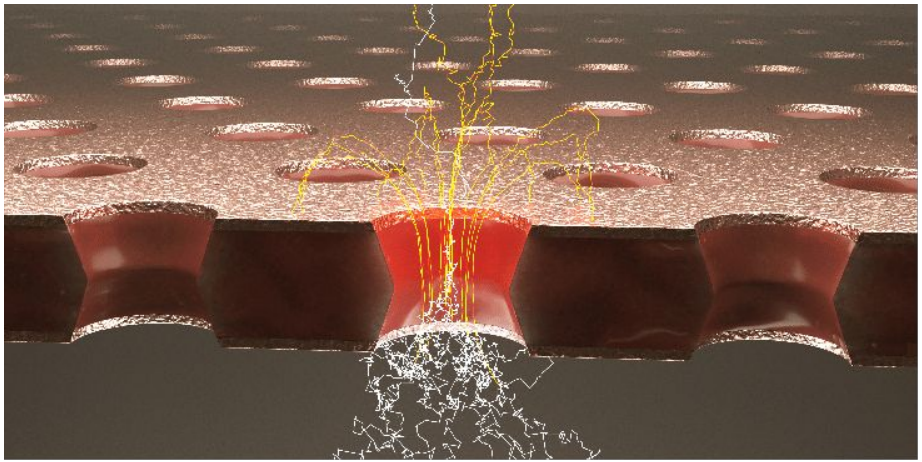
Imagens em hiper-resolução para inspeção de GEMs

Estudante: Caio de Sousa Ribeiro
Orientador: Tiago Fiorini da Silva

São Paulo, 15 de outubro de 2024

Multiplicadores gasosos de elétrons (GEM)

- Folha de poliimida com espessura de 50 μm revestida em ambos os lados com uma camada de 5 μm de cobre.
- Possui alta densidade de furos bicônicos.
- Alta capacidade de amplificação do sinal eletrônico.



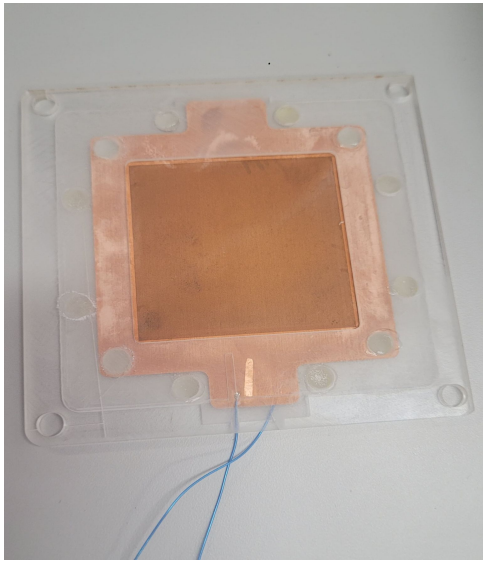
Produção e aprimoramento da tecnologia

- O grupo de pesquisa High-Energy Physics Instrumentation Center (HEPIC) estuda o desenvolvimento desta tecnologia e a aplicação em diversas áreas.

- Em particular, existe um foco na produção dos GEMs a partir de técnicas de ablação a laser e utilizando impressoras 3D.

Objetivos:

- Avaliar a consistência dos GEMs produzidos por meio de medições das características geométricas..
- Comparar diferentes métodos de produção, assegurando qualidade e viabilidade das placas em experimentos de alta precisão.



GEM produzido por ablação a laser, utilizado nos procedimentos do projeto.

Mestrado: Eduardo dos Santos Palermo

Aquisição de materiais

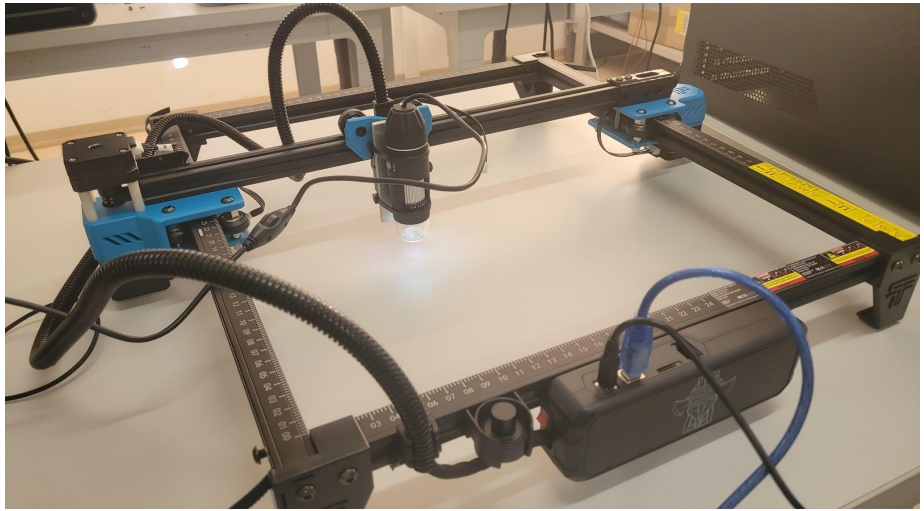
Microscópio USB.

- Posteriormente foi implementado um método de retirar margens de todos os lados da imagem, de uma maneira em que mantivesse a proporção 1:1.

```python
import cv2
import numpy as np

webcam = cv2.VideoCapture(0)

if webcam.isOpened():
    validacao, frame = webcam.read()
    frames_list = []
    while validacao:
        validacao, frame = webcam.read()
        cv2.imshow("Video da Webcam", frame)
        key = cv2.waitKey(5)
        if key == 112:  # p minúsculo
            img_name = "Frame_{}.png"
            cv2.imwrite(img_name, frame)
        elif key == 109:  # m minúsculo
            frames_list.append(frame)
        if key == 27:  # Esc
            break

# Calcule a média dos frames
mean_frame = np.mean(frames_list, axis=0).astype(np.uint8)
cv2.imwrite("Mean_Frame.png", mean_frame)
webcam.release()
cv2.destroyAllWindows()
```

Rotina utilizada para realizar aquisição de imagens utilizando as bibliotecas OpenCV e Numpy.

Aquisição de materiais



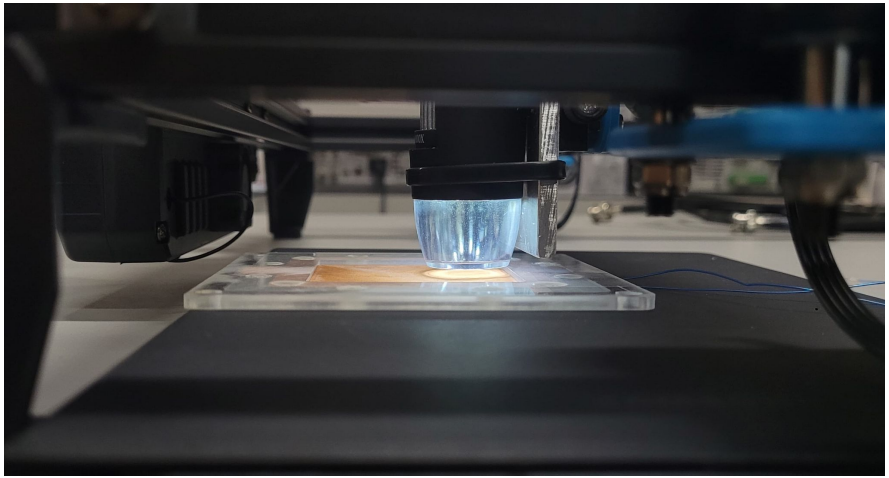
- Aquisição de gravadora a laser.
- Informações transmitidas através de uma conexão serial com o computador.

```python
import serial as serial

ser = serial.Serial(
        port = 'COM5', #Verificar a porta
        bytesize = 8,
        parity = 'N',
        stopbits = 1,
        baudrate = 115200,
        timeout = 10,
        xonxoff = False
)
ser.reset_output_buffer()
ser.reset_input_buffer()
val = 0

command = 'G90 \r\n'  # Modo de posicionamento absoluto
val = ser.write(command.encode(encoding='ascii', errors='strict'))
command = 'G21 \r\n'  # Define a unidade de medida como milímetros
val = ser.write(command.encode(encoding='ascii', errors='strict'))
command = 'G92 X0 Y0 \r\n'  # Define a posição atual como origem (0,0)
val = ser.write(command.encode(encoding='ascii', errors='strict'))

command = 'G0 X35 Y17 \r\n'  # Move para a posição (35,17)
val = ser.write(command.encode(encoding='ascii', errors='strict'))

ser.close()
```

Biblioteca pySerial e comandos em Gcode.

Configuração do scanner

- Definição de uma classe para a câmera com as funções relacionadas à aquisição de imagens.
- Implementação dos comandos mecânicos.



```python
camera = Camera()
frames_max = 5
x = float(input("Informe a dimensão alocada no eixo x (mm): "))
y = float(input("Informe a dimensão alocada no eixo y (mm): "))
print("Passo foi definido como 2.311 mm")
passo = 2.311
passo_str = str(passo)
y_div_passo = math.ceil(y / passo)
x_div_passo = math.ceil(x / passo)

set_abs_mode(ser)
pos_x = 0.00
pos_y = 0.00

for i in range(y_div_passo + 1):
    for j in range(x_div_passo + 1):
        if i % 2 == 0:  # Movimento da esquerda para a direita na grade
            pos_x = j * passo
        else:  # Movimento da direita para a esquerda na grade
            pos_x = (x_div_passo - j) * passo
        # Mover para a nova posição
        command = f'G0 X{pos_x} Y{pos_y} \r\n'
        val = ser.write(command.encode(encoding='ascii', errors='strict'))
        camera.capture_images(frames_max, round(pos_x, 2), round(pos_y, 2))
    pos_y += passo  # Atualizar a posição Y para a próxima linha

command = 'G0 X0 Y0 \r\n'  # Mover para (0,0)
val = ser.write(command.encode(encoding='ascii', errors='strict'))

camera.release()
```

Obtenção das imagens

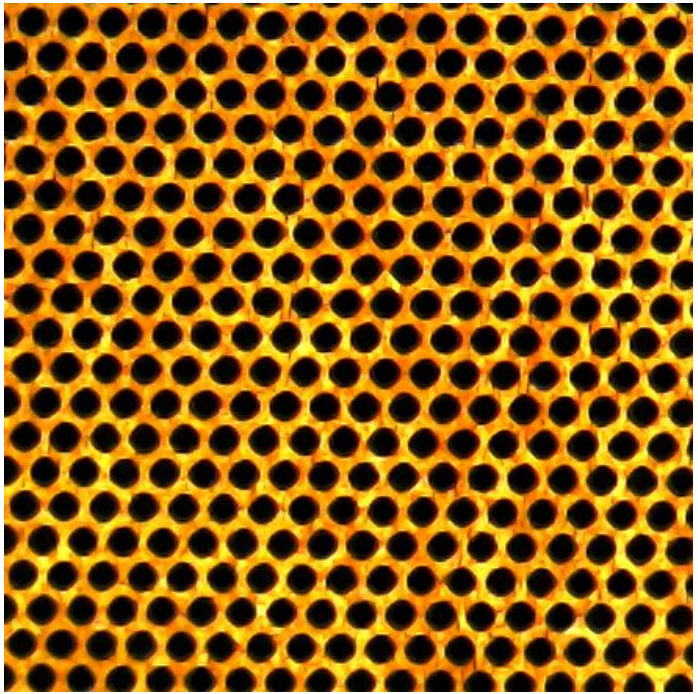
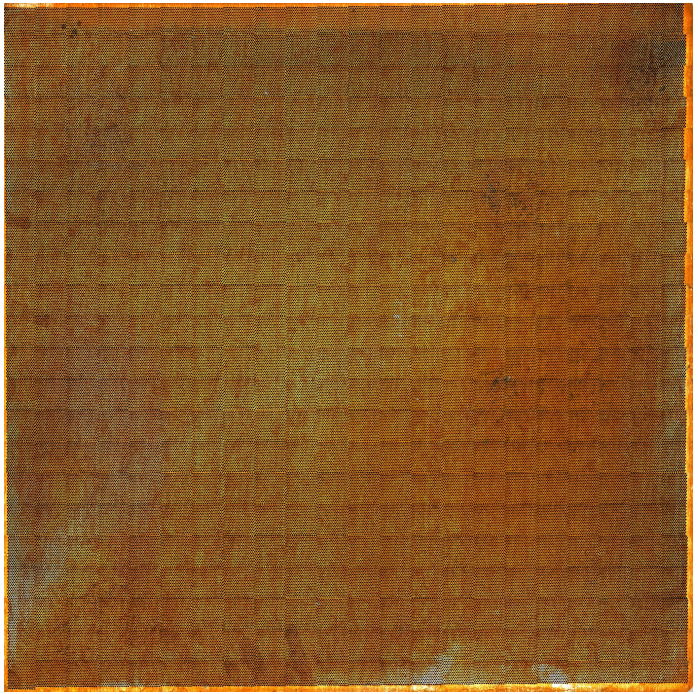


Imagem X_27.73-Y_32.35



Imagens concatenadas

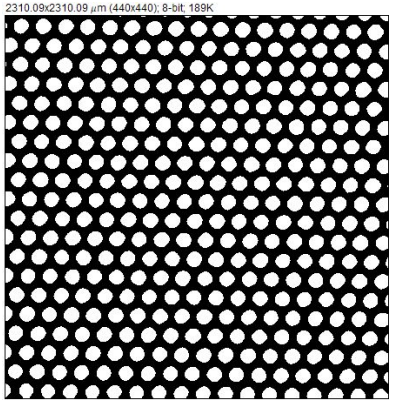
Análise das imagens

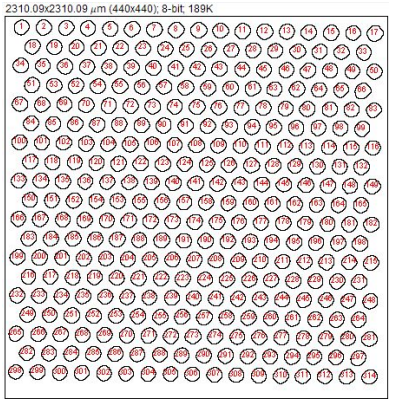
- Parte da rotina utilizada para operar o software ImageJ, incluindo procedimentos necessários para diferenciar os furos da superfície metálica.

- Escala definida como 5,25021 μm/pixel, a partir do calibrador do microscópio.

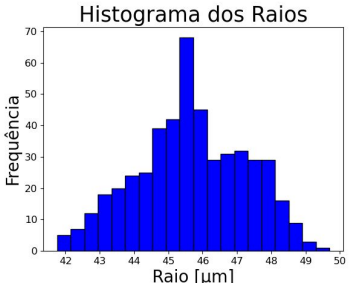
```
for (i = 0; i < list.length; i++) {
    if (endsWith(list[i], ".jpg") || endsWith(list[i], ".png")) {
        open(dir + list[i]);  // Abrir a imagem

        // Definir a escala da imagem
        run("Set Scale...", "distance=1 known=" + pixelWidth + " pixel=1 unit=" + unit);

        // Converter a imagem para escala de cinza
        run("8-bit");

        // Aplicar FFT Bandpass Filter
        run("Bandpass Filter...", "filter_large=40 filter_small=3 suppress=None tolerance=5");

        // Definir manualmente os valores de threshold usando o método Li
        setAutoThreshold("Li");
        run("Threshold...");

        // Converter para máscara
        run("Convert to Mask");

        // Limpar tabela de resultados anterior
        run("Clear Results");

        // Analisar partículas, habilitar circularidade, e salvar dados
        run("Analyze Particles...", "size=4500-8000 circularity=0.00-1.00 display exclude clear include summarize");

        // Arrays para armazenar os valores de área e circularidade
        areas = newArray(nResults());
        circularidades = newArray(nResults());

        // Coletar os dados de área e circularidade
        for (j = 0; j < nResults(); j++) {
            areas[j] = getResult("Area", j);
            circularidades[j] = getResult("Circ.", j);  // Coletar a circularidade corretamente
        }
```

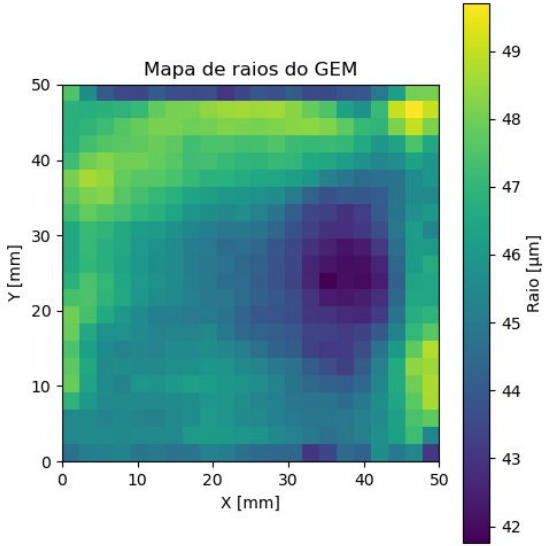
	Slice	Count	Total Area	Average Size	%Area	Mean	Circ.
207	X_27.73-Y_25.42.png	316	1903225.373	6022.865	35.664	255.000	0.946
208	X_27.73-Y_27.73.png	314	1898539.390	6046.304	35.576	255.000	0.949
209	X_27.73-Y_30.04.png	314	1942229.279	6185.444	36.395	255.000	0.947
210	X_27.73-Y_32.35.png	314	1987573.057	6329.850	37.245	255.000	0.948
211	X_27.73-Y_34.66.png	312	2046836.947	6560.375	38.355	255.000	0.946

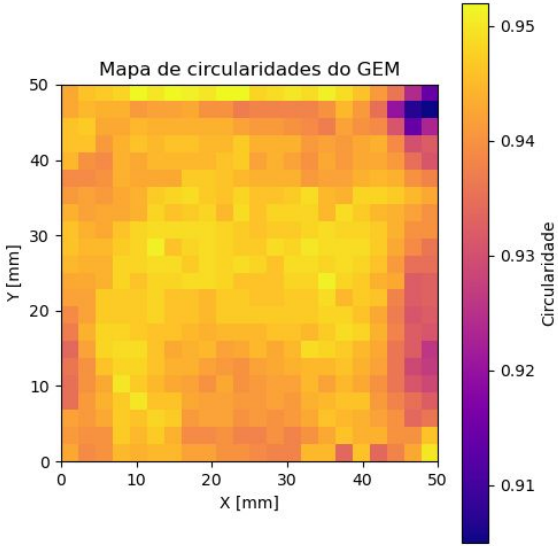


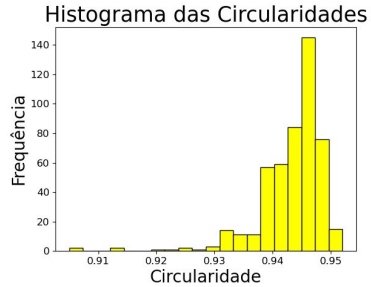


Resultados e Conclusões









$$\text{Raio} = \sqrt{\frac{\text{Área}}{\pi}}$$

Medição	Valor médio
Raio	45,7104(68) µm
Circularidade	0,94377(5)

$$\text{Circularidade} = \frac{4\pi(\text{Área})}{(\text{Perímetro}^2)}$$

Próximos passos

- Analisar GEM do CERN.
- Usar difusor de luz para ver o interior dos furos.

Referências

[1] M. Kalliokoski, T. Hilden, F. Garcia, J. Heino, R. Lauhakangas, E. Tuominen, R. Turpeinen, "*Optical scanning system for quality control of GEM-foils*", **Nucl. Instr. and Meth. Sec. A,** 664, 1, 223–230, 2012, https://doi.org/10.1016/j.nima.2011.10.058

[2] T. Hildén, E. Brücken, J. Heino, M. Kalliokoski, A. Karadzhinova, R. Lauhakangas, E. Tuominen, R. Turpeinen, "*Optical quality assurance of GEM foils*", **Nucl. Instr. and Meth. Sec. A**,770, 113–122, 2015, https://doi.org/10.1016/j.nima.2014.10.015

[3] Erik Brücken, Jouni Heino, Timo Hildén, Matti Kalliokoski, Vladyslav Litichevskyi, Raimo Turpeinen, Dezső Varga, "*Hole misalignment and gain performance of Gaseous Electron Multipliers*", **Nucl. Instr. and Meth. Sec. A,** 1002, 2021, 165271, https://doi.org/10.1016/j.nima.2021.165271

[4] Fabio Sauli, "*The gas electron multiplier (GEM): Operating principles and applications*", **Nucl. Instr. and Meth. Sec. A,**, 805, 2–24, 2016,, https://doi.org/10.1016/j.nima.2015.07.060.

[5] Python Software Foundation. Python Language Reference, versão 3.12. Disponível em: http://www.python.org

[6] Schneider, C., Rasband, W. & Eliceiri, K. NIH Image to ImageJ: "25 years of image analysis". Nat Methods 9, 671–675 (2012). https://doi.org/10.1038/nmeth.2089

Agradecimentos

À Fundação de Amparo à Pesquisa do Estado São Paulo (FAPESP #2023/15427-1)