

**UNIVERSIDADE DE SÃO PAULO
ESCOLA DE ENGENHARIA DE SÃO CARLOS
DEPARTAMENTO DE ENGENHARIA AERONÁUTICA**

**THIAGO BUCHIGNANI DE AMICIS
ORIENTADOR: PROF. ALVARO MARTINS ABDALLA**

INTRODUÇÃO AO RCS: ABORDAGEM COMPUTACIONAL E EXPERIMENTAL

**São Carlos
Agosto de 2020**

ÍNDICE

1	INTRODUÇÃO	6
2	OBJETIVOS	8
2.1	Introdução ao RCS	8
2.2	Introdução à análise computacional	8
2.3	Introdução à análise experimental	8
3	PRINCÍPIOS DO RADAR CROSS SECTION	9
3.1	Funcionamento de um radar convencional	9
3.2	Radares em aviação	9
3.3	Radares monostáticos e bi-estáticos	10
3.4	Coordenadas esféricas	10
3.5	Alcance	11
3.6	Equação do RCS	12
3.6.1	RCS em dBsm	12
3.7	Fatores que influenciam na medida do RCS	12
3.7.1	Material	13
3.7.2	Geometria	13
3.8	Radar a partir de um sensor LiDAR	14
4	INTRODUÇÃO AO POFACETS	16
4.1	Instalando o programa	16
4.2	Iniciando o POfacets	17
4.3	Importando e criando modelos	18
4.4	Cálculo do RCS monostático	19
4.5	Cálculo do RCS bi-estático	23
5	APLICAÇÃO PRÁTICA E EXPERIMENTAL	25
5.1	Materiais	25
5.1.1	Eletrônica	25
5.1.2	Construção	26
5.1.3	Softwares	26
5.2	Construção do radar	27
5.2.1	Modelagem em CAD	27
5.2.2	Construção do radar	28
5.3	Montagem do circuito	30
5.4	Validação do radar	32
6	RESULTADOS	35
6.1	Resultados para a aeronave F-35	36
6.1.1	Análise como scanner	36
6.1.2	Análise de RCS com o sensor LiDAR	36
6.2	Resultados para a aeronave X-29	38
6.2.1	Análise como scanner	38
6.2.2	Análise de RCS com o sensor LiDAR	39

7 CONCLUSÃO	41
7.1 Sugestões para trabalhos futuros	41
8 REFERÊNCIAS	43
APÊNDICE A	44
APÊNDICE B	51

LISTA DE SÍMBOLOS E ABREVIATURAS

θ	Azimute	
λ	Comprimento de onda	
σ	Radar Cross Section	
ϕ	Elevação	<i>CAD</i> Computer-Aided Design
f	Frequência	<i>LiDAR</i> Light Detection And Ranging
G	Ganho da antena	<i>RCS</i> Radar Cross Section
P_t	Potência emitida pelo radar	
r	Raio das coordenadas polares	
R	Distância entre corpo e radar	

Lista de Figuras

1	Esquema de radar: (a) monostático; (b) bi-estático	10
2	Coordenadas esféricas	11
3	Caça Lockheed F-117 Nighthawk	14
4	Onda emitida pelo sensor LiDAR	15
5	Instalando o programa	16
6	Selecionando o diretório	17
7	Tela inicial do POfacets	18
8	Criando uma fuselagem simples	18
9	Acessando os modelos do POfactes	19
10	Seleção da análise a ser realizada	19
11	Aba da análise monostática	20
12	Modelo facetado pronto para análise	21
13	Gráficos de RCS monostático do F35: (a) linear; (b) polar	22
14	Modelo 3D do F35 com RCS indicado na seção	22
15	Interface da análise bi-estática	23
16	Gráficos de RCS bi-estático do F35: (a) linear; (b) polar	24
17	Modelo 3D do F35 com RCS bi-estático indicado na seção	24
18	Montagem do radar em CAD	27
19	Detalhes da montagem: (a) base; (b) topo	28
20	Montagem final do radar	29
21	Detalhe da base do radar	29
22	Detalhe do topo do radar	30
23	Círculo para montagem do motor NEMA 17	31
24	Círculo para montagem do sensor LiDAR	32
25	Objeto utilizado para a verificação	33
26	Visualização da assinatura de radar da verificação	33
27	Visualização da superfície da verificação	34

28	Modelos de aeronave utilizadas com o radar	35
29	Resultado do scanner para a aeronave F-35: (a) vista superior; (b) vista isométrica	36
30	Resultado do RCS para a aeronave F-35: (a) em dBsm; (b) em m^2	37
31	Resultado do RCS para a aeronave F-35 utilizando POfacets	37
32	Gráficos polares para a aeronave F-35 em dBsm: (a) radar; (b) POfacets	38
33	Resultado do scanner para a aeronave X-29: (a) vista superior; (b) vista isométrica	39
34	Resultado do RCS para a aeronave X-29: (a) em dBsm; (b) em m^2	39
35	Resultado do RCS para a aeronave X-29 utilizando POfacets	40
36	Gráficos polares para a aeronave X-29 em dBsm: (a) radar; (b) POfacets	40

1 INTRODUÇÃO

A tecnologia de detecção de aeronaves surgiu em 1904, na Alemanha. Na época, a tecnologia possuia baixa precisão, era de difícil manufatura e os sistemas de detecção de ondas eram inefficientes. Mesmo assim, tal tecnologia mostrava-se necessária de ser aprimorada, uma vez que poder identificar a presença de aeronaves ou navios a longas distâncias era algo fundamental em tempos belicosos, entre as duas guerras mundiais.

Apenas durante a Segunda Guerra Mundial que essa tecnologia foi muito desenvolvida, principalmente pelos ingleses, que a utilizaram para detectar aeronaves alemãs e prever possíveis ataques. Graças a tal descoberta, a força aérea britânica teve grande vantagem pelos ares durante a guerra.

Desde então, os radares passam a assumir papel fundamental na aeronáutica para a detecção de aeronaves em voo (ou até mesmo em solo). Sua importância ficou evidenciada graças a sua aplicação na Segunda Guerra Mundial.

Todavia, com o desenvolvimento dos radares, surge também a necessidade de se 'ocultar'. Dentro do contexto militar principalmente, não se deseja que o inimigo consiga identificar a sua aeronave em voo com antecedência. Além disso, a baixa assinatura de radar por parte de uma aeronave funciona como um elemento surpresa a ser usado contra o inimigo.

Segundo essa tendência, aeronaves militares passaram a se preocupar com mais um parâmetro durante seu projeto: *Radar Cross Section* (RCS), ou seja, o quanto detectável ela é perante radares.

Uma aeronave famosa que foi projetada de forma a minimizar sua assinatura de radar é a *Northrop Grumman B-2 Spirit*. Sua geometria faz com que as ondas emitidas pelos radares sejam refletidas de forma a "confundir" o receptor e não indicar a presença dela.

Além disso, essa aeronave possui diversas modificações no que diz respeito à saída de gases dos motores e outras características que possam indicar sua presença através da temperatura do ar. Entretanto, tais características fogem do escopo deste trabalho.

Não apenas na aviação militar a análise de RCS é importante. No contexto da aviação civil, conhecer a assinatura de radar de cada aeronave distinta permite sua identificação caso

necessário.

Esses exemplos demonstram a importância da realização de cálculos para estimar o RCS de uma aeronave e também de métodos simples e eficientes para se testar em modelos em escala da aeronave. Além disso, revelam a importância deste assunto para a formação dos estudantes de engenharia.

Desta forma, esse projeto se apresenta como uma introdução ao estudo e análise computacional e experimental de RCS em aeronaves.

O projeto e a construção de um radar de pequena escala utilizando um sensor infravermelho do tipo LiDAR (*Light Detection and Ranging*) integrado com um Arduino MEGA. Também será utilizado o programa POfacets em MATLAB para fazer estimativas a respeito do RCS dos mesmos modelos e comparação de resultados obtidos através do radar construído.

2 OBJETIVOS

Tendo em vista a importância da assinatura de radar no contexto aeronáutico, deseja-se fornecer aos estudantes interessados uma base para que esses possam se iniciar na área.

Dessa forma, deseja-se avaliar diferentes abordagens para uma estimativa computacional e experimental da assinatura de radar de uma aeronave em escala.

2.1 Introdução ao RCS

Desenvolver um material didático simplificado com conceitos básicos sobre assinatura de radar, envolvendo desde uma abordagem intuitiva até a análise física do problema.

2.2 Introdução à análise computacional

Desenvolver uma material didático que explique o funcionamento do software POfacets para análise computacional de RCS de modelos de aeronave, o qual funciona em conjunto com o MATLAB, de forma a facilitar a imersão do estudante interessado na área.

2.3 Introdução à análise experimental

Construir um radar de baixo custo utilizando um sensor LiDAR, de forma a atuar como um scanner de objetos, de forma a introduzir a parte prática na análise da assinatura de radar de uma aeronave.

3 PRINCÍPIOS DO RADAR CROSS SECTION

3.1 Funcionamento de um radar convencional

Um radar é basicamente formado por um conjunto emissor-receptor de ondas, as quais podem ou não ser eletromagnéticas.

O emissor emite uma onda de curta duração, de forma a reduzir a interferência devido as demais ondas que podem estar presentes no ambiente, a qual produzirá um 'eco' quando encontrar com um objeto. O receptor, então, capta o eco gerado pela onda emitida.

Todavia, parte da energia enviada na forma da onda pelo radar é refletida em uma direção diferente daquela enviada pelo emissor ou então absorvida pelo próprio corpo. O restante é direcionado para o receptor do radar e denomina-se eco, e é conveniente descrever esse eco em termos de uma área.

O RCS de um objeto pode ser entendido como a área projetada de uma esfera sujeita a mesma onda (em amplitude, comprimento de onda e direção) que o corpo. Tal 'esfera' tem seu tamanho definido a partir do ângulo de incidência da onda em relação ao objeto, de forma que os dois possuam o mesmo ângulo de incidência.

3.2 Radares em aviação

Segundo Eugene (1), os comprimentos de onda mais utilizados por radares são aqueles com frequência entre 2000 e 12000MHz.

Frequências mais baixas conseguem se propagar por maiores distâncias, dessa forma são utilizadas para controle de tráfego aéreo e monitoramento. As ondas de maior frequência são úteis para restreios de curto alcance e controle de mísseis, em que os alvos estão geralmente próximos ao míssil.

Tais frequências correspondem a um comprimento de onda da ordem de 50mm.

Além disso, os corpos que estão sendo analisados (aeronaves) possuem envergaduras que chegam a mais de 60m.

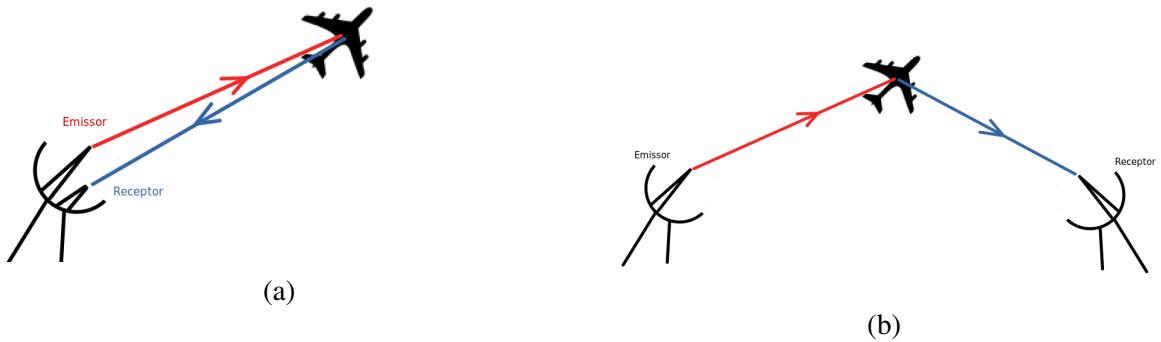
3.3 Radares monostáticos e bi-estáticos

Dependendo da posição relativa entre o emissor e o receptor de um radar, ele é classificado como bi-estático ou monostático.

Um radar monostático é aquele em que o emissor está posicionado muito próximo ao receptor. Caso o emissor e o receptor estejam separados por uma distância relativamente grande, da mesma ordem de grandeza do alcance do radar, então trata-se de um radar bi-estático.

A figura [1] mostra um esquema sobre a classificação dos radares de acordo com a posição do conjunto emissor/receptor.

Figura 1: Esquema de radar: (a) monostático; (b) bi-estático



Fonte: elaboração própria

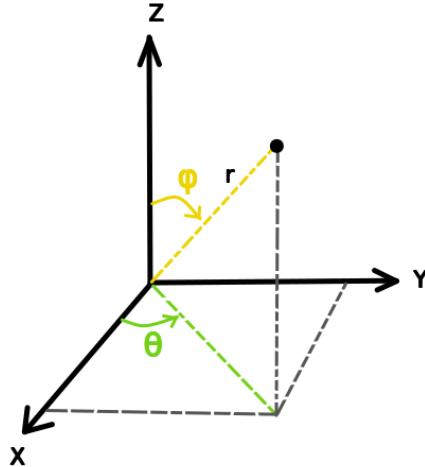
3.4 Coordenadas esféricas

O uso de um sistema de coordenadas esféricas permite a localização de objetos no espaço a partir de dois ângulos (elevação e azimute) e do raio.

Na figura [2], o ângulo θ representa o azimute, enquanto o ângulo ϕ representa a elevação correspondente; r é a distância da origem até o ponto em questão.

O uso de tais coordenadas funciona muito bem para o radar, uma vez que ao considerar que o emissor encontra-se na origem das coordenadas, os ângulos θ e ϕ são a orientação do radar no momento da emissão do sinal, enquanto a distância r se determina pelo tempo entre o momento que o sinal é emitido e percebido pelo receptor (Δt).

Figura 2: Coordenadas esféricas



Fonte: elaboração própria

Dessa forma, o parâmetro \mathbf{r} é definido por:

$$r = \frac{V_P \cdot \Delta t}{2} \quad (1)$$

em que V_P é a velocidade de propagação da onda, geralmente correspondendo à velocidade da luz, uma vez que radares geralmente emitem ondas do tipo eletromagnéticas.

A divisão por 2 é necessária pois o tempo total corresponde ao tempo de ida e volta do sinal.

3.5 Alcance

O alcance máximo de um radar é algo a ser determinado levando-se em conta o ruído que o receptor está constantemente recebendo em relação à intensidade do sinal de interesse.

Dessa forma, para garantir que a medição dada pelo radar seja válida, é necessário que a intensidade do sinal recebido seja maior do que a máxima intensidade de ruído que o radar recebe (P_{min}).

Segundo Eugene (1), para o caso em que o radar é do tipo monostático, a equação do alcance máximo fica definida como:

$$r_{MAX} = \left(\frac{P_t \cdot G^2 \cdot \lambda^2 \cdot \sigma}{(4\pi)^3 \cdot P_{min}} \right)^{\frac{1}{4}} \quad (2)$$

A partir da equação (2), percebe-se que para adequar o radar a uma determinada distância, sem variar a potência consumida, pode-se variar o comprimento de onda, apresentando a justificativa dos conceitos apresentados na seção 3.2.

3.6 Equação do RCS

A medida da assinatura de radar é, por definição, uma relação entre o quadrado dos módulos do campo elétrico das ondas recebida e enviada, quando a distância do radar até o objeto tende ao infinito, conforme indica a equação (3).

$$\sigma = \lim_{R \rightarrow \infty} 4\pi R^2 \frac{|E_s|^2}{|E_i|^2} \quad (3)$$

A distância infinita encontra boa aproximação com as distâncias reais entre um radar em solo e um avião no ar, da ordem de centenas de quilômetros, permitindo que a definição seja aplicada em casos reais.

3.6.1 RCS em dBsm

Ao utilizar a equação (3), temos um resultado em unidade de área. Todavia, também é comum se utilizar a unidade dBsm para indicar assinatura de radar. A conversão pode ser feita através da equação (4):

$$\sigma_{dBsm} = 10 \cdot \log_{10}(\sigma_m^2) \quad (4)$$

3.7 Fatores que influenciam na medida do RCS

Analizando a expressão (3), alguns parâmetros que influenciam na assinatura de radar de uma aeronave não estão evidenciados.

Isso pois a intensidade do sinal refletido de volta ao receptor pode diminuir em muito dependendo do tipo de material sob o qual as ondas estão refletindo e a geometria do corpo.

3.7.1 Material

O material do qual um corpo é feito influencia muito o eco que ele emite quando uma onda o 'atinge'. Isso se relaciona ao modo com que os átomos ou moléculas do material se organizam e reagem à presença ou à variação de um campo magnético externo. No caso do RCS, tal campo provém das ondas eletromagnéticas enviadas pelo radar.

Um bom material absorvente é aquele que conseguirá converter boa parte da energia que o atinge em energia térmica ou mecânica.

Um elemento que é comumente utilizado de forma a otimizar esse processo é o carbono, que atua como uma espécie de resistor, dissipando a energia da onda na forma de calor.

Além do carbono, existem materiais dielétricos, em que o índice de refração é um número complexo. As moléculas desses materiais formam diversos dipolos os quais se orientam a um campo magnético externo aplicado. Dessa forma, surge uma resistência à variação do campo. A energia que controla esse processo é absorvida pelo material na forma de calor.

3.7.2 Geometria

A geometria de um objeto também é um fator determinante para a assinatura de radar dele. As ondas emitidas pelo receptor, ao encontrarem com um corpo, irão se refletir em determinada direção dependendo do ângulo de incidência da onda com relação à superfície.

Para o radar, será apenas detectado o eco das ondas que cheguem de volta até a antena e consigam ser captados.

Dessa forma, pode-se projetar a geometria de uma aeronave de forma que as ondas que sobre ela incidirem sejam refletidas em uma direção diferente àquela que se encontra o receptor do radar.

A aeronave *Lockheed F-117 Nighthawk*, mostrado na figura [3] é um ótimo exemplo de design que segue como filosofia de projeto a diminuição do RCS. Para isso, a aeronave possui poucas curvas suaves e sua cauda é em V.

Figura 3: Caça Lockheed F-117 Nighthawk



Fonte: <https://www.af.mil/News/Photos/igphoto/2000605931/>

Dessa forma, as ondas incidentes sobre a aeronave tendem a serem refletidas em uma direção diferente àquela que se apresenta o receptor do radar.

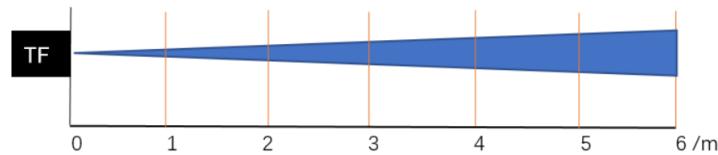
3.8 Radar a partir de um sensor LiDAR

O sensor LiDAR é um sensor de distância o qual mede a distância de um objeto a partir da reflexão da onda infravermelha emitida por ele. O tempo medido entre a emissão do sinal e a medição do mesmo permite avaliar a distância. Além disso, o sensor também é capaz de medir a 'força' (amplitude da onda) do sinal recebido.

O sensor utilizado é do tipo 'linear', ou seja, emite um feixe de luz na forma de uma pirâmide, resultando que atinge a superfície a ser medida com um quadrado de luz, como pode ser visto pela figura [4]. O ângulo de abertura do feixe é de $2,3^\circ$.

Existem outros tipos de sensor LiDAR mais complexos que emitem um feixe com ângulo de abertura maior, e são equipados com um maior número de emissores e receptores, de forma que são capazes de escanear em 360° toda uma região de uma única vez.

Figura 4: Onda emitida pelo sensor LiDAR



Fonte: Datasheet do LiDAR (2)

Segundo o datasheet do fabricante (2), o comprimento de onda do LiDAR utilizado nesse projeto é de 850nm, cerca de 60 mil vezes menor que aqueles utilizados na aviação.

Além disso, os modelos que foram analisados terão dimensão máxima de 20cm, dadas as limitações de projeto que serão discutidas na seção 5.2.

Outro ponto a ser considerado da análise utilizada pode ser entendido através da equação (3), em que para se medir a assinatura de radar de um objeto, considera-se que este está a uma distância muito grande do emissor. Todavia, o radar utilizado, conforme apresentado na seção 5, estará a menos de 50cm do alvo.

Dessa forma, uma vez que tanto o tipo de onda utilizado quanto as dimensões do envolvidas são muito diferentes, é necessária uma análise comparativa desses resultados para se avaliar a validade, ou não, da análise em escala.

4 INTRODUÇÃO AO POFACETS

O software POfacets (3) é um conjunto de rotinas computacionais que funcionam dentro do programa MATLAB, em conjunto com uma interface gráfica do próprio POfacets.

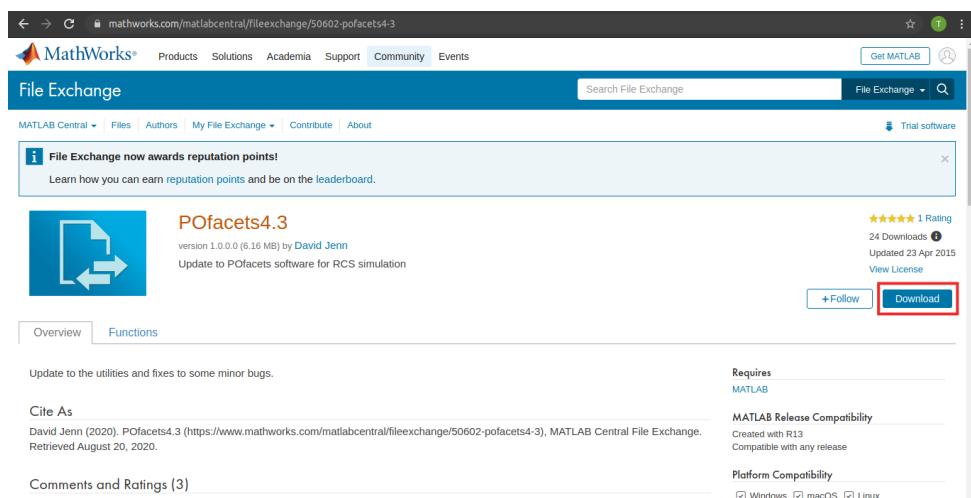
Tal software é a implementação numérica e computacional da teoria apresentada na seção 3, o qual permite calcular o RCS de objetos complexos, como aeronaves completas. Para tal, é utilizado o método *Physical Optics*.

Além disso, o programa é muito leve, não sendo necessário um computador potente para utilizá-lo. Com um computador de 1 CPU e 4GB de RAM é possível rodar o programa sem dificuldades, apesar de que os cálculos podem levar alguns poucos minutos para serem finalizados.

4.1 Instalando o programa

O software POfacets pode funcionar em qualquer sistema operacional, uma vez que é composto apenas por rotinas em MATLAB. Para instalar o programa, deve-se seguir para o site <<https://www.mathworks.com/matlabcentral/fileexchange/50602-pofacets4-3>> e clicar no botão indicado na figura [5].

Figura 5: Instalando o programa



Fonte: elaboração própria

Será instalado um arquivo compactado, o qual deve ser extraído todo dentro de uma única

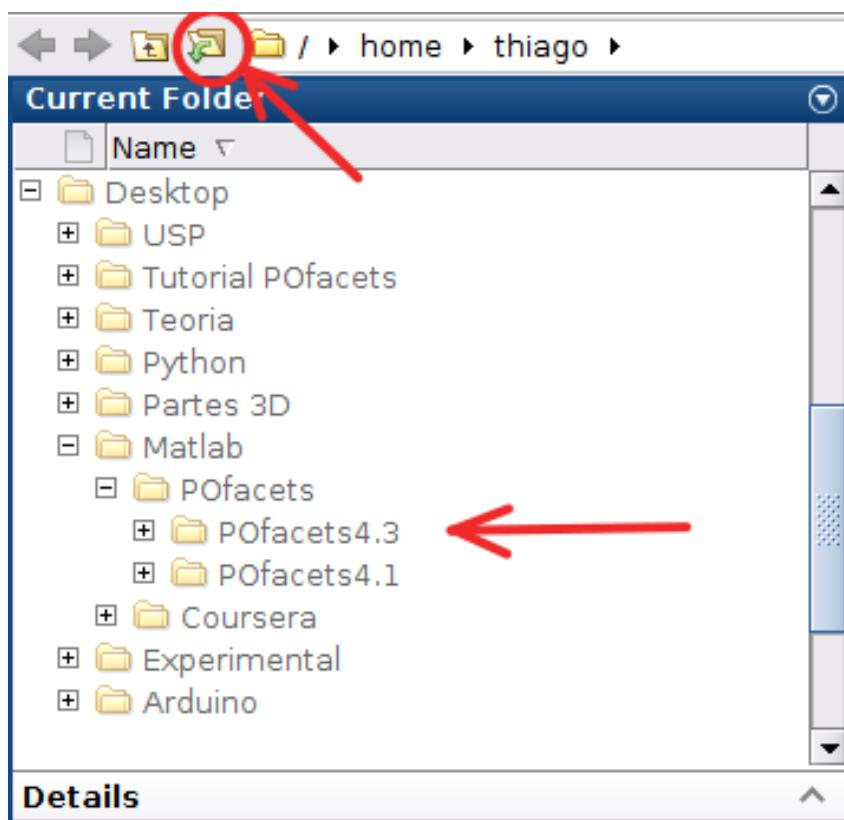
pasta.

Todo o funcionamento do programa é intuitivo devido a interface gráfica desenvolvida. A seguir, será demonstrado como o software pode ser utilizado a partir do cálculo do RCS de um modelo da aeronave *Lockheed Martin F-35 Lightning II*.

4.2 Iniciando o POfacets

Para iniciar o software, basta abrir a pasta em que se encontram as rotinas do programa dentro do MATLAB, conforme indicado na figura [6]. O usuário pode pesquisar por uma pasta específica alocada em seu computador através do botão *Browse for folder* na parte superior da figura, ou navegar diretamente pela aba *Current Folder* do MATLAB.

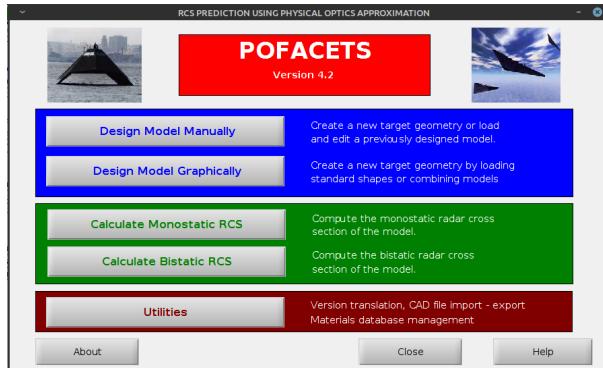
Figura 6: Selecionando o diretório



Fonte: elaboração própria

Após selecionado o diretório adequado, o programa é iniciado ao digitar o comando 'po-facets' na linha de comando. Após alguns instantes, a interface gráfica do programa se inicia, conforme ilustrado na figura [7].

Figura 7: Tela inicial do POfacets



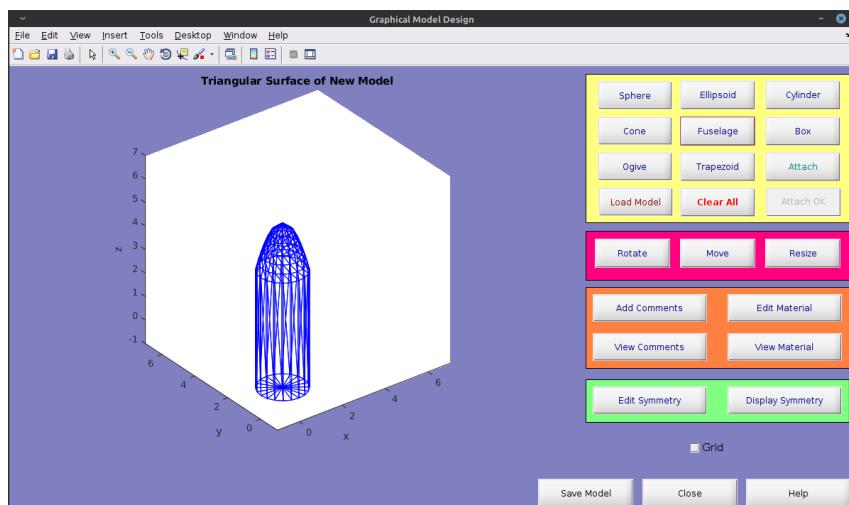
Fonte: elaboração própria

4.3 Importando e criando modelos

Dentro da interface do programa, há a opção de criar um modelo ou então modificar modelos que o usuário já possui. O software reconhece arquivos .STL e alguns tipos de arquivos de CAD, assim como planilhas indicando a posição dos vértices que formam a malha do modelo a ser analisado.

A figura [9] ilustra a tela em que se podem criar modelos novos a partir do zero. Na imagem, foi adicionada uma fuselagem simples, meramente ilustrativa.

Figura 8: Criando uma fuselagem simples



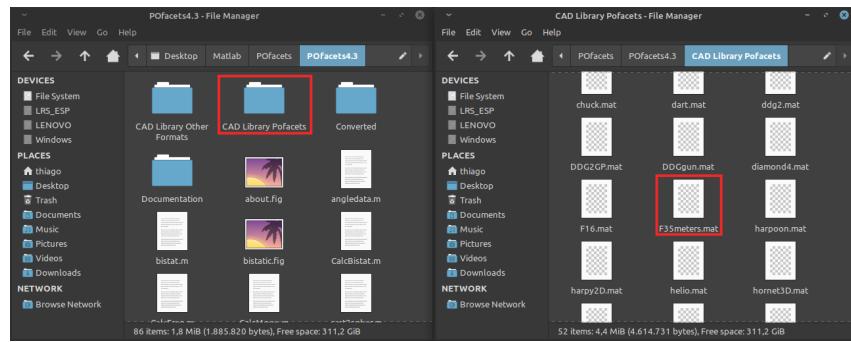
Fonte: elaboração própria

Nota-se que o usuário pode construir geometrias complexas a partir de formas simples como elipsóides, cilindros, paralelepípedos e até uma fuselagem simples. Todavia, construir geome-

trias complexas como uma aeronave completa mostra-se inviável a partir desse método.

Também existem alguns modelos que foram baixados juntamente com o programa, os quais podem ser acessados na pasta *CAD Library Pofacets*. O modelo da aeronave F-35 utilizada encontra-se nesse diretório, conforme indica a figura

Figura 9: Acessando os modelos do POfacets



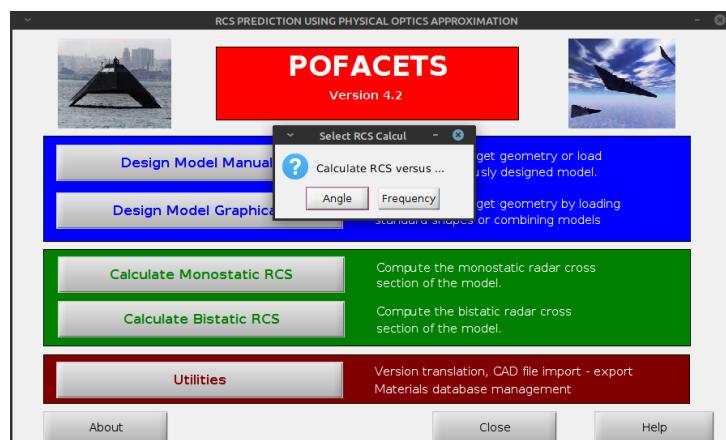
Fonte: elaboração própria

4.4 Cálculo do RCS monostático

Segundo no cálculo da assinatura de radar da aeronave F-35, o usuário pode escolher, entre os comandos em verde, se deseja calcular o RCS bi-estático ou monostático.

Ao selecionar qualquer uma das opções, podemos optar por realizar uma análise variando o ângulo de incidência da onda enviada pelo radar ou então a frequência da onda, como mostrado em [10].

Figura 10: Seleção da análise a ser realizada



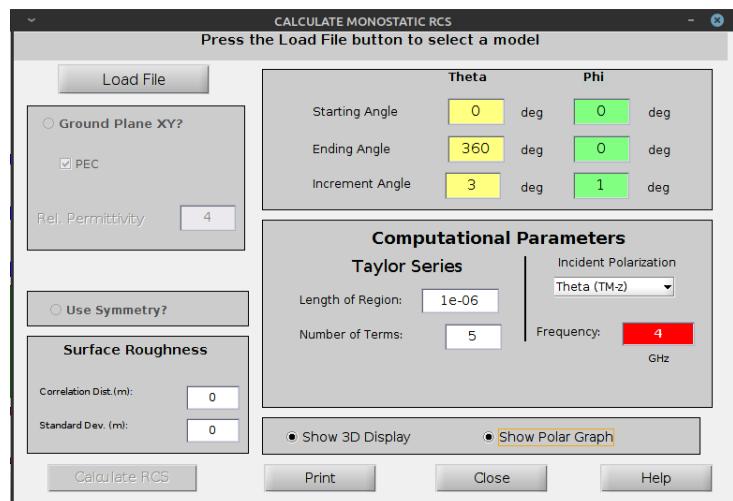
Fonte: elaboração própria

Ambas as análises geram os mesmos resultados, caso sejam dadas as mesmas condições de ângulo e frequência, e deve ser selecionada pensando em qual parâmetro é mais importante para a análise.

Seguindo com a análise, como o sensor utilizado para a parte experimental deste projeto é do tipo monostático, será feita primeiramente a análise monostática de nosso modelo.

Independente da escolha feita entre variar ângulo ou frequência, as interfaces serão muito similares. Será realizado o cálculo com uma frequência fixa de 4GHz, baseado naquilo que foi discutido na seção 3.2.

Figura 11: Aba da análise monostática



Fonte: elaboração própria

A figura [11] contém diversos parâmetros os quais podem ser variados de forma a se obter diferentes resultados ou de forma a tornar os resultados obtidos mais precisos.

A seleção dos ângulos de azimute (ϕ) e elevação (θ), assim como os intervalos, definem a(s) seção(ões) a serem analisadas pelo programa. Por experiência do usuário, recomenda-se variar θ de 0 a 360° e escolher apenas um valor de ϕ , pois ao computar diversos resultados distintos de uma única vez, os gráficos ficam poluídos. Além disso, ao calcular diversas posições o programa pode levar muito tempo para rodar completamente.

Outro parâmetro importante é definir a frequência da onda desejada.

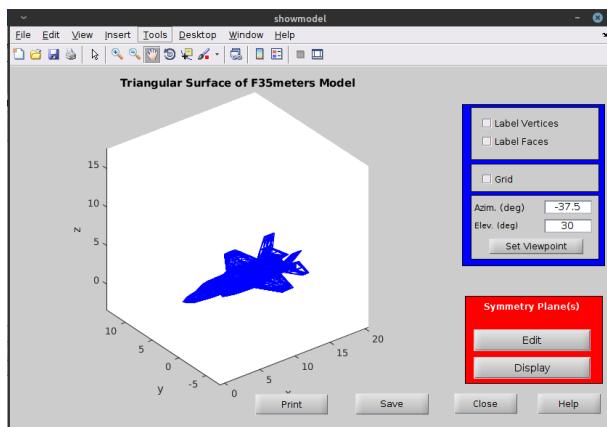
Também há a opção de variar os termos das séries de Taylor que o programa utiliza para o cálculo do RCS numericamente. Aumentar o número de termos e diminuir o tamanho da

região são opções que tornam os resultados mais precisos, mas também aumentam o tempo computacional.

Tendo definido esses parâmetros, deve-se carregar o modelo a ser analisado clicando na opção *Load File*. Será utilizado o modelo da aeronave F-35 da biblioteca do POfacets.

Após selecionar o arquivo desejado, uma nova aba surge com o modelo facetado de seu arquivo, apenas para fim de visualização, conforme mostra a figura [12].

Figura 12: Modelo facetado pronto para análise



Fonte: elaboração própria

Ao fechar a aba, a opção *Calculate RCS* indicado na figura [11] se tornará disponível para clicar, iniciando assim o cálculo do RCS do modelo selecionado.

Também ficará disponível a opção *Ground Plane XY*, a qual irá adicionar nos cálculos o efeito do solo. Esse recurso é muito útil no caso em que se está simulando aeronaves em baixas altitudes.

O solo também pode ser definido como um condutor elétrico perfeito, através do botão *PEC*. Nesse caso, deve-se definir a permissividade elétrica do solo para que seja possível realizar a simulação.

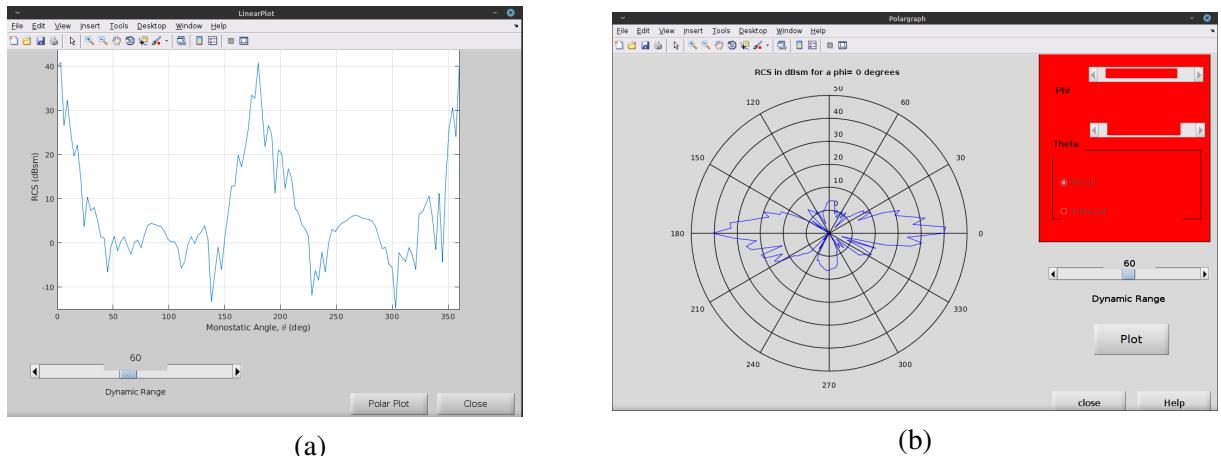
Para a visualização dos resultados, pode-se criar gráficos polares com a assinatura de radar da seção analisada e um gráfico 3D com o modelo facetado e a respectiva assinatura de radar indicada na seção selecionada pelos ângulos ϕ e θ .

Ao final do cálculo do RCS, três gráficos são gerados. Na figura [13] estão os gráficos da assinatura de radar em forma linear e polar, de maior utilidade para um usuário. Isso pois esse

gráfico permite que se extraiam valores numéricos exatos para cada ponto da seção analisada.

Estes gráficos são interativos, sendo possível alterar a escala e a seção que se está sendo analisada (variando os parâmetros θ e ϕ).

Figura 13: Gráficos de RCS monostático do F35: (a) linear; (b) polar



(a)

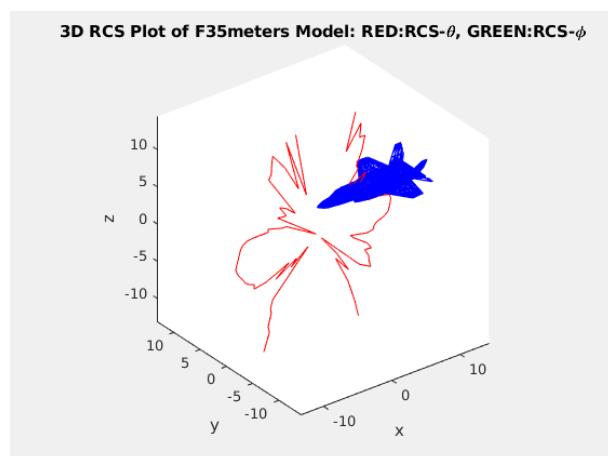
(b)

Fonte: elaboração própria

Além dos gráficos, há a opção de salvar os resultados em uma planilha ou arquivo de texto, o que permite a fácil integração dos resultados do POfacets com códigos computacionais em qualquer linguagem.

Também é gerado uma imagem que facilita a visualização do resultado, ao integrar os valores numéricos com o modelo em 3D da sua aeronave. O gráfico em questão está representado na figura [14].

Figura 14: Modelo 3D do F35 com RCS indicado na seção



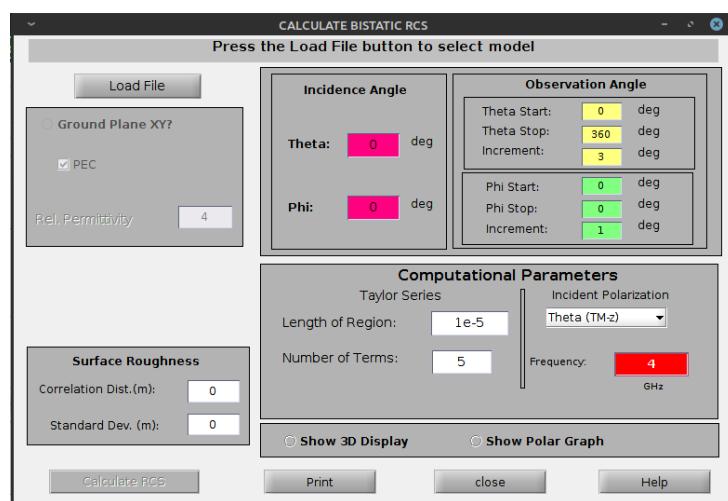
Fonte: elaboração própria

4.5 Cálculo do RCS bi-estático

Assim como para o caso monostático, ao selecionar a opção *Calculate Bistatic RCS* (vide figura [7]) pode-se selecionar se deseja variar a frequência da onda ou o ângulo durante a simulação.

Ambas as interfaces são similares, desta forma foi selecionada a opção de ângulo, conduzindo o usuário a interface mostrada na figura [15].

Figura 15: Interface da análise bi-estática



Fonte: elaboração própria

Similar à análise monostática, definem-se os ângulos da(s) seção(ões) a serem analisadas, a frequência da onda, parâmetros de rugosidade e parâmetros das séries de Taylor.

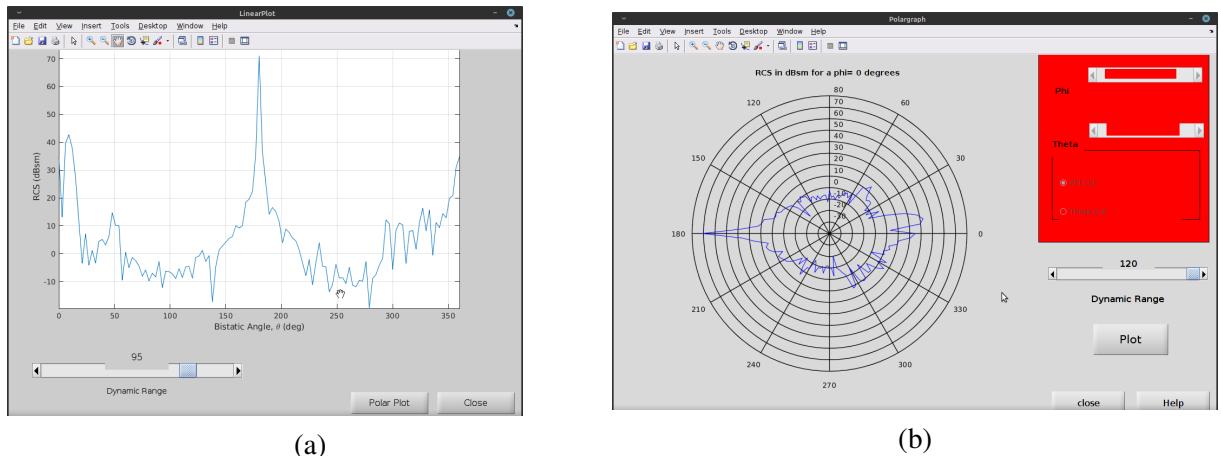
Todavia, deve-se definir agora os ângulos θ e ϕ para a onda incidente.

Para prosseguir com a simulação, importa-se um modelo utilizando *Load File*. Novamente será utilizado a aeronave F-35 disponível na biblioteca do POfacets.

Após carregar o modelo, a opção *Calculate RCS* se torna disponível para clicar, assim como os parâmetros do *Ground Plane*.

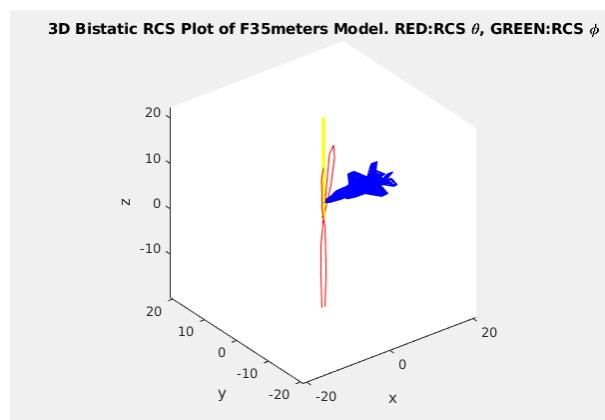
Ao fim da simulação, os mesmos três gráficos apresentados na seção 4.4 são gerados, conforme indicado nas figuras [16] e [17].

Figura 16: Gráficos de RCS bi-estático do F35: (a) linear; (b) polar



Fonte: elaboração própria

Figura 17: Modelo 3D do F35 com RCS bi-estático indicado na seção



Fonte: elaboração própria

5 APLICAÇÃO PRÁTICA E EXPERIMENTAL

De forma a abordar uma aplicação prática do RCS, foi pensado em um equipamento simples e barato, utilizando uma eletrônica acessível e de fácil aprendizado, de forma a introduzir mesmo os estudantes sem conhecimento na área de robótica para a análise experimental.

Dessa forma, optou-se por se utilizar sensores acessíveis no mercado, tal qual o LiDAR, integrado com uma placa Arduino.

5.1 Materiais

5.1.1 Eletrônica

Para a montagem da parte eletrônica do radar, foram utilizados:

- 1 placa Arduino MEGA;
- 2 motores de passo NEMA 17;
- drivers A4988 para motor de passo;
- 1 sensor LiDAR TFMini;
- protoboard;
- fontes 12V;
- jumpers.

A placa Arduino será a responsável por controlar todo o funcionamento do radar. A escolha por uma placa Arduino MEGA se deu por conta do sensor TFMini consumir muita memória dinâmica. Todavia, caso se deseje utilizar outro tipo de sensor, como um sensor ultrassônico, uma placa Arduino UNO é capaz de suportar todo o funcionamento.

Os motores de passo, ligados aos drivers, controlam o movimento em X e Y do radar.

A protoboard é utilizada para se realizar a conexão de todos os fios de forma simples, sem a necessidade de soldagem.

O código utilizado para controlar o radar encontra-se no Apêndice A, junto de comentários que explicam o funcionamento deste.

5.1.2 Construção

Para a construção da estrutura do radar, foram necessários:

- 2 fusos TR8;
- vigas de madeira;
- rolamentos 608rs;
- 2 acoplamentos flexíveis 5x8mm;
- peças feitas em impressora 3D.

Toda a construção do radar e utilização das peças está explicada na seção 5.2.

Deve-se destacar que o uso de uma impressora 3D para a montagem do radar foi escolhido por conveniência e disponibilidade. O uso de manufatura aditiva permite criar peças resistentes e precisas, e quando se dispõe de uma, seu uso é relativamente barato.

5.1.3 Softwares

Para garantir o funcionamento do radar e o tratamento dos dados de saída, foram utilizados alguns softwares:

- Arduino;
- Python;
- SolidEdge ST10;
- Ultimaker Cura;

O software Arduino foi utilizado para a elaboração do código do radar. As saídas adquiridas pelo radar são plotadas no monitor serial e salvas em arquivo de texto. Utilizando um script em Python, os dados são tratados e plotados de forma que seja permitida a visualização. O código encontra-se comentado no Apêndice B.

Além disso, foi utilizado o software de CAD SolidEdge para montar um esquema em computador do radar, de forma a evitar erros de encaixe durante a construção.

O Ultimaker Cura foi o fatiador utilizado para preparar as peças feitas em CAD para a impressão 3D.

5.2 Construção do radar

5.2.1 Modelagem em CAD

O projeto do radar se iniciou através do software SolidEdge ST10, onde foi realizada a modelagem em CAD da estrutura do radar, a qual pode ser vista nas figura [18] e [19].

Durante a modelagem, não foram adicionados a parte eletrônica e elementos como rolamentos, castanhas, parafusos e acopladores para o motor.

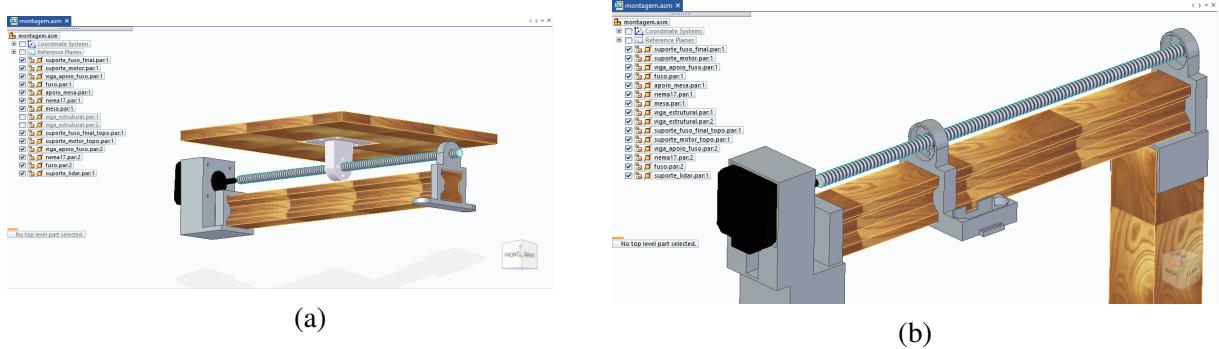
Figura 18: Montagem do radar em CAD



Fonte: elaboração própria

Além disso, após a construção do radar, mostrou-se necessário o uso de suportes para a mesa na qual o objeto deve ficar apoiado, os quais não estão representados na montagem.

Figura 19: Detalhes da montagem: (a) base; (b) topo



Fonte: elaboração própria

As cores e texturas utilizadas correspondem a cada tipo de material utilizado. As partes de madeira estão indicadas pela textura marrom. As peças impressas em impressora 3D estão representadas em cinza ou em branco. Em preto, está o motor de passo. Os fusos metálicos são representados pela textura de rosca feita pelo programa.

5.2.2 Construção do radar

A construção do radar se deu seguindo o modelo feito em computador.

As partes de madeira foram feitas com o auxílio de um marceneiro. As peças metálicas foram compradas pela internet.

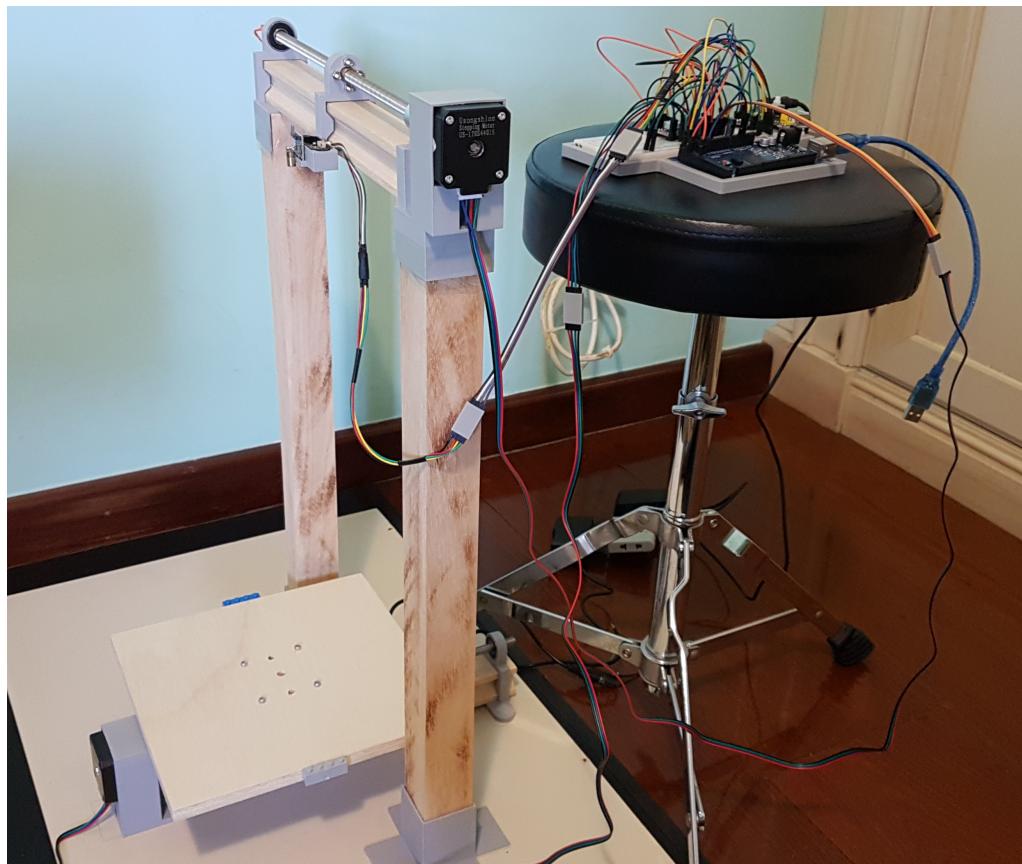
As peças feitas em impressora 3D foram impressas em PLA na impressora Creality Ender-3. A escolha do filamento se deu pelo fato da impressora ser aberta, dificultando o uso de outros tipos de material.

Com todas as partes prontas, a montagem do radar está indicada na figura [20].

Algumas pequenas mudanças foram feitas em relação ao modelo em CAD devido a instabilidade de algumas partes. Na parte inferior do radar, foram adicionados dois suportes para os pilares de madeira, os quais facilitam a fixação em um apoio fixo.

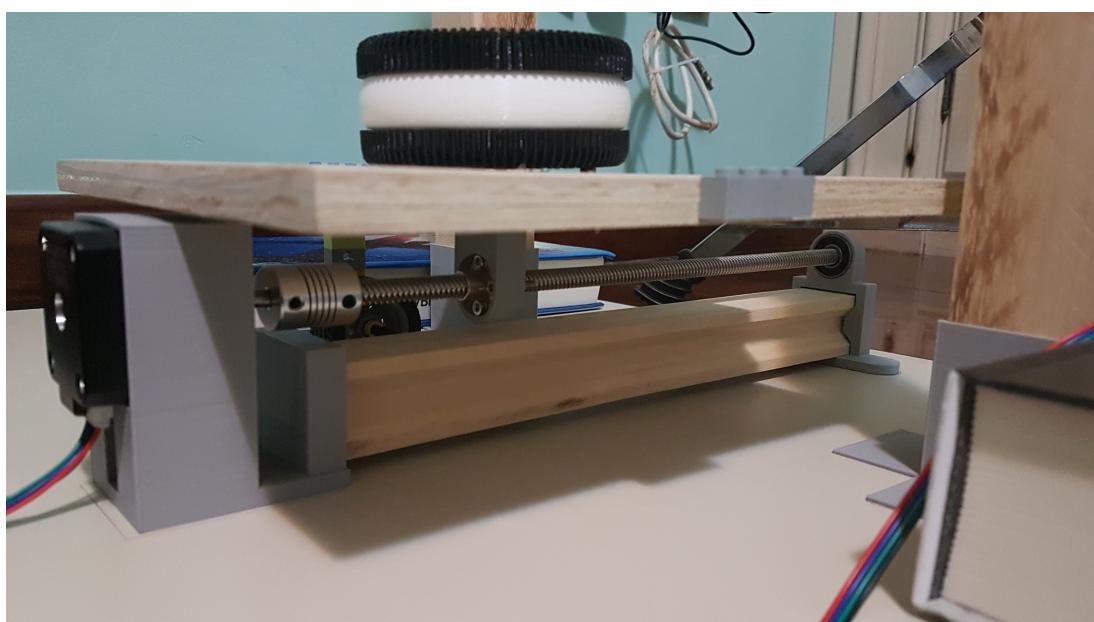
Além disso, como forma de manter a mesa perpendicular ao solo, instalou-se uma roda de suporte, a qual pode ser vista atrás do fuso na figura [21]. Nessa mesma imagem, está o primeiro objeto que foi utilizado como teste para o radar, cujos resultados estão apresentados na seção 5.4.

Figura 20: Montagem final do radar



Fonte: elaboração própria

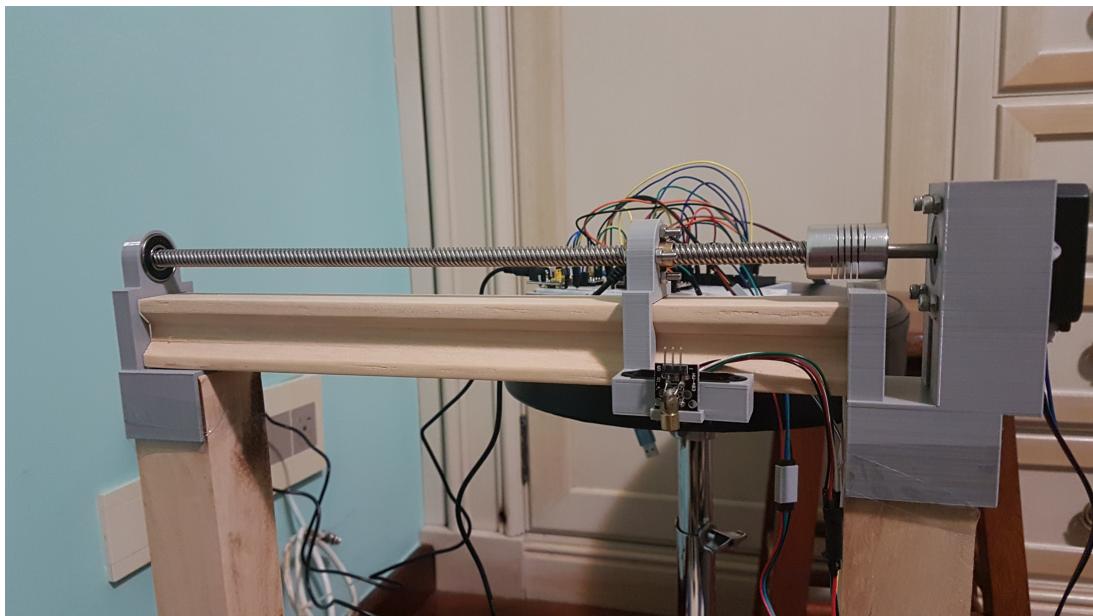
Figura 21: Detalhe da base do radar



Fonte: elaboração própria

Para a parte superior do radar, destaca-se um módulo de laser, o qual aponta para o ponto em que o LiDAR irá tomar a distância. Na figura [22], é possível visualizá-lo desconectado. Isso pois a presença do laser pode afetar as medidas realizadas pelo sensor LiDAR, dessa forma, liga-se o laser apenas no início do processo de funcionamento do radar de forma que seja possível posicioná-lo orientado para a posição desejada.

Figura 22: Detalhe do topo do radar



Fonte: elaboração própria

5.3 Montagem do circuito

A montagem do circuito para o funcionamento completo do radar envolve a ligação de inúmeros jumpers, vários se sobrepondo, em uma protoboard, tornando o esquema do conjunto completo algo pouco claro.

Dessa forma, foram elaborados esquemas individuais para o motor ligado ao driver e para o sensor LiDAR de maneira isolada. Os esquemas foram feitos ligando os fios a uma placa Arduino UNO, porém a mudança para uma placa Arduino MEGA pode ser facilmente realizada a partir do código apresentado no Apêndice A.

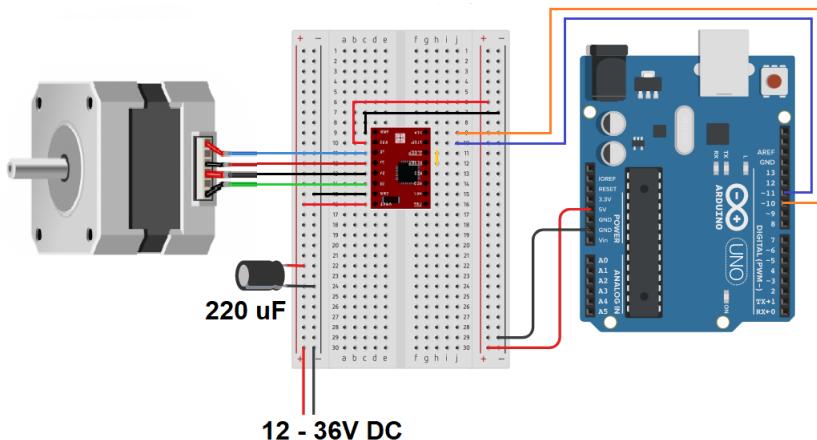
Na figura [23], deve-se conectar os fios do motor aos pinos dedicados às bobinas do driver A4988. Todavia, deve-se verificar com um multímetro quais são os pinos do motor ligados a uma mesma bobina. Para o caso do NEMA 17, apenas 4 pinos estão ligados dois a dois nas

bobinas. Sendo assim, usando o recurso de medir resistência do multímetro, pode-se identificar quais são os pares de pinos de cada bobina.

O motor deve ser alimentado com uma tensão entre 12V e 36V (isso pode variar dependendo do motor utilizado). Por isso é necessário o uso de uma fonte externa. O uso do capacitor de $220\mu F$ serve para evitar variações de tensão as quais o driver A4988 é suscetível.

Por fim conecta-se as saídas STEP e DIR do driver à duas portas digitais do Arduino de forma que seja possível controlar o movimento do motor e a sua direção.

Figura 23: Circuito para montagem do motor NEMA 17



Fonte: elaboração própria

Outro ponto importante sobre o driver A4988 é que ele possui um pequeno potenciômetro que permite regular a corrente máxima. Isso evita que o motor queime por eventuais picos de corrente que possam surgir.

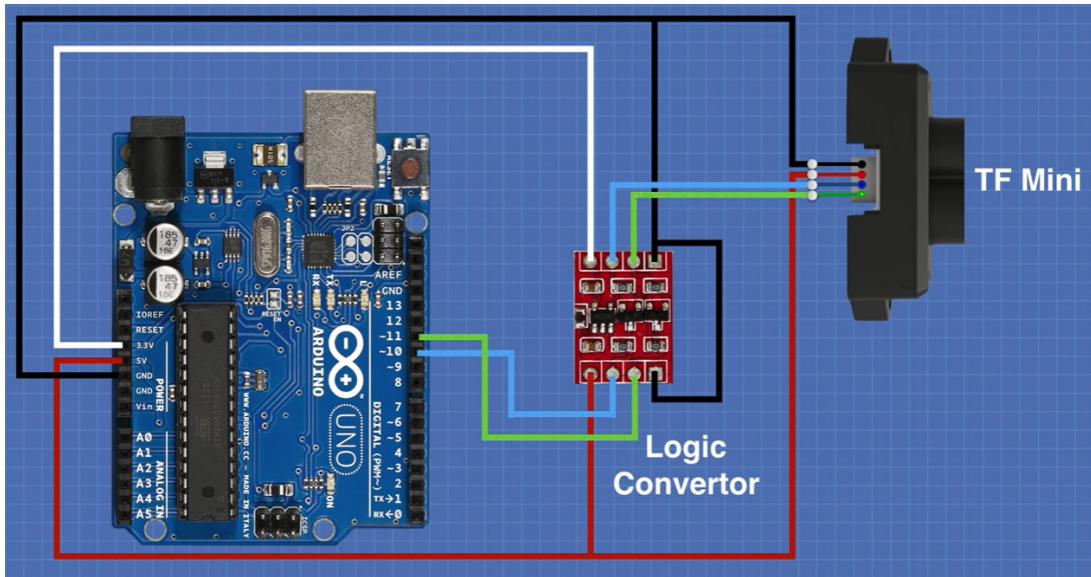
Além disso, para que o driver funcione, é necessário primeiramente energizá-lo com a tensão de 5V do Arduino e então ligar a fonte externa de 12V-36V.

Para o sensor LiDAR, o circuito foi montado com base no datasheet (2), as saídas TX e RX do dispositivo funcionam em um nível lógico de 3,3V, porém as portas digitais do Arduino funcionam com tensão de 5V. Para corrigir esse problema é utilizado um conversor de nível lógico bidirecional, o qual faz com que a informação enviada pelo sensor seja lida adequadamente pelo Arduino.

A figura [24] ilustra como deve ser feita a ligação dos fios. Por experiência, recomendo que seja utilizada uma protoboard para realizar tal ligação, tornando possível que mudanças sejam

feitas de maneira rápida sem a necessidade de realizar soldas.

Figura 24: Circuito para montagem do sensor LiDAR



Fonte: <https://www.dfrobot.com/product-1702.html>

Uma mudança que foi realizada em relação ao circuito apresentado na figura foi o uso de uma fonte externa para o uso do LiDAR. Ao invés de utilizar os pinos de 5V e 3,3V do Arduino, foi utilizada uma fonte regulável para protoboard, de forma que o LiDAR puxasse o mínimo possível de corrente do Arduino, otimizando o funcionamento dos motores.

5.4 Validação do radar

Após a construção do radar e implementação da parte eletrônica e verificações preliminares do código utilizado no Arduino foi testado o mecanismo completo utilizando um objeto simples, apenas para verificar se os resultados seriam coerentes.

O objeto selecionado foi uma caixa em formato de biscoito Oreo, conforme indicado na figura [25]. Tal escolha se deu pois o objeto é basicamente um cilindro com detalhes na superfície. Dessa forma, seria possível verificar a precisão do radar e como pequenos detalhes da superfície podem afetar uma medida.

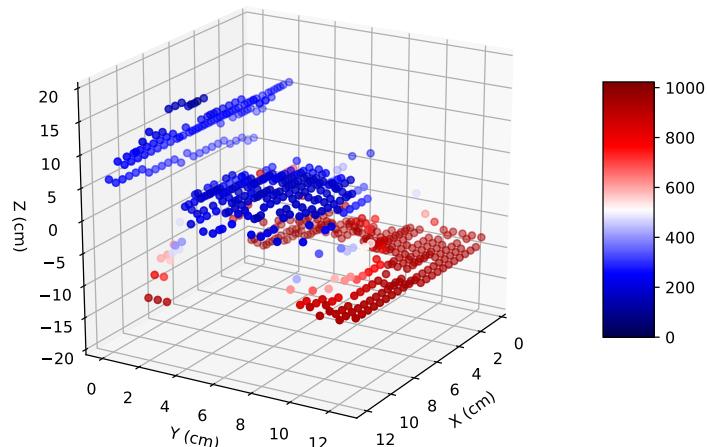
Figura 25: Objeto utilizado para a verificação



Fonte: <https://www.thingiverse.com/thing:3447657>

A partir das saídas do Arduino, utilizando o código em Python apresentado no Apêndice B, foi plotado um gráfico tridimensional de pontos, representado pela figura [26], utilizando um mapa de cores que representa a intensidade do sinal captado pelo LiDAR, variando de 0 a 1023, que é o intervalo de leitura de uma porta digital do Arduino.

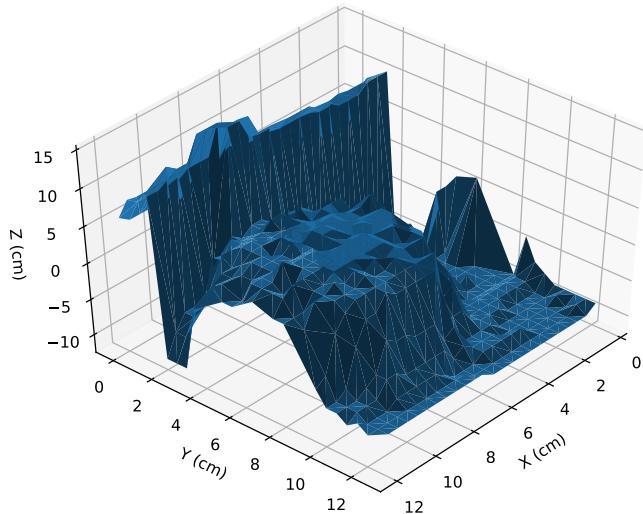
Figura 26: Visualização da assinatura de radar da verificação



Fonte: elaboração própria

Também foi gerado um gráfico de superfície para tornar a visualização do objeto mais agradável, o qual é representado pela figura [27].

Figura 27: Visualização da superfície da verificação



Fonte: elaboração própria

A partir deste gráfico, nota-se que alguns detalhes da superfície foram reconhecidos, demonstrados pelas saliências presentes na superfície circular.

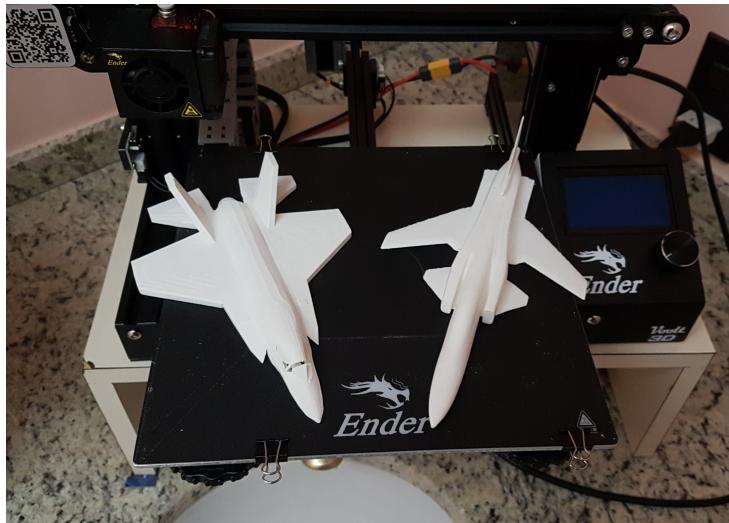
A falta de detalhes se deve à baixa resolução utilizada, visto que tratava-se apenas de uma verificação do mecanismo. Cada medida do radar distava 5mm da anterior, sendo maior que os pequenos detalhes da superfície do objeto.

Dessa forma, o radar estava finalizado e seu funcionamento verificado para se utilizar em modelos mais complexos, como aeronaves em escala.

6 RESULTADOS

Feita a validação do funcionamento do radar, prosseguiu-se para o teste da assinatura de radar de aeronaves em escala. Para isso, foram testadas duas aeronaves: *Lockheed Martin F-35 Lightning II* (à esquerda) e *Grumman X-29* (à direita). Os modelos impressos em impressora 3D estão representados na figura [28].

Figura 28: Modelos de aeronave utilizadas com o radar



Fonte: elaboração própria

A escolha dos modelos se deu levando em conta a biblioteca de modelos disponível pelo software POfacets, com o qual foram comparados os resultados.

Foram testadas duas abordagens a partir dos resultados coletados pelo Arduino:

1. Scanner: avaliação da proximidade das coordenadas coletadas com o modelo, verificando se medidas discrepantes da coordenada Z coletada correspondem a medidas menores de assinatura de radar;
2. RCS: avaliação de seções do objeto analisado a partir da intensidade do sinal registrado pelo sensor, a partir da equação (3).

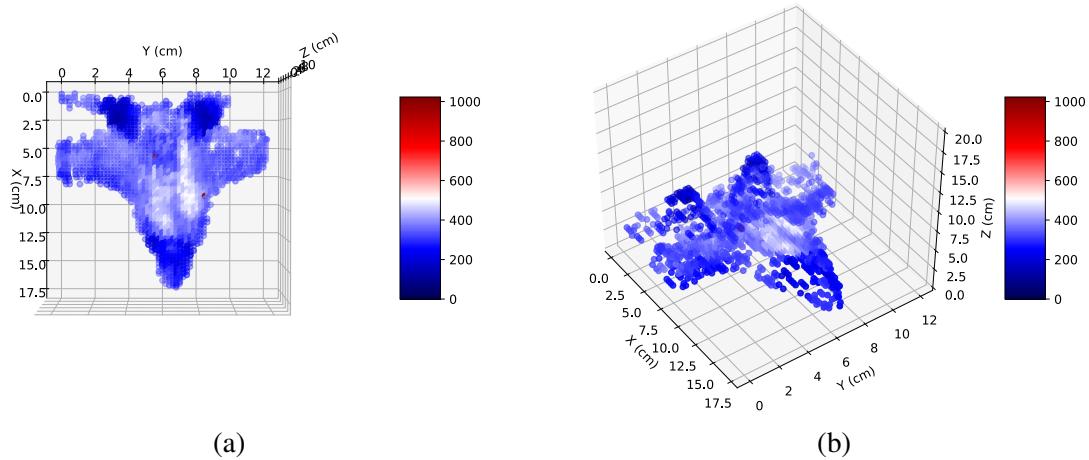
Para essa análise, foi considerado que o módulo do campo elétrico recebido é igual ao valor lido pelo Arduino e o módulo do campo elétrico enviado é igual ao valor máximo que o Arduino pode receber em pinos digitais (1023);

6.1 Resultados para a aeronave F-35

6.1.1 Análise como scanner

Para a aeronave F-35, utilizou-se uma alta densidade de pontos (2mm em X e 3mm em Y), e os resultados obtidos através do Arduino estão indicados na figura [29]. Através da análise deles, nota-se que a forma do avião está nitidamente definida. Pode-se perceber as asas, a fuselagem e a cauda em V bem definidas.

Figura 29: Resultado do scanner para a aeronave F-35: (a) vista superior; (b) vista isométrica



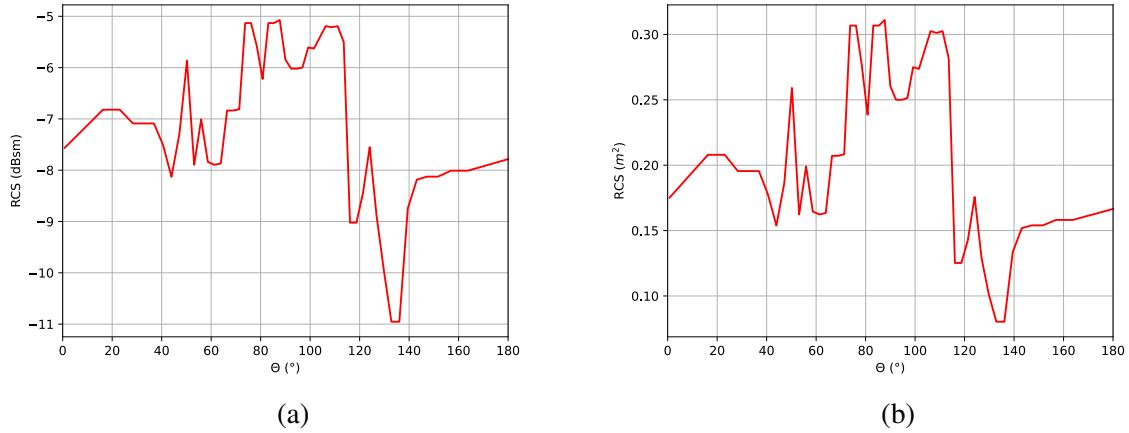
Fonte: elaboração própria

Além disso, a região da cauda, a qual esperava-se que apresentasse uma baixa assinatura de radar devido a sua geometria, foi a região da aeronave a qual o sensor captou o sinal com menor intensidade, indicando que o método de análise de RCS mostra-se promissora.

6.1.2 Análise de RCS com o sensor LiDAR

Para analisar o comportamento da assinatura de radar do modelo utilizado, escolheu-se a seção central da aeronave (plano de simetria da aeronave) e foram plotadas as curva de RCS vs. θ em m^2 e em dBsm, conforme representado na figura [30].

Figura 30: Resultado do RCS para a aeronave F-35: (a) em dBsm; (b) em m^2

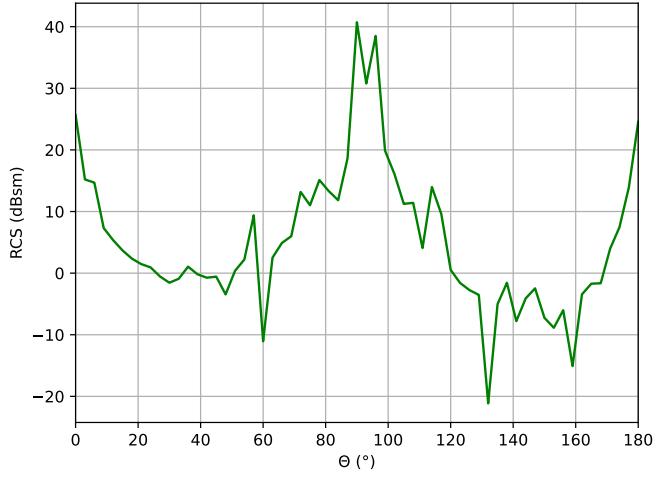


Fonte: elaboração própria

De forma a comparar os resultados do radar, a mesma seção foi simulada se utilizando o POfacets na condição de $f = 3GHz$, um valor de frequência geralmente utilizada para radares de longo alcance.

O gráfico linear gerado está representado na figura [31].

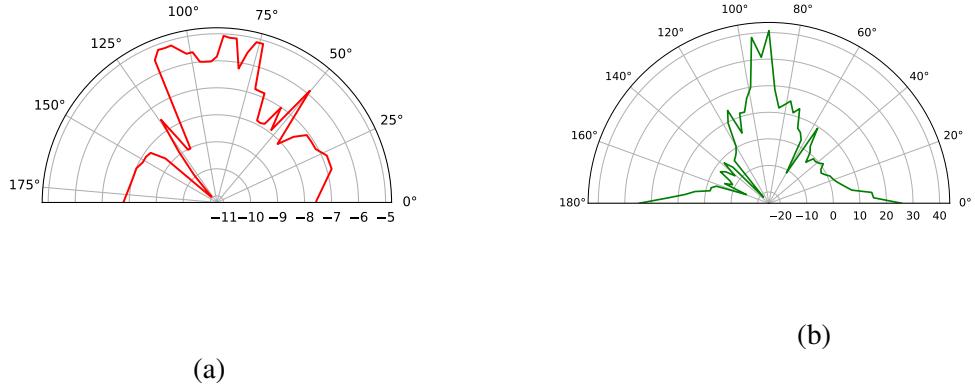
Figura 31: Resultado do RCS para a aeronave F-35 utilizando POfacets



Fonte: elaboração própria

Como é usualmente utilizado para análises de RCS, foram gerados gráficos polares tanto para o radar quanto para o POfacets de forma a se comparar os resultados, conforme ilustra a figura [32].

Figura 32: Gráficos polares para a aeronave F-35 em dBsm: (a) radar; (b) POfacets



Fonte: elaboração própria

6.2 Resultados para a aeronave X-29

6.2.1 Análise como scanner

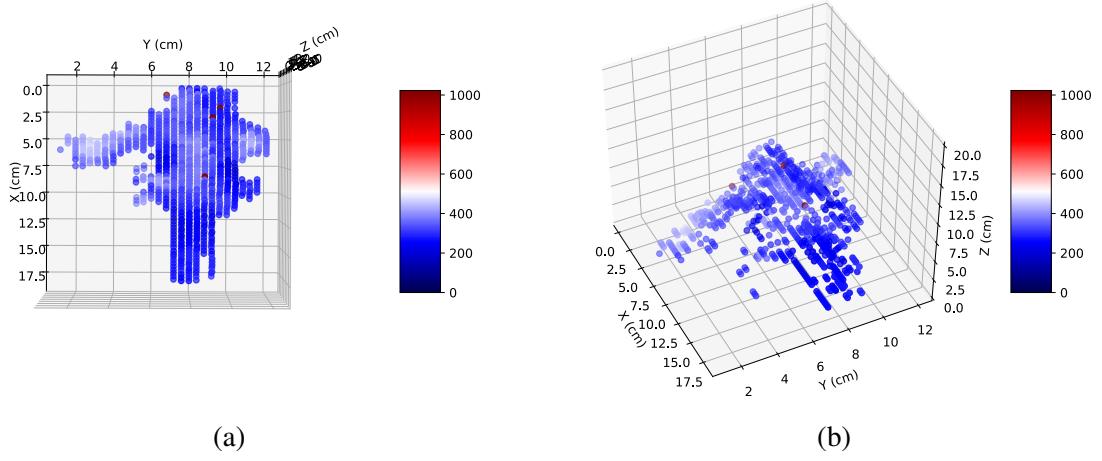
Para a aeronave X-29, utilizou-se uma baixa densidade de pontos (4mm entre medidas em X e Y), e os resultados obtidos através do Arduino estão indicados na figura [33].

Através da análise deles, nota-se que a forma do avião está bem definida, porém não tão nítida quanto a da aeronave F-35 (figura [29]). Isso se deve à baixa densidade de pontos utilizada. Mesmo assim, é possível identificar as duas asas, a fuselagem e os *cannards* mais a frente das asas.

A empenagem vertical não foi detectada, indicando que a seção que a continha não foi varrida pelo radar. A parte da semi-asa esquerda que não consta na figura [33] foi removida de forma a otimizar a leitura do gráfico.

Utilizando uma baixa densidade de pontos, o radar se mostra pouco confiável, porém ainda sendo possível definir a aeronave devido às suas características marcantes.

Figura 33: Resultado do scanner para a aeronave X-29: (a) vista superior; (b) vista isométrica

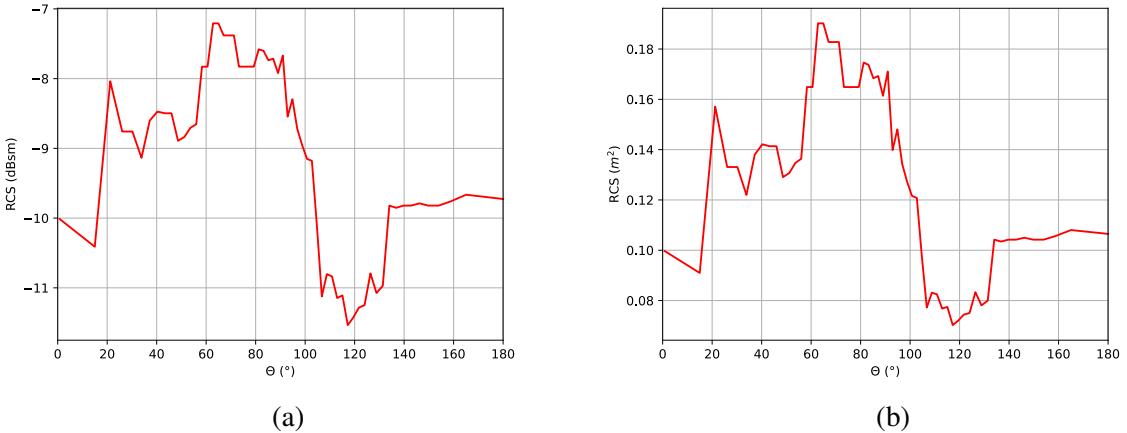


Fonte: elaboração própria

6.2.2 Análise de RCS com o sensor LiDAR

De maneira análoga ao procedimento adotado para a aeronave F-35, escolheu-se a seção central (plano de simetria) da aeronave de forma a plotar a curva de RCS vs. θ , conforme representado na figura [34].

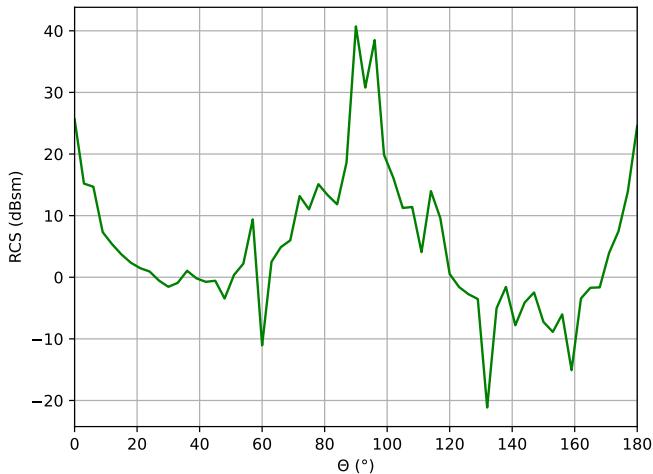
Figura 34: Resultado do RCS para a aeronave X-29: (a) em dBsm; (b) em m^2



Fonte: elaboração própria

Novamente, foi realizada uma comparação com os resultados da simulação feita através do POfacets na condição de $f = 3GHz$, através da qual foi gerado o gráfico [35].

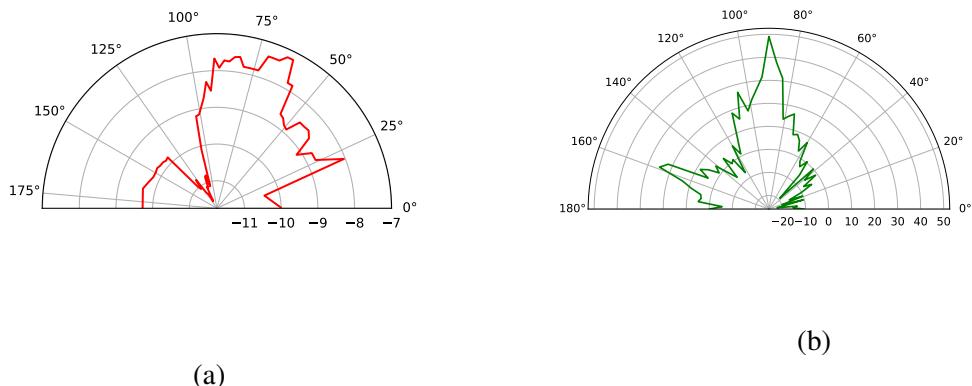
Figura 35: Resultado do RCS para a aeronave X-29 utilizando POfacets



Fonte: elaboração própria

Os gráficos polares foram plotados de forma a facilitar a comparação e visualização dos resultados, e estão indicados na figura [36].

Figura 36: Gráficos polares para a aeronave X-29 em dBsm: (a) radar; (b) POfacets



Fonte: elaboração própria

A diferença na ordem de grandeza observada entre os resultados do software POfacets e do radar se explicam pela diferença da ordem de grandeza dos diferentes modelos, conforme discutido na seção 3.8.

7 CONCLUSÃO

A partir dos resultados obtidos na seção 6, verifica-se que ambas as abordagens utilizadas mostram-se promissoras como forma de se testar a assinatura de radar de aeronaves de forma experimental e preliminar.

A abordagem na forma de scanner não traz grande informação sobre o RCS de uma aeronave, mas ao juntar as informações de coordenadas cartesianas com os valores da intensidade do sinal recebido é possível de se visualizar as regiões da aeronave em que a assinatura de radar é menor ou maior.

Além disso, o radar acabou se comportando com um scanner 3D. Como ele não foi projetado para funcionar como tal, o mecanismo se comporta como um scanner 'ineficiente', sendo possível realizar algumas mudanças de forma a otimizar essa segunda função.

Em relação à abordagem do cálculo do RCS a partir dos valores de intensidade lidos pelo Arduino, apesar dos resultados apresentados pelo experimento e pelo software não serem similares, precebe-se que aeronaves diferentes produziram assinaturas distintas, assim como era esperado.

Dessa forma, é possível criar e tabelar padrões de RCS específicos daquele sensor para cada aeronave, sendo possível se determinar de qual aeronave se trata uma leitura de RCS a partir de sensor LiDAR.

A maior limitação do uso de um LiDAR como radar é o seu baixo alcance, uma vez que seu comprimento de onda é muito pequeno se comparado com as ondas utilizadas por radares convencionais de aviação.

7.1 Sugestões para trabalhos futuros

Ao longo do desenvolvimento deste projeto, foi notado que diversas mudanças e melhorias poderiam ter sido implementadas de forma a otimizar o mecanismo de funcionamento do radar desenvolvido, enquanto os resultados revelaram novas possibilidades.

O uso de uma eletrônica mais robusta trará grandes vantagens ao funcionamento do radar, principalmente em relação ao uso de um sensor LiDAR 360° e um microcontrolador mais

preciso que um Arduino. O uso de perfis de alumínio na estrutura também traria maior estabilidade à estrutura do que as vigas de madeira utilizadas.

Além disso, pode-se desenvolver um scanner 3D a partir de um mecanismo semelhante ao desenvolvido neste trabalho, porém otimizado de forma a funcionar em menor tempo e com maior precisão.

8 REFERÊNCIAS

- [1] KNOTT, E. F.; SHAEFFER, J. F.;; TULEY, M. T. **Radar Cross Section.** 2ed. Raleigh, NC: SciTech Publishing, Inc. 2004.
- [2] TFmini Infrared Module Specification, Benewake (Beijing) Co. Ltd. Disponível em: <<https://cdn.sparkfun.com/assets/5/e/4/7/b/benewake-tfmini-datasheet.pdf>>.
- [3] David Jenn (2020). POfacets4.3 (<https://www.mathworks.com/matlabcentral/fileexchange/50602-pofacets4-3>), MATLAB Central File Exchange. Retrieved June 29, 2020.
- [4] ANTUNES, L. O. **Breve introdução a RCS e Tutorial do programa POFACETS V4.3.** 2019. Dissertação - Departamento de Engenharia Aeronáutica, Universidade de São Paulo, São Carlos, 2019.

APÊNDICE A - Código comentado do radar

O código apresentado é aquele utilizado para o controle do radar se utilizando um sensor LiDAR TFMIni e dois motores de passo NEMA 17 controlados por drivers A4988.

O avanço de 0.04mm/passo é o padrão do fuso TR-8 utilizado. O motor foi utilizado em modo full-step, com um passo de 1.8° /passo (ou seja, 200 passos representam uma volta completa e um avanço de 8mm).

Para o funcionamento adequado, deve-se posicionar o objeto em um apoio sobre a mesa, de forma que fique a pelo menos 2cm de altura, pois os valores de distância registrados pelo sensor que apresentarem diferença menor que 1cm em relação a distância do sensor até a mesa são descartados pelo código.

Também deve-se definir as dimensões do objeto nas variáveis X_corpo e Y_corpo de forma que os motores movimentem o sistema apenas o necessário, otimizando o funcionamento do mecanismo. A dimensão máxima deve ser de 200mm.

Na hora de definir as dimensões, deve-se ter em mente que o motor X é aquele que move o sensor, enquanto o motor Y é aquele que move a mesa.

O usuário também pode definir o intervalo de distância entre medidas, em milímetros, a partir das variáveis dx e dy.

A ligação no Arduino pode ser feita seguindo os esquemas apresentados nas figuras [23] e [24], porém atentando-se as portas corretas do Arduino, as quais estão definidas em código.

Por fim, todos os dados de saída são enviados ao monitor serial. Através do monitor serial, o usuário também controla quando a medida do LiDAR até a mesa deve ser feita para registrar uma medida padrão e, após isso, quando a varredura completa pode iniciar.

```
1 #include <SoftwareSerial.h>
2 #include "TFMIni.h"
3
4 // Define as variaveis do fuso
5 float avanco = 0.04; // mm/passo
```

```

6
7 // Define o tamanho do corpo
8 int X_corpo = 120; // mm
9 int X_passo = X_corpo/avanco; // em passos
10 int Y_corpo = 120; // mm
11 int Y_passo = Y_corpo/avanco; // em passos
12
13 // Define as posicoes do corpo
14 float x = 0.00; // mm
15 float dx = 5.0; // intervalo entre medidas (mm)
16 int dx_passo = dx/avanco; // em passos
17 float y = 0.00; // mm
18 float dy = 5.0; // intervalo entre medidas (mm)
19 int dy_passo = dy/avanco; // em passos
20 int z = 0;
21
22 // Variaveis do radar
23 int dist_mesa = 0.0; // distancia entre a mesa e o radar
24 int str_mesa = 0.0; // forca do sinal dessa leitura da mesa
25 SoftwareSerial mySerial(50,51);
26 TFMini tfmini;
27
28 // Definindo os pinos de cada motor
29 const int step_x = 48;
30 const int dir_x = 49;
31 const int en_x = 52;
32 const int step_y = 40;
33 const int dir_y = 41;
34 const int en_y = 53;

```

```

35
36 void setup() {
37
38     pinMode(en_x,OUTPUT);
39     digitalWrite(en_x,HIGH);
40     pinMode(en_y,OUTPUT);
41     digitalWrite(en_y,HIGH);
42
43     Serial.begin(115200);
44     while (!Serial);
45     Serial.println("Preparando os motores e o radar...");
46
47     Serial.println();
48     mySerial.begin(TFMINI_BAUDRATE);
49     tfmini.begin(&mySerial);
50
51     // Iniciar o radar
52     Serial.println("Podemos medir a distancia ate a mesa?");
53     // a pergunta acima DEVE ser respondida apenas com s
54     while (Serial.available()==0){}
55
56     delay(1000);
57     tfmini.getDistance();
58     tfmini.getRecentSignalStrength();
59     delay(250);
60     tfmini.getDistance();
61     tfmini.getRecentSignalStrength();
62     delay(250);
63     tfmini.getDistance();

```

```

64 tfmini.getRecentSignalStrength();
65 delay(250);
66 dist_mesa = tfmini.getDistance();
67 str_mesa = tfmini.getRecentSignalStrength();
68 Serial.println("Distancia ate a mesa: ");
69 Serial.println(dist_mesa);
70 Serial.println("Forca do sinal: ");
71 Serial.println(str_mesa);
72 Serial.println();
73
74 // Inicia os motores
75 pinMode(step_x,OUTPUT);
76 pinMode(dir_x,OUTPUT);
77 pinMode(step_y,OUTPUT);
78 pinMode(dir_y,OUTPUT);
79
80 Serial.println("Podemos iniciar a varredura?");
81 while (Serial.available()==2) {
82 }
83 Serial.println("Iniciando varredura de radar");
84 Serial.println("x,y,z,str"); // formato da saia
85
86 digitalWrite(dir_y,HIGH);
87
88 digitalWrite(en_x,LOW);
89 digitalWrite(en_y,LOW);
90
91 }
92

```

```

93 void loop() {
94
95     digitalWrite(dir_x,LOW); // sentido de avanco
96     for(int i=0; i < dx_passo; i++) { // andar um dx
97         digitalWrite(step_x,HIGH);
98         delayMicroseconds(500);
99         digitalWrite(step_x,LOW);
100        delayMicroseconds(500);
101    }
102    delay(500);
103    // faz a medicao da distancia
104    x = x+dx;
105
106    // precisamos descartar uma medicao
107    tfmini.getDistance();
108    tfmini.getRecentSignalStrength();
109    delay(250);
110    // mede o valor definitivo
111    uint16_t dist = tfmini.getDistance();
112    uint16_t strength = tfmini.getRecentSignalStrength();
113    z = dist_mesa-dist;
114    if (abs(z) > 1){ // imprime os valores
115        Serial.print(x);
116        Serial.print(',');
117        Serial.print(y);
118        Serial.print(',');
119        Serial.print(z);
120        Serial.print(',');
121        Serial.println(strength);

```

```

122 }
123 delay(500);
124
125 if (x >= X_corpo) {
126     digitalWrite(dir_x,HIGH); // inverte a direcao de x
127     for (int i=0; i < X_passo; i++) { // retorna ao inicio
128         digitalWrite(step_x,HIGH);
129         delayMicroseconds(500);
130         digitalWrite(step_x,LOW);
131         delayMicroseconds(500);
132     }
133     x = 0;
134     delay(500);
135     for (int j=0; j < dy_passo; j++) { // anda dy
136         digitalWrite(step_y,HIGH);
137         delayMicroseconds(500);
138         digitalWrite(step_y,LOW);
139         delayMicroseconds(500);
140     }
141     y = y + dy;
142 }
143 delay(500);
144 if ((y >= Y_corpo) && (x >= X_corpo)) { // indica o fim da
varredura
145     Serial.println("Fim da varredura de radar...");
146     while (1) {
147         Serial.println("Pode desconectar os motores");
148         delay(60000);
149     }

```

150 }

151 }

APÊNDICE B - Código para o tratamento e visualização dos dados

O código apresentado funciona devidamente ao se salvar as saídas do monitor serial do Arduino em um arquivo de texto, inclusive os dados não numéricos.

Deve-se informar ao código o nome e localização do arquivo de forma que este rode sem levantar erros.

```
1 import matplotlib.pyplot as plt
2 import numpy as np
3
4 def separador(string,sep):
5     """
6         Funcao para ler os valores do Arduino e separa-los em listas
7     """
8     lista = []
9     k = 0
10    j = k
11    for i in range(len(string)):
12        if string[i] in sep:
13            j = i
14            pal = string[k:j]
15            k = j+1
16            lista.append(float(pal))
17    return lista
18
19 file = open('saidas_arduino/x29_radar.txt','r')
20 output = file.readlines()
21 file.close()
22
23 # distancia ate a mesa e forca do sinal
24 dist_mesa = float(output[4][0:len(output[4])-1])
25 str_mesa = float(output[6][0:len(output[6])-1])
26
27 # loop para identificar a linha em que comeca a leitura
28 line_xyz = 0
```

```

29 for i in range(50):
30     if output[i] == 'x,y,z,str\n':
31         line_xyz = i+1
32         break
33
34 # definindo as informacoes para calculo do RCS
35 Y_sec = 84 # mm
36 X_rcs = []
37 theta_rcs = []
38 RCS_sec = []
39
40 # salva os dados em vetores
41 X = []
42 Y = []
43 Z = []
44 F = []
45 for i in range(line_xyz, len(output)-1):
46     linha = separador(output[i], ('\n', ',', ','))
47     if linha[2] < 10:
48         # salva as coordenadas
49         X.append(linha[0]/10)
50         Y.append(linha[1]/10)
51         Z.append(linha[2])
52         if linha[3] > 1023: # deveria ser o limite
53             linha[3] = 1023
54         F.append(linha[3])
55
56         # salva o RCS
57         if linha[1] == Y_sec:
58             X_rcs.append(linha[0]/10)
59             rcs = 4*np.pi*((dist_mesa-linha[2])**2)
60             rcs *= ((linha[3]/100)**2)/(1023**2)
61             RCS_sec.append(rcs)
62
63 X = np.array(X)

```

```

64 Y = np.array(Y)
65 Z = np.array(Z)
66 F = np.array(F)
67
68 # plota um grafico de pontos
69 fig = plt.figure()
70 ax = fig.add_subplot(111, projection='3d')
71 ax.set_zlim(-20,20)
72 surf = ax.scatter(X,Y,Z,c=F,cmap='seismic',vmin=0,vmax=1023)
73 ax.set_xlabel('X (cm)')
74 ax.set_ylabel('Y (cm)')
75 ax.set_zlabel('Z (cm)')
76 ax.set_zlim(0,20)
77 fig.colorbar(surf, shrink=0.5, aspect=5)
78 plt.show()
79
80 # plota uma superficie com o resultado
81 fig = plt.figure()
82 ax = plt.subplot(projection='3d')
83 ax.plot_trisurf(X,Y,Z)
84 ax.set_xlabel('X (cm)')
85 ax.set_ylabel('Y (cm)')
86 ax.set_zlabel('Z (cm)')
87 plt.show()
88
89 # plota o RCS de uma secao definida
90 r = (max(X_rcs)-min(X_rcs))/2
91 for i in X_rcs:
92     if i < r:
93         theta = np.arccos((r+min(X_rcs)-i)/r - 0.0001)*180/np.pi
94     else:
95         theta = 180 - np.arccos((i-min(X_rcs)-r)/r)*180/np.pi
96     theta_rcs.append(theta)
97 RCS_dbsm = [10*np.log10(i) for i in RCS_sec]
98

```

```

99 # em m2
100 img, graf = plt.subplots()
101 graf.set(xlabel = '$\Theta$ ($^\circ$)',
102             ylabel = 'RCS ($m^2$)')
103 plt.xlim(0,180)
104 plt.ylim()
105 graf.grid()
106 graf.plot(theta_rcs,RCS_sec,color='red')
107 plt.savefig('X29_rcs_m2.svg')
108 plt.show()
109
110 # em dBsm
111 img, graf = plt.subplots()
112 graf.set(xlabel = '$\Theta$ ($^\circ$)',
113             ylabel = 'RCS (dBsm)')
114 plt.xlim(0,180)
115 plt.ylim()
116 graf.grid()
117 graf.plot(theta_rcs,RCS_dbsm,color='red')
118 plt.savefig('X29_rcs_dbsm.svg')
119 plt.show()
120
121 # Graficos polares
122 theta_rcs = np.radians(theta_rcs)
123
124 # m2
125 fig = plt.figure()
126 ax = fig.add_subplot(111, polar=True)
127 c = ax.plot(theta_rcs, RCS_sec, color='red')
128 ax.set_thetamin(0)
129 ax.set_thetamax(180)
130 plt.savefig('X29_m2_polar.svg')
131 plt.show()
132
133 # dBsm

```

```
134 fig = plt.figure()
135 ax = fig.add_subplot(111, polar=True)
136 c = ax.plot(theta_rcs, RCS_dbsm, color='red')
137 ax.set_thetamin(0)
138 ax.set_thetamax(180)
139 plt.savefig('X29_dbsm_polar.svg')
140 plt.show()
```