UNIVERSIDADE FEDERAL RURAL DO SEMIÁRIDO - UFERSA
CURSO DE BACHARELADO EM CIÊNCIA E TECNOLOGIA
Trabalho de Conclusão de Curso (2021.2)

INTEGRAÇÃO DE UMA CÉLULA SOLAR COM ARDUINO E PYTHON.

Alisson Silva Amorim Duarte[1], Leonardo Augusto Casillo[2], Daniele Simone da Silva Casillo[3]

Resumo: Este trabalho visa implementar a associação do hardware livre Arduino e a linguagem de programação de código aberto Python para aprimorar um sistema de rastreamento solar. Tem-se como objetivo aprimorar o aproveitamento de um mini painel solar com o auxílio de componentes eletrônicos como servo motores e sensores de luminosidade. Tal aprimoramento será avaliado com base nas características das tensões geradas, sendo, maior duração dos picos de energia e decaimento acentuado da tensão conforme o passar do dia. Com este projeto foi possível verificar o aumento na eficiência na geração de tensão quando utilizado um sistema para ajuste contínuo dos eixos do painel.

Palavras-chave: Python; Arduino; microprocessador; energia solar.

1. INTRODUÇÃO

A energia solar vem sendo amplamente difundida entre as matrizes energéticas do mundo, não sendo diferente no Brasil. Por ser um dos meios menos agressivos ao meio ambiente, atrelado a fácil instalação e manutenção, esta opção vem crescendo a passos largos pelo país. As regiões Sul e Sudeste, apresentam os maiores índices de crescimento quando se fala em matriz energética solar, porém é a região do semiárido que comporta o maior potencial para geração de energia solar do território nacional conforme o portal de energia solar do Brasil [1].

Este projeto tem como propósito fundamentar a implementação de um pequeno sistema de rastreamento solar que possui a capacidade de aumentar a produção energética do equipamento, aprimorando ainda mais a relação de custo x benefício do sistema. Tanto o hardware quanto software utilizados são de emprego livre e foram escolhidos com intuito de possuir baixo custo sem perder a eficiência, além de ampla gama de periféricos e processos que aproximam a relação do usuário com o controle operacional.

2. DESENVOLVIMENTO

2.1.1. Microcontroladores

Um microcontrolador é um sistema de processamento integrado que reúne um núcleo de processamento, memórias dos tipos voláteis e não voláteis, periféricos e conectores para entrada e saída de dados [2]. O microcontrolador utilizado no circuito em questão foi o Arduino R3 por possuir um baixo custo e uma ampla gama de sensores e periféricos associados ao fato de ser um hardware livre e o tamanho compacto possibilita uma fácil integração com sistemas [13]. Sua IDE é construída para suportar a linguagem baseada em C, que apesar de apresentar uma rápida velocidade de execução, possui uma estrutura pouco legível e baixa integração da comunidade, tornando mais complexo a construção de sistemas diversificados.

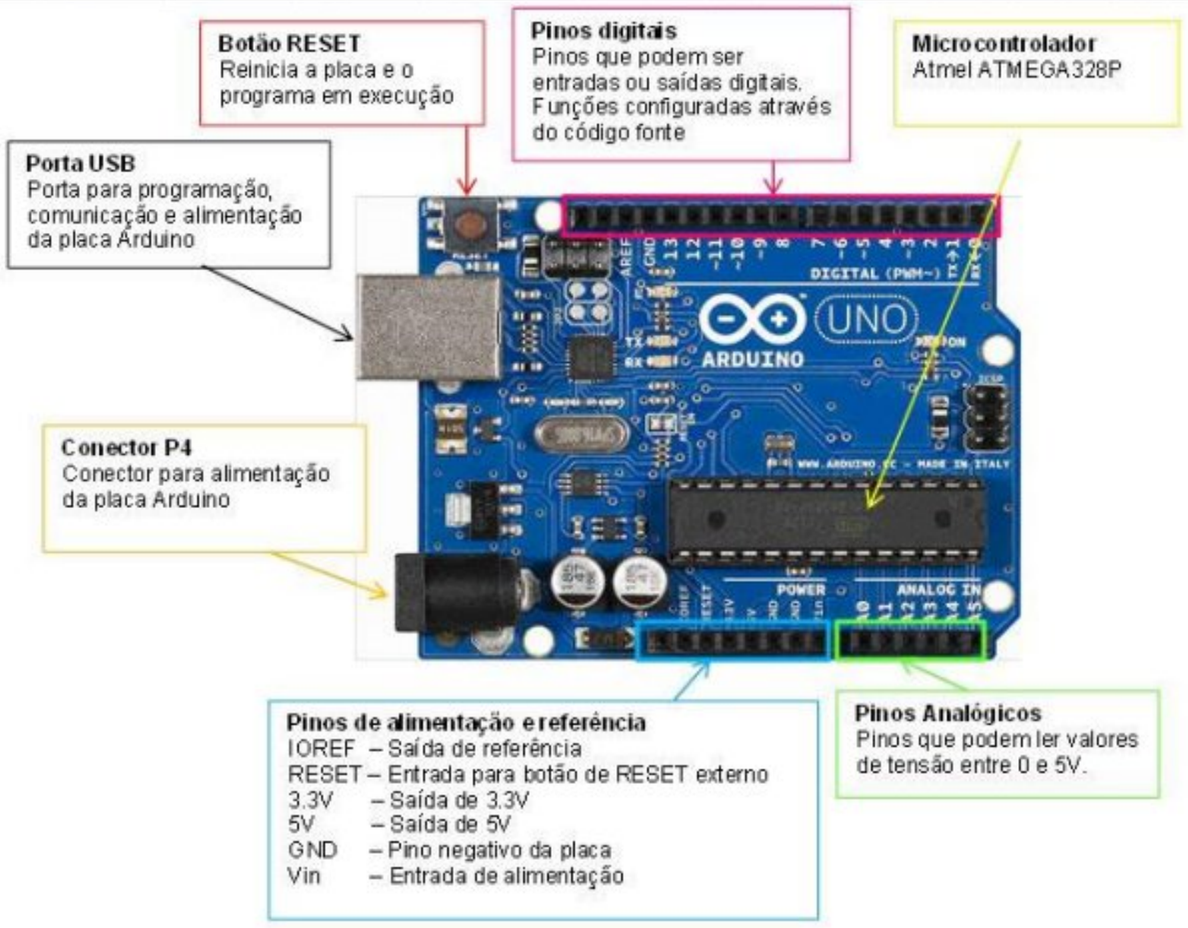

Figura 1. Modelo Arduino R3. (Fonte: Baú da eletrônica, 2018)

A grande vantagem do Arduino R3 é possuir entradas para conexões analógicas e digitais. As entradas analógicas podem ler dados variados com valores entre o intervalo 0 e 1 por exemplo. Uma entrada digital interpreta de forma binária, onde qualquer sinal recebido será interpretado como 1 e a ausência de qualquer sinalização será interpretada como zero. As operações lógicas e comandos são determinados a partir dessas leituras.

2.1.2. Sensores

Um sensor é qualquer dispositivo que responde algum tipo de estímulo, podendo ser físico ou químico, em sinais elétricos. O tipo do sensor, material de construção e arquitetura, depende da natureza do estímulo que este deve detectar, podendo ser magnetismo, temperatura, movimento, umidade, luminosidade e afins (Prof. Carlos Augusto, 2010).

Normalmente os sensores conectados ao Arduino são classificados em digitais e analógicos. Quando um sensor analógico é conectado na entrada analógica ele pode devolver grande variedade de números entre um intervalo por exemplo entre 0 e 1. Caso esse sensor seja conectado em uma entrada digital qualquer sinal recebido será interpretado como 1 e a ausência de sinal será interpretada como 0.

Para este projeto foram utilizados quatro sensores analógicos LDR (Light Dependent Resistor). Estes sensores possuem dois terminais e por trabalharem como resistores, não apresentam polaridade. Sua superfície é feita de sulfeto de chumbo, material semicondutor, e apresenta uma resistência na faixa de megaohm, quando a luz incide sobre ele, a energia luminosa transferida provoca o deslocamento de elétrons da camada de valência para as camadas mais externas, logo o componente se preenche de elétrons livres e afastados do núcleo provocando a redução ampla da resistência, que muda para faixa de dezenas ou centenas de ohms dependendo da luminosidade [3].

Figura 2. Modelos de LDR vendidos no mercado. (Fonte: manualdaeletronica, 2019)

2.1.3. Servo

Os servos são motores que consistem em controlar a variação de posição de forma precisa, seja pelo ângulo, seja pela velocidade de rotação. Seu circuito opera em uma malha fechada e consiste de forma central em um motor, funcionando a corrente contínua ou alternada, um potenciômetro, engrenagens para aumento de torque e um circuito de controle que recebe sinais do tipo *PWM*, este tipo de comunicação consiste em controlar a potência de um aparelho através do tempo em que a variação de tensão ocorre, de forma que se em certo período T, a variação de tensão aumenta por 0.2T, a variação média da tensão seria equivalente a 20% da tensão total [4].

Seu circuito de controle monitora a conexão com o microcontrolador conferindo o pulso a cada 20ms, quando o microcontrolador envia um pulso, este pode ter largura entre 1us a 2ms e esta largura irá determinar a variação do ângulo pelo servo [5].

O grande benefício do servo motor é a capacidade de gerar um torque alto em relação ao seu tamanho, permitindo que o sistema ao ter um ângulo fixado, não seja alterado pelo peso do próprio circuito.

Figura 3. Micro servo motor (Fonte: manualdaeletronica, 2019)

2.1.4. Protoboard

A Protoboard é uma placa com centenas de entradas para conexão de periféricos eletrônicos, normalmente possui entradas centrais, onde cada linha de cinco furos estão conectadas em série e quatro linhas laterais conectadas em série entre si, porém com maior capacidade de conexão, a linha vertical interior possui polo positivo e a vertical exterior é destinada para o uso da conexão *GND* ou terra.

2.2. Python

Python é uma linguagem de programação desenvolvida em 1991 por Guido van Rossum [6], esta linguagem tem como seu guia a simplicidade e versatilidade. É definida como de propósito geral além de ser interpretada, com tipagem dinâmica, caracterizada por sua fácil leitura e necessidade de poucas linhas de código para desempenhar a mesma função em comparação com outras linguagens equivalentes. Por priorizar o esforço humano sobre o esforço computacional, Python é considerada uma linguagem de alto nível, além de contar com uma equipe de desenvolvimento do seu código aberto, a linguagem também conta com um robusto corpo de bibliotecas padrões e *frameworks* desenvolvidos livremente por terceiros criando assim um amplo acervo de mecanismos e utilidades para aplicações práticas em diversas áreas.

2.3. Metodologia

Inicialmente foi estabelecido o protocolo de comunicação Firmata pela própria IDE do Arduino através do pacote StandardFirmata. O protocolo Firmata atua como um interpretador dos comandos escritos em Python e emitidos pela comunicação Serial entre a placa e o computador, permitindo o controle do sistema pela própria IDE da linguagem utilizada [7].

Após concluída a instalação do protocolo toda a comunicação posterior com o circuito é escrita em Python, enviada por comunicação Serial e executada pelo Firmata [8].

Em sequência para estabelecer a comunicação da IDE Pycharm com o Arduino deve-se importar a biblioteca pyfirmata para que o código escrito em python seja enviado com a formatação serial, compatível com a porta na

qual a placa estabelece comunicação com o computador, após instalar e importar esta biblioteca, deve-se importar os seguintes objetos “Arduino”, “SERVO” e “util”, com isso os modulos iniciais básicos de comunicação estão instalados no ambiente. Em seguida basta efetivar a comunicação com a criação da variável “uno”:

```
from pyfirmata import Arduino, SERVO, util
import time
uno = Arduino("/dev/ttyACM0")
```

Esta variável recebe o tipo de microcontrolador que está sendo utilizado e um atributo indicando o endereço da comunicação serial, com isso sempre que a variável “uno” for chamada, ela irá referenciar a porta de conexão serial da placa com o computador.

O sistema de rastreamento, consiste na comparação de luminosidade entre quatro sensores LDR dispostos nas arestas da placa solar. Quando a variação dos valores recebidos pelos sensores for acima do tolerável, o sistema irá entender que deve se ajustar até que os valores lidos sejam os mais uniformes possíveis, devendo parar quando a superfície da placa se posicionar de forma perpendicular aos raios solares incidentes. Para ler cada um dos sensores foi criada uma função própria.

```
def leitura01():
    "variável uno deve sempre ficar no mesmo bloco da leitura"
    uno = Arduino("/dev/ttyACM0")
    it = util.Iterator(uno)
    it.start()
    ldr_leitura = uno.get_pin("a:0:i")
    "por default, ldr_leitura ficará com o A0"
    time.sleep(1)
    ldr_leitura = ldr_leitura.read()
    if ldr_leitura == 0 or ldr_leitura > 1:
        ldr_leitura = 0.5
    return ldr_leitura

def leitura02():
    uno = Arduino("/dev/ttyACM0")
    it = util.Iterator(uno)
    it.start()
    ldr_leitura = uno.get_pin("a:1:i")
    "por default, ldr_leitura ficará com o A1"
    time.sleep(1)
    ldr_leitura = ldr_leitura.read()
    if ldr_leitura == 0 or ldr_leitura > 1:
        ldr_leitura = 0.5
    return ldr_leitura

def leitura03():
    uno = Arduino("/dev/ttyACM0")
    it = util.Iterator(uno)
    it.start()
    ldr_leitura = uno.get_pin("a:2:i")
    "por default, ldr_leitura ficará com o A2"
    time.sleep(1)
    ldr_leitura = ldr_leitura.read()
    if ldr_leitura == 0 or ldr_leitura > 1:
        ldr_leitura = 0.5
    return ldr_leitura

def leitura04():
    uno = Arduino("/dev/ttyACM0")
    it = util.Iterator(uno)
    it.start()
    ldr_leitura = uno.get_pin("a:3:i")
    "por default, ldr_leitura ficará com o A3"
    time.sleep(1)
    ldr_leitura = ldr_leitura.read()
    if ldr_leitura == 0 or ldr_leitura > 1:
        ldr_leitura = 0.5
    return ldr_leitura
```

A palavra "*def*" indica o ínicio de uma função. Uma função em Python é um bloco de código que é executado sempre que invocado, isso permite que não seja necessário repetir as mesmas instruções em diferentes momentos do desenvolvimento de um código.

A função leitura()0N é encarregada de respectivamente em cada linha de:

-Direcionar onde irá ocorrer o processamento dos comandos subsequentes.

-Criar uma variável que irá iterar os valores enviados e recebidos.

-Iniciar a iteração

-Criar a variável "ldr_leitura", esta variável recebe o o método "*get_pin*" que tem como função apontar o endereço da porta física do Arduino onde o sensor ldr está conectado, respectivamente o endereço se refere ao tipo de porta sendo "a = analógica" ou "d = digital", um número de 0 a 13 correspondente com a numeração da entrada e o tipo ação que deve ser feita, sendo "i = *input*" que consiste em receber os valores lidos pela entrada,

"o = *output*" que consiste em passar valores ou comandos para o canal ou "p = *pulse withd modulation*" no qual consiste em transmitir um comando de forma ajustável como um potenciômetro digital.
-Pausar a execução da função por 1 segundo, a pausa se torna necessária para evitar o sobrecarregamento da comunicação, evitando erros e bugs no sistema. Este tempo pode ser removido ou variado conforme a capacidade de processamento do computador host.
-Atribuir outro método à variável "ldr_leitura", o comando ".*read*()" permite que os valores recebidos pelo input sejam lidos pelo computador.
-Por fim o comando "*return*" retorna os dados lidos pelo comando anterior. Podendo serem exibidos na tela ou armazenados diretamente em um banco de dados, conforme a necessidade do usuário.

A interação com o servo motor é feita de forma simples. Primeiro foi usado os seguintes comandos para estabelecer o tipo de porta usada e o tipo de firmware que seria usado:

```
uno.digital[3].mode = SERVO
uno.digital[6].mode = SERVO
```

Após configurar as portas digitais "3" e "6" para receberem as conexões do tipo servo, elas podem ser usadas para rotacionar os motores. Para poder relacionar a rotação de ambos os servos com os dados captados nos sensores, precisamos criar as seguintes funções com o parâmetro fixo "*angle*", com os valores do parâmetro não passados, a função poderá ser alimentada com valores dinâmicos quando for invocada.

```
def ajuste(angle):
    uno.digital[3].write(angle)
    time.sleep(0.1)

def ajuste2(angles):
    uno.digital[6].write(angles)
    time.sleep(0.08)
```

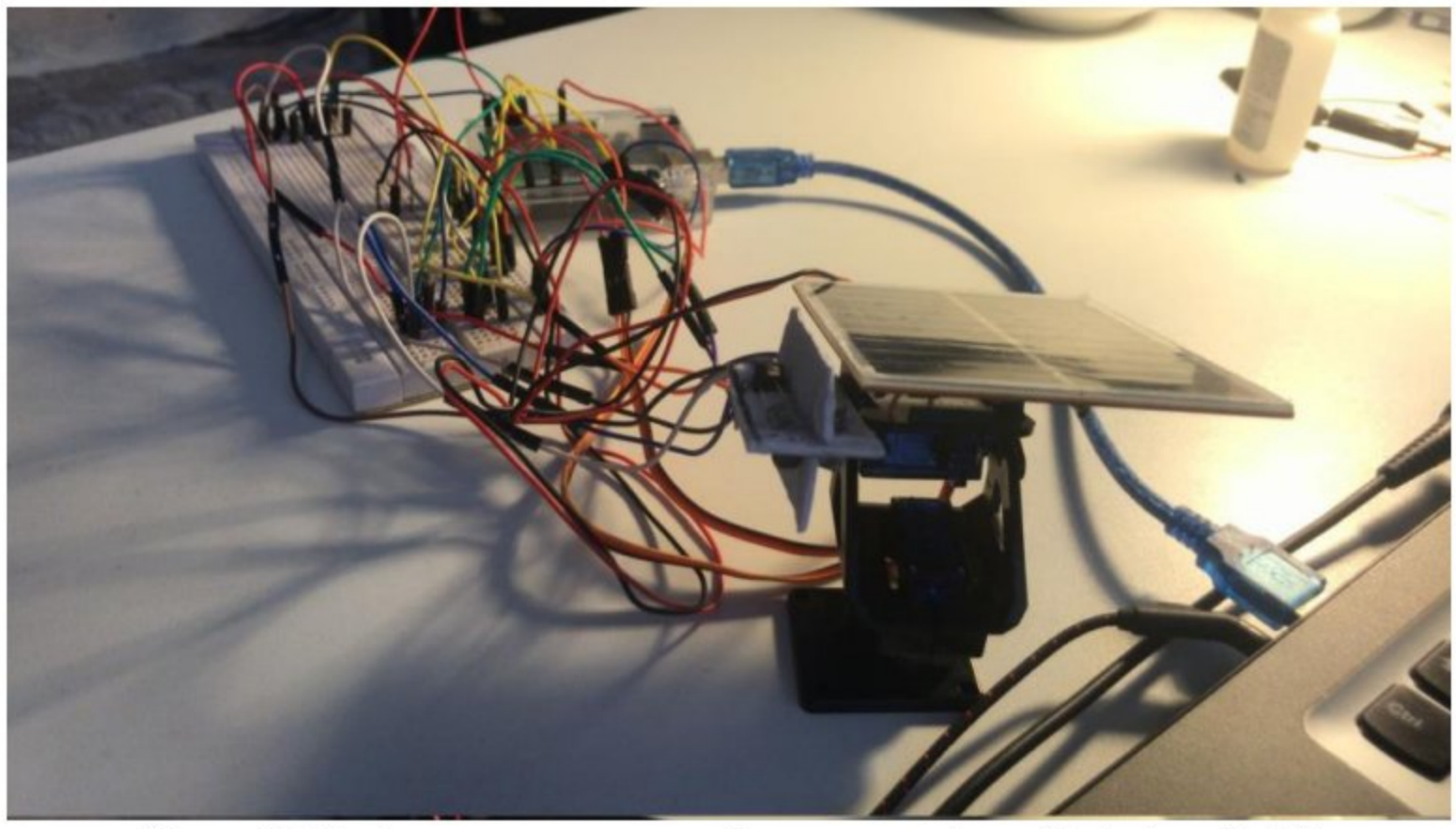

Figura 6. Estrutura que armazena ambos servo motores.(Autoria própria)

Para que o sistema possa ser monitorado e até mesmo estudado, é necessário que os dados gerados por ele sejam armazenados.

A função desenvolvida para registro de informações conta com dois comandos principais, ao ser chamada o comando "*with open*" possibilita que um arquivo de texto seja selecionado, o segundo parâmetro dado pela letra " 'a' ", faz alusão ao termo em inglês "*append*", traduzido para o português, significa "juntar". Com este trecho tem-se a determinação de abrir um arquivo e informar à máquina que informações novas serão juntadas às informações já presentes. Denominamos esse processo arbitrariamente de "arquivo", então sempre que quisermos selecionar o arquivo "*dataset_ldr.csv*" para anexar novas informações, precisamos apenas invocar a palavra "arquivo". Em seguida criamos uma variável denominada "registro" e atribuímos a ela a ação de escrever, através da palavra "*writer*". Feito isto, podemos determinar que linhas contento as leituras dos quatro

sensores e o horário delas sejam escritas em formato de texto sempre que a função "registrar()" for chamada, através do comando "*writerow*" e "*str*()".

Para a definição do momento da leitura, foram utilizadas duas bibliotecas, sendo elas "*pytz*" e "*datetime*", respectivamente elas capturam o fuso horário escolhido e a data atual sempre que são invocadas.

```
import datetime
import pytz
from csv import writer

def tempo():
    hAtual = pytz.timezone('Brazil/East')
    data_agora = str(datetime.datetime.now(hAtual))
    return data_agora[0:16]

def registrar():
    with open('dataset_ldr.csv', 'a') as arquivo:
        registro = writer(arquivo)
        registro.writerow([leitura01(), leitura02(),leitura02(),
                           leitura04(),str(tempo())])
```

Foram criadas duas variáveis chamadas "x" e "y" nas quais receberam os valores "90" e "5". Estas variáveis serão usadas para posicionar ambos os servos nos eixos horizontal e vertical antes de ser iniciado as leituras.

Após criar todas as variáveis e funções base do circuito, é necessário ativar seu funcionamento, este procedimento ocorre através da criação de um loop de repetição, onde é estabelecido uma condição, e sempre que essa condição for validada os comandos embutidos dentro do loop irão ser executados. Dentro do *loop* existe uma série de verificações condicionais que irão aferir as leituras e executar um comando a partir delas.

```
while True:
    ldr1 = int(leitura01() * 100)
    ldr2 = int(leitura02() * 100)
    ldr3 = int(leitura03() * 100)
    ldr4 = int(leitura04() * 100)
    registrar()
    if ldr1+ldr2 > ldr3+ldr4:
        if x != 170:
            x += 10
        ajuste2(x)


    if ldr1 + ldr2 < ldr3 +ldr4:
        if x != 0:
            x += -10
        ajuste2(x)


    if ldr1 + ldr3 > ldr2 +ldr4:
        if y != 70:
            y += 5
        ajuste(y)


    if ldr1 + ldr3 < ldr2 + ldr4:
        if y != 0:
            y += -5
        ajuste(y)
```

Para visualização dos resultados, foram desenvolvidas duas funções, sendo elas: "relatorio_ldr()" e "relatorio_geral()". A primeira função acessa o registro de leituras dos quatro sensores utilizados e transforma

em uma tabela juntamente com a data e hora corrente, após a criação da tabela, os dados são processados para um posterior relatório na forma de tabela contendo dados estatísticos e na forma de boxplot com os pontos representantes da leitura destacados. A segunda função realiza um processamento análogo porém pela necessidade de melhor entendimento da produção energética, a visualização ocorre através de um gráfico de linha, onde o desempenho pode ser observado de forma contínua, identificando altos e máximos decorrentes do desempenho real do equipamento.

```
import seaborn as sns
import matplotlib.pyplot as plt
def relatorio_ldr():
    df = pd.read_csv('dataset_ldr.csv')
    df = pd.DataFrame(data=df)
    sns.stripplot(data=df, color=".3")
    sns.boxplot(data=df)
plt.plot()
print(plt.show(), df)
```

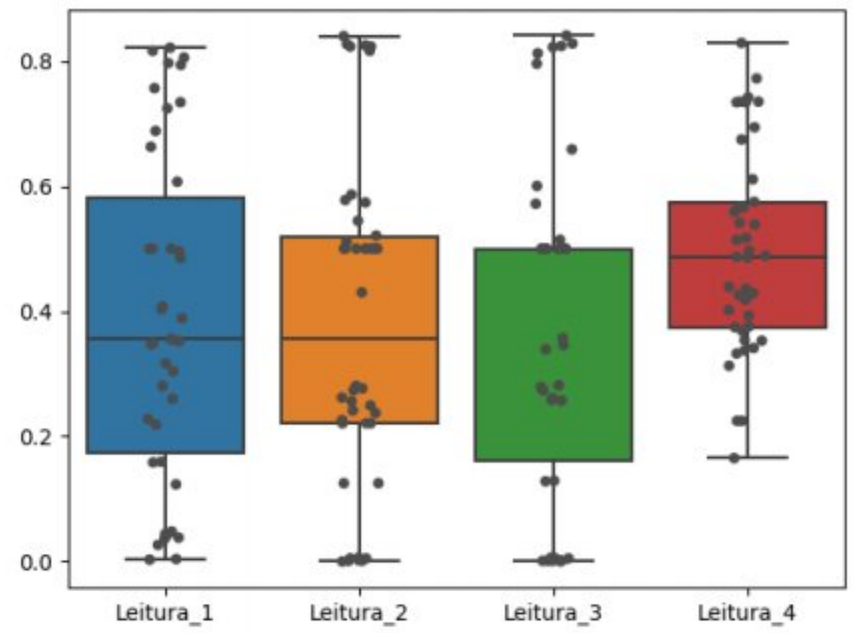

Foto 4. Visualização de leituras no formato de boxplot. (Autoria própria)

	Leitura_1	Leitura_2	Leitura_3	Leitura_4
count	42.000000	42.00000	42.000000	42.000000
mean	0.391329	0.37939	0.385705	0.493621
std	0.262868	0.26303	0.270742	0.162543
min	0.002900	0.00000	0.001000	0.165200
25%	0.174225	0.22090	0.161025	0.374650
50%	0.355300	0.35530	0.500000	0.486300
75%	0.580250	0.51855	0.500000	0.573075
max	0.821100	0.83970	0.840700	0.828900

Figura 5. Tabela de valores estatísticos sobre leituras de sensores.(Autoria própria)

Na figura abaixo está a representação do ciruito construído para o rastreamento solar.

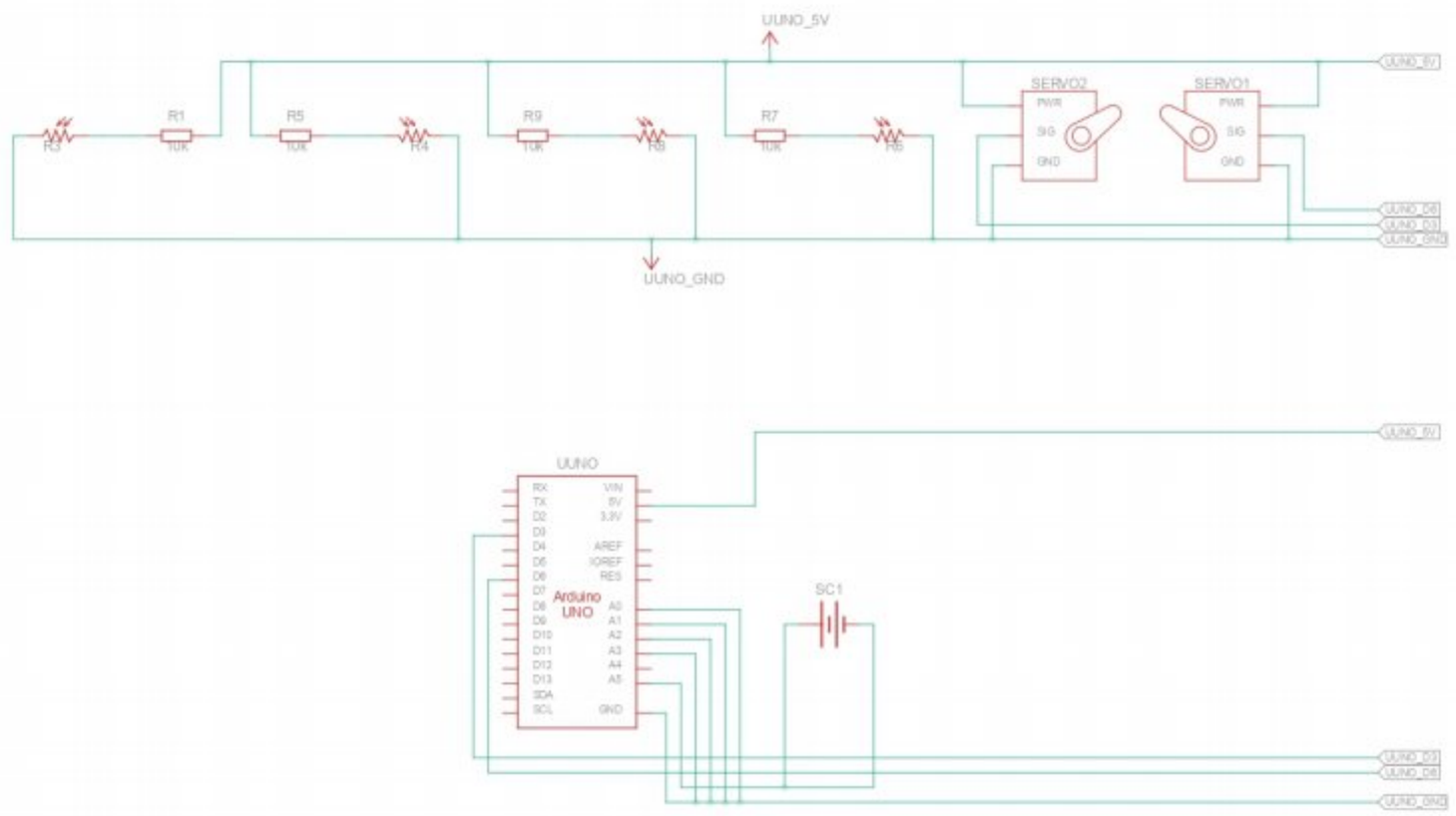

Figura 7. Sketch do circuito montado na plataforma Tinkercad.(Autoria própria)

Em seguida segue a estrutura já completa com todas as conexões montadas.

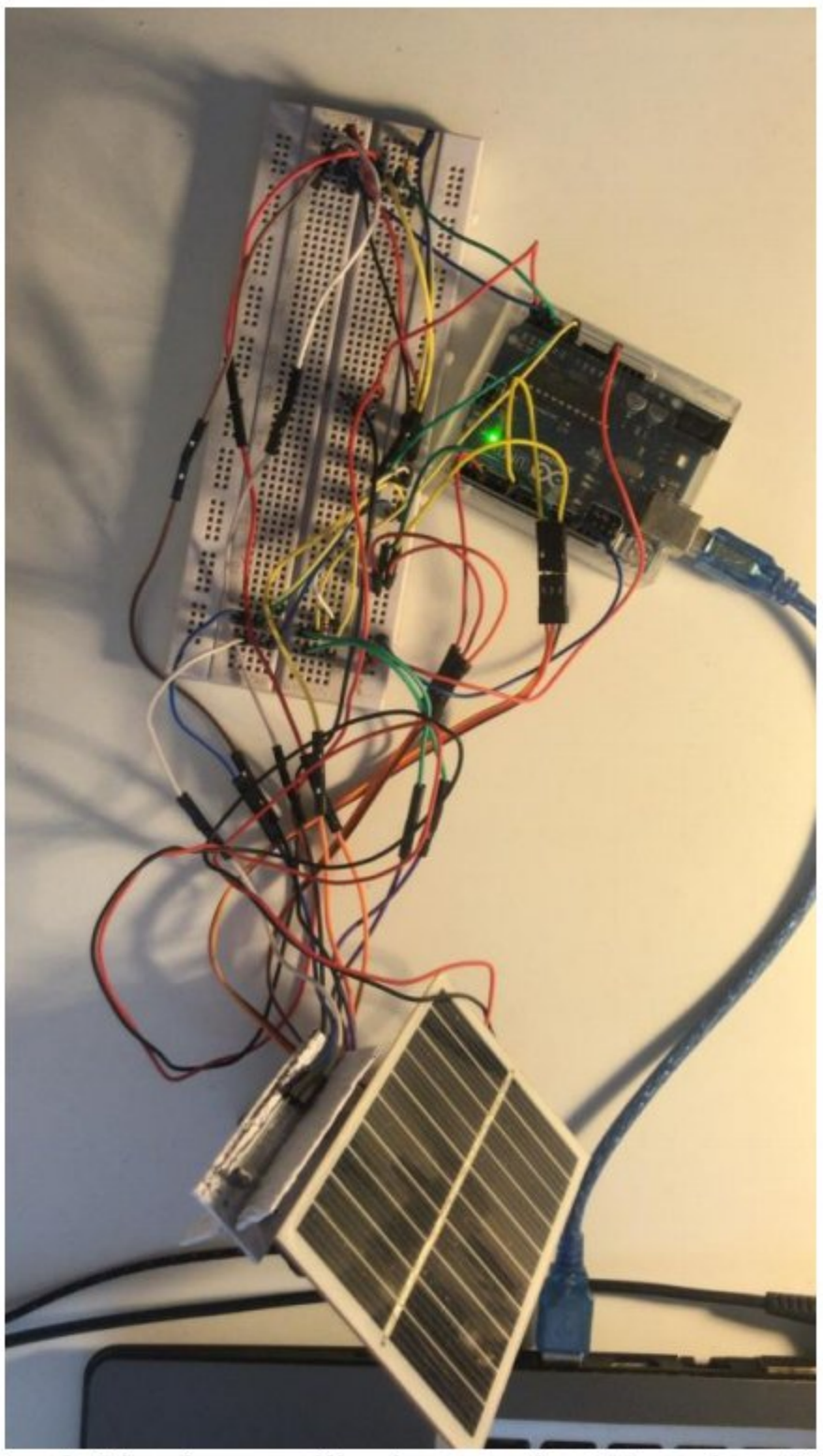

Figura 8. Plataforma utilizada no projeto.(Autoria própria)

2.4. Resultados

Foi realizado a medição de ddp gerada durante dois dias, sendo o primeiro dia reservado para a leitura de energia gerada sem o sistema de rastreamento no qual o painel solar ficou no chão de maneira estática e no segundo dia com o sistema de rastreamento ativado.

A figura a seguir descreve as leituras de tensão do primeiro caso.

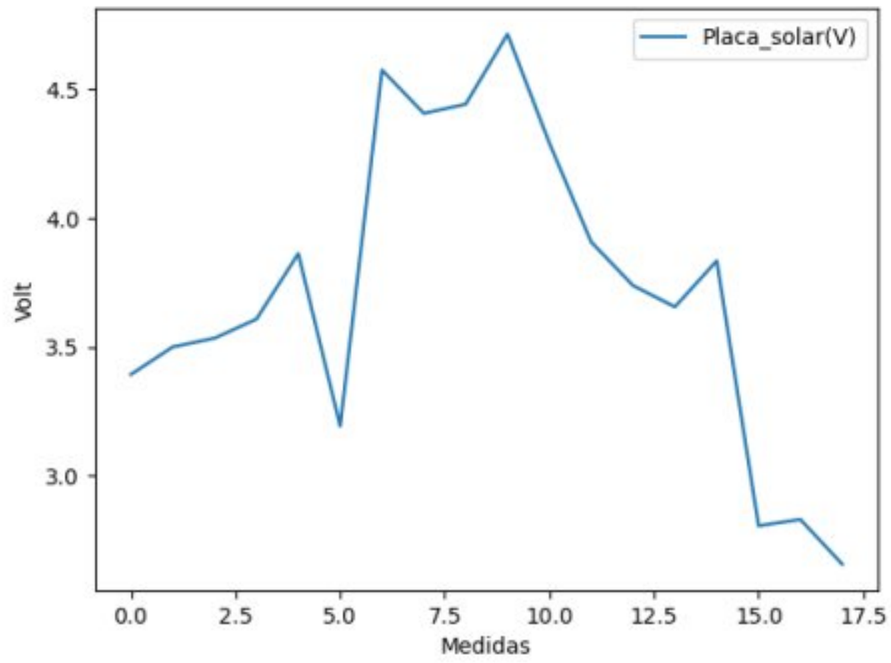

Figura 6. Leituras de tensão do painel solar sem o rastreador.(Autoria própria)

Percebe-se que as leituras da tensão apresentam uma elevada dispersão de valores devido a perda do ângulo de incidência mais propício para captação de luz.

Enquanto as leituras do segundo dia apresentam uma maior uniformidade entre os resultados. Tal uniformidade acontece devido ao ângulo de incidência da luz solar na superfície do painel ser mais uniforme, esta caracacterística induz um efeito fotovoltaico mais preciso, mantendo por mais tempo o pico de tensão.

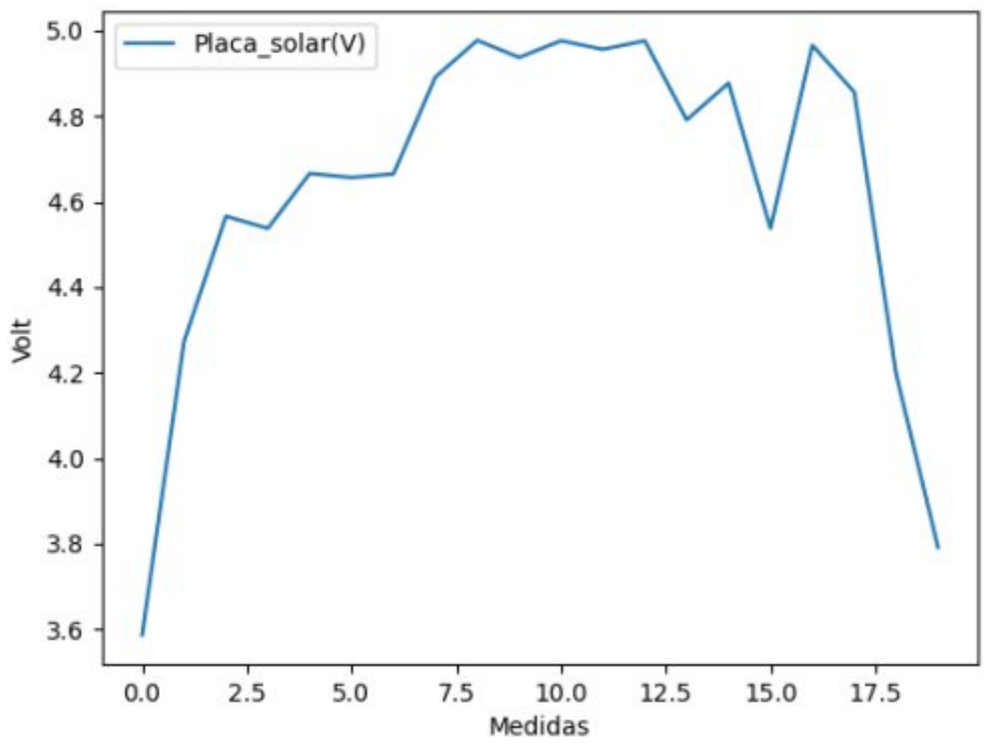

Figura 7. Leituras da tensão do painel solar com o rastreador.(Autoria própria)

Foi visto pela diferença entre as médias colhidas, um melhor desempenho na tensão com o sistema de rastreamento de aproximadamente 20%, tal diferença reflete no valor investido para a criação de uma estação completa ao explorar de forma mais eficaz o potêncial de cada um dos paineis.

Em certos momentos das leituras, em ambos os dias, há uma queda de tensão em momentos que a mesma deveria continuar constante, tais perdas de aproximadamente 0.5V foram causadas pelas condições atmósfericas no momento da leitura, principalmente pela grande movimentação de nuvens no céu. Há também pequenas variações no fluxo de até 0.2V em vários momentos, tais variações já eram esperadas pelas condições do equipamento.

É notável pelas medidas de dispersão apresentadas na tabela a seguir, que os valores medidos no segundo dia aprimoram o grau de aproveitamento do período diurno para a captação de energia solar.

Tabela 1. Medidas de dispersão e média.(Autoria própria)

Modo	Desvio padrão (V)	Variância (V^2)	Média (V)
Sem rastreamento	0.6118	0.3743	3.7189

Com rastreamento	0.3982	0.1586	4.6341

Analisando a amplitude dos dados, foi evidênciado uma diferença de 25% entre os valores mínimos de tensão, onde o sistema sem rastreamento em certos momentos estava gerando apenas 53.1% da capacidade da placa, o sistema com rastreamento em seu menor índice utilizou 71.7% da capacidade de tensão total do painel.

Tabela 2. Amplitude de valores medidos (Autoria própria)

Modo	Máximo (V)	Mínimo (V)
Sem rastreamento	4.714	2.655
Com rastreamento	4.977	3.586

3. Conclusões

A implementação do sistema de rastreamento solar de fato apresenta uma melhora significativa no desempenho do painel. Apesar de inconsistências meteorológicas, a estação solar construída com um sistema de rastreamento eficaz trás vantagens como melhor aproveitamento do espaço físico e mais eficiência da estação.

A despeito de não atingir os índices esperados de 40% de melhoria, o projeto proporcionou segurança de desempenho, sem a presença de falhas que tornassem o sistema inoperante nem problemas que comprometessem o rastreamento.

É fundamental destacar a homogeneidade dos resultados do segundo dia de coletas, onde o pico de energia foi mantido próximo da sua capacidade máxima durante a maior parte do período, tal característica consolida a segurança energética da estação, garantindo uma rápida recuperação da tensão que eventualmente se perde ocasionada pelas variações do ambiente natural, tal linearidade também pode amenizar problemas de dissipação no armazenamento de energia que será posteriormente utilizada.

4. Referências Bibliográficas

[1] Energia Solar no Brasil. Portal Solar, 2022. Disponível em: < https://www.portalsolar.com.br/energia-solar-no-brasil.html >. Acesso em: 10 abril 2022.

[2] OLIVEIRA, C. L. V.; ZANETTI, H. P. **Projetos com Python e Arduino** - Como desenvolver projetos práticos de eletrônica, automação e IOT. São Paulo: Saraiva, 2020. Disponível em: https://app.minhabiblioteca.com.br/#/books/9788536533575/. Acesso em: 13 abr. 2022.

[3] Servo Motor. [2013?] Disponível em <https://www.feis.unesp.br/Home/departamentos/engenhariaeletrica/aula-4---servo-motor-13-03-2013-final.pdf>. Acesso em 13 abril 2022.

[4] Arduino PWM. Arduino Tutorials: A collection of Arduino Web Tutorials of Hobbyists, [2013?]. Disponível em < https://www.arduino-tutorials.com/arduino-pwm/ >. Acesso em: 13 abril 2022.

[5] ALVES, Pedro. Ldr e como funciona. Manual da eletronica, 2019. Disponível em < https://www.manualdaeletronica.com.br/ldr-o-que-e-como-funciona/ >. Acesso em: 20 abril 2022.

[6] About. Python.org. 2022. Disponível em < https://www.python.org/>. Acesso em 20 abril 2022.

[7] Firmata Protocol,[2017?]. Arduino.cc. Disponível em < https://docs.arduino.cc/hacking/software/FirmataLibrary >. Acesso em 10 mar. 2022.

[8] Firmata Protocol Documentation, [2017?]. Disponível em < https://github.com/firmata/protocol.com>. Acessado em: 10 mar. 2022.

[9] TORRES, F. E.; MARTINS, H. R. Apostila didática PICMinas: Sistemas microcontrolados, V.1 Editora Axoon, 2010.

[10] Light Dependent Resistor: Photoresistor. EletronicsNotes. Disponível em < https://www.electronics-notes.com/articles/electronic_components/resistors/light-dependent-resistor-ldr.php >. Acesso em 15 jun. 2022.

[11]Duarte, André Victor. Elaboração de um controle eletrônico para utilização em uma estufa a partir de um forno elétrico doméstico. 2017. 8f. Trabalho de Conclusão de Curso (Bacharel em Ciência e Tecnologia). Universidade Federal Rural do Semi Árido.

[12] Duarte, Gustavo de Faria. Desenvolvimento de um protótipo de seguidor solar biaxial. 2019. 77f. Trabalho de conclusão de curso(Bacharel em Engenharia Elétrica). Escola politécnica. Universidade Federal do Rio de Janeiro.
[13] MONK, Simon. Programação com arduino: começando com sketches (Tekne). [Digite o Local da Editora]: Grupo A, 2017. 9788582604472. Disponível em: https://app.minhabiblioteca.com.br/#/books/9788582604472/. Acesso em: 09 jun. 2022.
[13] Monk, Simon, S.M. Programação com arduino: começando com sketchs(Tekne). 1° edição. São Paulo: Bookman Editora LDTA, 2013
[14] SENSORES. Unesp. Disponível em < https://www.feg.unesp.br/Home/PaginasPessoais/ProfMarceloWendling/4---sensores-v2.0.pdf >. Acesso em 20 mai. 2022.
[15] MENESES, André. Mando Engenharia. Tudo sobre LDR(Resistor Dependente da Luz). Disponível em< http://mundoengenharia.com.br/tudo-sobre-ldr-resistor-dependente-da-luz/ >. Acesso em 20 mai. 2022.
[16] Plataforma Arduino, universo de oportunidade. Baú da eletrônica.2018. Disponível em <https://blog.baudaeletronica.com.br/plataforma-arduino/>. Acesso em 22 jun. 2022.

MINISTÉRIO DA EDUCAÇÃO
Universidade Federal Rural do Semi-Árido – UFERSA
Centro de Ciências Exatas e Naturais – CCEN

ATA DE DEFESA DE TRABALHO DE CONCLUSÃO DE CURSO

Às 14:30 horas do dia dezessete de junho de dois mil e vinte e dois, de forma remota a partir da sala virtual meet.google.com/sgy-onxg-hjb, reuniu-se a Banca Examinadora de defesa de trabalho de conclusão de curso de autoria do aluno **ALISSON SILVA AMORIM DUARTE**, aluno do curso de Bacharelado em Ciência e Tecnologia desta universidade, N° de matrícula 2015010762, com o título "**INTEGRAÇÃO DE UMA CÉLULA SOLAR COM ARDUÍNO E PYTHON**". A Banca Examinadora ficou assim constituída por quatro membros: Prof. Dr. Leonardo Augusto Casillo, presidente da banca e orientador do Trabalho de Conclusão de Curso; Profª. Drª. Danielle Simone da Silva Casillo, coorientadora, Prof. Dr. Danniel Cavalcante Lopes e Prof. Dr. Paulo Henrique Lopes Silva, como membros. Concluída a defesa, procedeu-se o julgamento pelos membros da banca examinadora, em reunião fechada, tendo o aluno sido considerado **APROVADO**. E para constar, eu, Leonardo Augusto Casillo, lavrei a presente ata que, após lida e aprovada pelos membros da banca examinadora, será assinada por todos.

Mossoró, 17 de junho de 2022.

Assinatura dos membros da Banca Examinadora.

Leonardo Augusto Casillo
Assinado de forma digital por Leonardo Augusto Casillo
Dados: 2022.06.20 11:13:33 -03'00'

Prof. Dr. LEONARDO AUGUSTO CASILLO – UFERSA
Presidente e orientador

Danielle Simone da Silva Casillo
Assinado de forma digital por Danielle Simone da Silva Casillo
Dados: 2022.06.20 11:10:29 -03'00'

Prof. Dr. DANIELLE SIMONE DA SILVA CASILLO – UFERSA
coorientadora

Prof. Dr. DANNIEL CAVALCANTE LOPES – UFERSA
Primeiro Membro

PAULO HENRIQUE LOPES SILVA: 05095966492

Assinado digitalmente por PAULO HENRIQUE LOPES SILVA:05095966492
DN: CN=PAULO HENRIQUE LOPES SILVA:05095966492, OU=UFERSA - Universidade Federal Rural Do Semi-Arido, O=ICPEdu, C=BR
Razão: Eu estou aprovando este documento
Localização:
Data: 2022-06-17 21:31:26
Foxit Reader Versão: 9.0.1

Prof. Dr. PAULO HENRIQUE LOPES SILVA - UFERSA
Segundo Membro