) by following these steps:

1. Prompts the user to select and average Dark, X-ray, and Gain images.
2. Minimizes row-sequential readout noise in the averaged images. (ATN method)
3. Performs calculations to subtract the Dark image from the X-ray and Gain images, applies weighted averaging, and computes the FFC array.
4. Adjusts the FFC array by adding a scaled standard deviation to make most negative values positive.
5. Displays the standard deviation of the FFC array and provides an option to save the result as a '.mat' file.

Uses functions:

- GB_AVG_images(): Function to average selected images. (**Appendix A.2.i**)
- GB_minimize_noise(): ATN method for row-noise in images. (**Appendix B**)
- GB_calculate_weighted_average_image(): Function to compute a weighted average of an image array. (**Appendix A.2.ii**)

Author: Grant Budge

```
%-----
```

Section 1

```
%-----

%
% AVERAGE DARK AD
%
fprintf('SELECT ALL DARK IMAGES.\n');
pause(1.5);
% Average all your Dark images by calling the averaging function.
try
    AD = GB_AVG_images();
    disp('Averaging Dark Images...');
catch ME
    disp('You have not selected any images.');
```

```
    return;
end
if ~exist('AD','var') || isempty(AD)
```

```

        disp('You either cancelled your selection or didnt select valid images.')
        pause(1);
        disp("Ending Script.")
        return;
    end
    %
    % AVERAGE XRAY AX
    %
    fprintf('SELECT ALL XRAY IMAGES.\n');
    pause(1.5);
    try
        % Average all of your xray images by calling the averaging function.
        AX = GB_AVG_images();
        disp('Averaging XRAY Images...');
        pause(1);
    catch ME
        disp('You have not selected any images.');
```

```

        return;
    end
    if ~exist('AX','var') || isempty(AX)
        disp('You either cancelled your selection or didnt select valid images.')
        pause(1);
        disp("Ending Script.")
        return;
    end
    %
    % AVERAGE GAIN AG
    %
    fprintf('SELECT ALL GAIN IMAGES.\n');
    pause(1.5);
    try
        % Average all of your Gain images by calling the averaging function.
        AG = GB_AVG_images();
        disp('Averaging Gain Images...');
        pause(1);
    catch ME
        disp('You have not selected any images.');
```

```

        return;
    end
end

```



```

if ~exist('AG','var') || isempty(AG)
    disp('You either cancelled your selection or didnt select valid images.')
    pause(1);
    disp("Ending Script.")
    return;
end
%
%-----

```

SELECT ALL DARK IMAGES.

You have not selected any images.

Section 2

```

%-----

AX = GB_minimize_noise(AX,1);
AD = GB_minimize_noise(AD,1);
AG = GB_minimize_noise(AG,1);
%
%
%-----

```

Section 3

```

%-----
%
%
% AX - AD which will give AS, Average Subtracted
%
AS = AX-AD;
%
%
% AG - AD which will give AGC, Average Gain Corrected
%
AGC = AG-AD;
%
%
% Call the weighted average of the AGC = ACGA
%
ACGA = GB_calculate_weighted_average_image(AGC);

```

```

%
%
% AS * ACGA = ASACGA (no real name for this, just multiplying the entire
% avergae subtraction array by the weighted average gain value)
%
ASACGA = AS .* ACGA;
%
%
% ASACGA / AGC = Flat Field Corrected FFC
FFC = ASACGA ./ AGC;

%-----

```

Section 4

```

%-----
%
% Find the standard deviation
std_ffc = std(FFC(:), 'omitnan');
disp('The standard deviation is:');
disp(std_ffc);
% Make most of the negative numbers positive.
fprintf('You will be adding %.3f to your array.', 3*std_ffc);
FFC = FFC + 4.*std_ffc;
%
% we can save the result
[file, path] = uiputfile('*.mat', 'Save the final array');
if isequal(file, 0)
    disp('You decided not to save the final array.');
```

```

else
    save(fullfile(path, file), 'FFC'); % Save the FFC array in .mat format
    disp(['You saved as: ', fullfile(path, file)]);
end

```

A.2.i GB_AVG_images

Code Description

GB_AVG_images - Image Averaging Function

This function calculates the pixel-wise average of multiple 16-bit TIFF images while excluding invalid pixels in triangular regions and edge rows. Designed specifically for flat-field correction in X-ray imaging systems using Hamamatsu S11684-12 detector.

Key Features:

1. Interactive multi-file selection of 16-bit TIFF images
2. Automatic exclusion of: - Top/bottom 3 rows - Triangular regions at image corners
3. NaN handling for robust averaging with partial valid data
4. Validation of image bit-depth and file integrity
5. Progressive summation to handle large datasets

Inputs:

- User-selected 16-bit TIFF files via dialog box

Outputs:

- AVG_array: Double-precision array of averaged pixel values

Called By:

- GB_FFC.m (Flat Field Correction routine)

Dependencies:

- Image Processing Toolbox
- MATLAB R2024a or newer

Author: Grant Budge

```
function [AVG_array] = GB_AVG_images()

    AVG_array = [];
    % Using 'uigetfile' to get multiple images.
    [files, path] = uigetfile({'*.tif;*.tiff', 'ONLY 16-bit tif files (*.tif, *.tiff)'}, ...
                              'Select Images', 'MultiSelect', 'on');

    % If you cancel your selection, end the script
    if isequal(files, 0)
        disp('No images selected. ');
        return;
    end

    % if only one image is selected, we have to convert the 'files' to a
    % cell array
    if ischar(files)
```

```

        files = {files};
    end

    % we need to store and count each image selected.
    counter = 0;
    image_sum = [];
    valid_pixel_count = [];

    % Process each of the selected image
    for i = 1:length(files)
        try
            % Read the selected image i
            image = imread(fullfile(path, files{i}));
        catch ME
            warning('Could not read image: %. Error: %s', files{i}, ME.message);
            continue;
        end

        % Make sure that each image is 16 bits.
        if ~isa(image, 'uint16')
            warning('The selected image: \n %s \n is not 16-bits, it will be
skipped.', files{i});
            continue;
        end

        % Convert the image (i) to a double for the calculation process
        image_array = double(image);

        % Add together the images using rows, columns, and channels
        % This will create a dummy variable, image_sum
        % All images will be added to the last image sum
        % The final image_sum will be taken for averaging
        if counter == 0
            [rows, columns, channels] = size(image_array);
            image_sum = zeros(rows, columns, channels);
            valid_pixel_count = zeros(rows, columns, channels);
        end

        % create a mask that discounts the unwanted pixels. This will turn
        % the top triangles into NaN and will also take care of the top 3

```

```

% and bottom 3 rows
mask = ones(rows,columns);
for r = 1:117
    mask(r, 1:(117-r+1)) = 0; % Top left triangle
    mask(r,columns - (117-r):columns) = 0; % top right triangle
end
mask(1:3,:)=0; % top three rows
mask(end-2:end,:) = 0; %bottom three rows

% apply the mask to the image
image_array(mask == 0) = NaN;

%now we can count valid pixels and find sums properly
image_sum = image_sum + image_array;
valid_pixel_count = valid_pixel_count + ~isnan(image_array);
counter = counter + 1;

end

% If the images have been added together, calculate the average
if counter > 0
    %avoid division by zero
    valid_pixel_count(valid_pixel_count ==0)=NaN;
    AVG_array = image_sum ./ valid_pixel_count;

    % We return an array to GB_FFC.m
else
    disp('No valid images were selected.');
```

AVG_array = [];

```

end
end
```

A.2.ii GB_calculate_weighted_average_image

Code Description

GB_calculate_weighted_average_image - Weighted Pixel Averaging Function

This function computes a weighted average of valid pixels in a 2D array while excluding specific detector regions. Designed for Hamamatsu S11684-12 CMOS detector data processing in flat-field correction workflows.

Key Features:

1. Hardware-specific masking of: - Top/bottom 3 rows - Triangular corner regions (rows 1-117)
2. Column-wise NaN handling for robust averaging
3. Computes global weighted average across all valid pixels
4. Generates uniform output array with average value

Inputs: - ACG_array: 2D array of corrected gain values (typically from AG-AD calculation)

Outputs: - ACGA: Uniform array filled with computed weighted average value

Called By: - GB_FFC.m (Flat Field Correction routine)

Dependencies:

- MATLAB base functions
- Requires input array from dark-corrected gain images

Author: Grant Budge

```
function ACGA = GB_calculate_weighted_average_image(ACG_array)

% Read in the array
image1 = ACG_array;

% Create a binary mask for the triangular regions to be excluded.
% Mask is specific to S11684-12 CMOS detector by Hamamatsu.
[rows, cols] = size(image1);
mask = ones(rows, cols);

% Define the triangular areas to exclude (top-left and top-right)
for i = 1:117
    mask(i, 1:(117-i+1)) = 0; % Top-left triangle
    mask(i, cols-(117-i):cols) = 0; % Top-right triangle
end
for i = 1:3
    mask(i, 1:cols) = 0; % Top 3 rows
    mask(rows-i+1, 1:cols) = 0; % Bottom 3 rows
end

% Apply the mask to set excluded pixels to NaN, for the sake of having
% consistent values to ignore. (everything ignored is NaN)
```

```

image1(mask == 0) = NaN;

% Initialize some variables for a weighted average
total_sum = 0;
total_valid_count = 0;

% Go through each column to calculate the weighted averages
for i = 1:cols
    col_values = image1(:, i);

    % Remove all the NaN values from the column
    valid_values = col_values(~isnan(col_values));
    valid_count = length(valid_values);

    % Compute the average.
    if valid_count > 0
        total_sum = total_sum + sum(valid_values); % sums ONLY the valid
pixel values
        total_valid_count = total_valid_count + valid_count; % keeps track of
how many pixels are valid.
    end
end

% Calculate the weighted average of the array.
if total_valid_count > 0
    weighted_average_value = total_sum ./ total_valid_count;
else
    weighted_average_value = NaN;
end

% Create a new image of the same size with the average value
% ACGA = Average Corrected Gain Average
ACGA = ones(size(image1)) .* weighted_average_value;

```

A.3 GB_image_rotatation.m

Code Description

Focal Spot Image Processing Script

This script processes flat-field corrected (FFC) X-ray focal spot images for analysis and measurement. Provides interactive tools for alignment and preparation of focal spot images for dimensional measurements.

Key Features:

1. Flexible input: Load existing FFC array or generate new correction
2. Interactive contrast adjustment for optimal visualization
3. Real-time image rotation alignment via user-drawn line
4. Manual cropping interface for region-of-interest selection
5. Dual-format saving (.mat and .tif) for processed images

Inputs:

- User choice: Precomputed FFC array (.mat) or new GB_FFC() generation
- Interactive line drawing for rotation alignment
- Manual cropping selection

Outputs:

- cropped_image_array: Processed focal spot image
- Saved files: .mat (data) and .tif (visualization)

Dependencies:

- GB_FFC.m
- GB_image_window_and_level.m (**Appendix A.3.i**)
- GB_crop_image.m (**Appendix A.3.ii**)

Author: Grant Budge

```
clc; clear; close all;
```

Step one is to call FFC and to return an array that we can work with

```
choice = input('Would you like to:\n    1. Load an existing FFC array\n    2. Create\n    a new FFC array\nEnter 1 or 2: ');

if choice == 1
    % You can select which .mat file to choose from. that way its an array.
    [file, path] = uigetfile('*.mat', 'Select the saved FFC array');
    if isequal(file, 0)
        disp('No file selected. Exiting script.');
```

```
        return;
    else
        loadedData = load(fullfile(path, file));
        image_array = loadedData.FFC;
        disp(['Loaded FFC array from: ', fullfile(path, file)]);
    end
```



```

elseif choice == 2
    %Create a new FFC array
    image_array = GB_FFC();
else
    error('Invalid input. Please restart and enter 1 or 2.');
```

end

```

array_minimum_value = min(image_array(:));
if array_minimum_value < 0
    image_array = image_array - array_minimum_value;
end

%
%
%
```

Step two is to adjust the contrast and window so that its easier to work with visually.

```

[final_level, final_window] = GB_image_window_and_level(image_array, 1);
disp(['Final Level: ', num2str(final_level), ' \n Final Window: ',
num2str(final_window)]);
%
close all;
%
% Adjust the values of image_array using final level and final window
min_val = final_level - final_window / 2;
max_val = final_level + final_window / 2;
%
% put image_array to the range [min_val, max_val]
adjusted_image_array = max(min(image_array, max_val), min_val);
%
% Normalize the adjusted image to a range of [0, 65535]
adjusted_image_array = (adjusted_image_array - min_val) / (max_val - min_val) *
65535;
%
%Display the adjusted image
fig_adjusted = figure;
imshow(adjusted_image_array, [], 'Colormap', gray);
title('The Adjusted Image');
colorbar;
```

```
%  
%  
%
```

We need to rotate the image such that the focal spot is horizontal / vertical

```
% Loop until happy with alignment  
is_happy = false;  
roi = [];  
  
while ~is_happy  
    % Ask the user to draw the line  
    disp('Please draw a line along the length of the focal spot.');  
    % Bring the adjusted image figure to front  
    figure(fig_adjusted);  
  
    % Delete the last line if it exists  
    if ~isempty(roi) && isvalid(roi)  
        delete(roi);  
    end  
  
    roi = drawline;  
  
    % Endpoints for slope calculation  
    position = roi.Position;  
    x1 = position(1, 1);  
    y1 = position(1, 2);  
    x2 = position(2, 1);  
    y2 = position(2, 2);  
  
    % Compute the angle with respect to the x-axis  
    rotation_angle = atan2(y2 - y1, x2 - x1) * (180 / pi);  
  
    % Rotate image  
    rotated_image_array = imrotate(adjusted_image_array, rotation_angle, 'nearest',  
    'crop');  
  
    % Display the rotated image
```

```

fig_rotated = figure;
imshow(rotated_image_array, [], 'Colormap', gray);
title(['Rotated Image (Rotated by ', num2str(rotation_angle), ' degrees)']);
colorbar;

% Ask user if they are happy with the rotation
choice = input('Are you happy with the alignment? (1 = Yes, 2 = No): ');

if choice == 1
    is_happy = true; % Exit loop
    disp(['The image was rotated by ', num2str(rotation_angle), ' degrees to
align focal spot.']);
    close(fig_adjusted); % Close the original adjusted image figure
else
    close(fig_rotated); % Close the rotated image figure
    disp('Redoing line selection...');
end
end

```

Now we can crop around the focal spot

```

% Call the new function to crop the rotated image array
cropped_image_array = GB_crop_image(rotated_image_array);

figure;
imshow(cropped_image_array, [], 'Colormap', gray);
title('Cropped Focal Spot');
colorbar;

```

We can save this NEW rotated and cropped array as both a .tif and a .mat

```

%
% .mat
%
[file, path] = uinputfile('*.mat', 'Save the Cropped and Rotated array as a .mat');
if isequal(file, 0)
    disp('You decided not to save the final array.');
```

```

else
    save(fullfile(path, file), 'cropped_image_array');
```

```
disp(['You saved as: ', fullfile(path, file)]);  
end
```

Published with MATLAB® R2024b

A.3.i GB_image_window_and_level.m

Code Description

GB_image_window_and_level - Interactive Contrast Adjustment Tool

This function provides interactive window/level adjustment for X-ray focal spot images with automated contrast recommendations. Integrates with Hamamatsu detector analysis pipelines for optimized image visualization.

Key Features:

1. Dual-mode operation: Automated percentile-based contrast suggestions
2. Interactive terminal interface for manual refinement
3. Real-time image updates with colorbar display
4. Comprehensive statistical reporting (min/max/mean/median/mode)
5. Input validation to prevent invalid contrast ranges

Inputs:

- input_array: 2D image data (typically flat-field corrected)
- fig_num: Figure handle for display

Outputs:

- final_level: Center value of contrast range
- final_window: Width of contrast range

Called By:

- GB_image_rotation.m

Dependencies:

- Image Processing Toolbox (imshow)
- Statistics and Machine Learning Toolbox (prctile)

Author: Paul Johns

```
function [final_level, final_window] = GB_image_window_and_level(input_array,
fig_num)
    fmtstr_6 = ['\n', 'Min = ', '%6d  \n', ' avg = ', '%8.1f  \n', ' med = ', '%6d  \n', '
mode = ', '%6d \n', ' max = ', '%6d  \n'];
    fmtstr_7 = ['\n', 'Current level = ', '%8d'];
    fmtstr_8 = ['Data range = ', '%8d', ' Current window = ', '%8d'];

    % Get array stats
    array_maximum_value = max(input_array(:));
    array_minimum_value = min(input_array(:));
    mode_array_value = mode(input_array, 'all');
    NaN_excluded_array = input_array(~isnan(input_array));
    mean_array_value = mean(NaN_excluded_array);
    median_array_value = median(NaN_excluded_array);

    % Auto contrast using percentiles
    lowerPercentile = prctile(input_array(:), 0.1);
```

```

upperPercentile = prctile(input_array(:), 99.9);

% Recommended values
autoContrastMin = lowerPercentile;
autoContrastMax = upperPercentile;
recomended_level = (autoContrastMin + autoContrastMax) / 2;
recomended_window = autoContrastMax - autoContrastMin;

% Ensure window is not zero
if recomended_window <= 0
    recomended_window = 1;
end

% Initialize level and window
array_level = recomended_level;
array_window = recomended_window;

% Display initial image
figure(fig_num);
imshow(input_array, "DisplayRange", [array_level - 0.5 * array_window,
array_level + 0.5 * array_window]);
colorbar;

% Interactive Adjustment
flag = 1;
while (flag)
    fprintf(fmtstr_6, array_minimum_value, mean_array_value, median_array_value,
mode_array_value , array_maximum_value);
    fprintf(fmtstr_7, array_level);
    fprintf('\nRecommended Level: %.2f \n', recomended_level);
    fprintf('Recommended Window: %.2f \n', recomended_window);

% Get user input
val1 = input('Enter new level (negative number to skip): ');
if (val1 <= 0)
    flag = 0;
else
    val2 = input('Enter new window (negative number to skip): ');
    if (val2 <= 0)

```

```

        flag = 0;
    end
end

if (flag)
    if (~isempty(val1)), array_level = val1; end
    if (~isempty(val2)), array_window = val2; end

    % Ensure valid limits
    array_value_lower_limit = array_level - 0.5 * array_window;
    array_value_upper_limit = array_level + 0.5 * array_window;
    if (array_value_upper_limit == array_value_lower_limit)
        array_value_upper_limit = array_value_lower_limit + 1;
    end

    % Update display
    imshow(input_array, "DisplayRange", [array_value_lower_limit,
array_value_upper_limit]);
    colorbar;
end
end

% Return final level and window values
final_level = array_level;
final_window = array_window;
end

```

Published with MATLAB® R2024b

A.3.ii GB_crop_image

Code Description

GB_crop_image - Interactive Image Cropping Function

This function provides interactive region-of-interest (ROI) selection for cropping focal spot images in X-ray detector analysis workflows.

Inputs:

- image_array: 2D array of processed image data (typically rotated FFC output)

Outputs:

- cropped_image_array: Subregion of input array containing focal spot

Called By:

- GB_image_rotation.m

Dependencies:

- Image Processing Toolbox (imcrop, drawrectangle)

Author: Grant Budge

```
function cropped_image_array = GB_crop_image(image_array)
    close all;

    % Display for cropping
    figure;
    imshow(image_array, [], 'Colormap', gray);
    title('Select the region to crop around the focal spot');
    colorbar;

    % Cropping region of interest (in green because I like green)
    roi = drawrectangle('Label', 'Crop Region', 'Color', 'g');

    % take the position of the roi
    position = roi.Position;

    % crop the region
    cropped_image_array = imcrop(image_array, position);
    close(gcf);
end
```


Appendix B

B.1 GB_minimize_noise.m

Code Description

GB_minimize_noise - Adaptive Threshold Normalization (ATN) for Noise Reduction

This function implements the ATN method to minimize structured noise in radiographic images by normalizing row intensities using edge column averages. Particularly effective for Hamamatsu S11684-12 detector images with characteristic edge artifacts.

Key Features:

1. Dual-mode operation: Automatic (recommended) or manual column selection
2. Adaptive normalization using edge column statistics
3. NaN-aware calculations for robust processing
4. Row-wise correction preserving central image features

Methodology:

1. Samples left/right edge columns (default: cols 1-150 & 851-1000)
2. Computes row-wise averages of edge regions
3. Calculates global normalization constant 'a'
4. Applies multiplicative correction to entire image

Inputs:

- input_array: 2D image array with noise patterns
- input_value: Control flag (0=auto, 1=manual)

Outputs:

- minimized_noise_array: Corrected image array

Usage Notes:

- Automatic mode recommended for standard detector configurations
- Manual mode allows custom column ranges for specialized applications

Author: Grant Budge

Affiliation: Carleton University

```
function [minimized_noise_array] = GB_minimize_noise(input_array, input_value)

%-----
```

Section 1:

```
%-----
% The needed stats of the array will be found.
%
OG = input_array;
% Size
[rows, columns] = size(OG);
```

Section 2:

```
%-----  
% This will ask the user for a selection of columns on the left side of  
% their radiograph. It will tell them the size they have to work with and  
% recommend a standard setting.  
fprintf('\n');  
fprintf('The number of columns in your array is %f', columns);  
fprintf('\n We suggest your left side column to be around 1-150 and right side to be  
around 851-1000. \n');  
flag = 1;  
  
while true  
    % THIS IS TO MAKE IT SO YOU CAN MAKE THIS AUTOMATIC FOR DARK AND  
    % FOR GAIN IMAGES.  
    if input_value == 0  
        i=1;  
        j=150;  
        k=851;  
        l=1000;  
        flag = 0;  
        fprintf('\n Perfect! Lets fix that noise.\n');  
        break;  
    else  
        auto = input('For automatic averaging, enter 0. To manually select columns,  
enter any number other than 0: ');  
        if isnumeric(auto)  
            if auto == 0  
                i=1;  
                j=150;  
                k=851;  
                l=1000;  
                flag = 0;  
                fprintf('\n Perfect! Lets fix that noise.\n');  
                break;  
            else  
                break;  
            end  
        end  
    end  
end
```

```

        else
            disp('Please enter a valid numeric value.');
```

end

```

    end
end
while flag
    i = input('Enter the starting column of the LEFT side that you would like to
average (1): ');
    j = input('Enter the ending column of the LEFT side that you would like to
average (150): ');
    k = input('Enter the starting column of the RIGHT side that you would like to
average (851): ');
    l = input('Enter the ending column of the RIGHT side that you would like to
average (1000): ');
    % Check the conditions
    if j > i && k > j && l > k && l <= columns
        flag=0; % Exit the loop if all conditions are satisfied
    else
        fprintf('Invalid input. Please ensure that:\n');
        fprintf(' - %d is greater than %d\n',j,i);
        fprintf(' - %d is greater than %d\n',k,j);
        fprintf(' - %d is greater than %d\n',l,k);
        fprintf(' - %d is not greater than the number of columns (%d)\n\n', l
,columns);
    end
end
%
%-----
```

Section 3:

```

%-----
% The math for averaging the columns
%
left = OG(:, i:j); % make a matrix of the left selection (1-150).
right = OG(:,k:l); % make a matrix of the right selection (851-1000).
averaging_array = zeros(rows, j-i+1 + l-k+1); % make an averaging array the size of
the selected columns.
averaging_array(:,1:j-i+1) = left; % let the averaging array contain the left matrix.
```

```

averaging_array(:,j-i+2:end) = right; % let the averaging array contain the right
matrix.

% This averaging array is now filled with every value that the user
% selected. Some values may be NaN. We're going to average all of the
% Number values and ignore the NaN in EACH ROW. The average of each row
% will be put into a new array. We need to remember that if all of the
% numbers in the row are NaN, then we need to return NaN value. this will
% happen in the top 3 rows and bottom 3 rows every time.

averaged_array=zeros(rows,1);

% for all of the rows in averaging_array, we need to average the number and
% add it to the averaged array
for row = 1:rows
    row_values = averaging_array(row, :);
    % Check if the entire row is NaN
    if all(isnan(row_values))
        averaged_array(row) = NaN; % Keep NaN if all values are NaN
    else
        % Calculate the average of number values
        averaged_array(row) = mean(row_values(~isnan(row_values)));
    end
end

% Now we have our averaged array. we need to normalize it.
% Our normalization constant is "a"
a = mean(averaged_array(~isnan(averaged_array)));
averaged_array = a./averaged_array;

% This averaged array is now normalized and we can multiply value in the
% specific rows to make the noise better.

minimized_noise_array = OG.*averaged_array;

```