

INSTITUTO TECNOLÓGICO DE AERONÁUTICA



Hilton Alex Soares de Lima

**DESENVOLVIMENTO DE UMA SIMULAÇÃO
MONTE CARLO PARA MÍSSIL AR-AR GUIADO
A LASER**

Trabalho de Graduação
2019

Curso de Engenharia Aeroespacial

Hilton Alex Soares de Lima

**DESENVOLVIMENTO DE UMA SIMULAÇÃO
MONTE CARLO PARA MÍSSIL AR-AR GUIADO
A LASER**

Orientador

Prof. Dr. Luís Eduardo Vergueiro Loures da Costa (ITA)

Co-orientador

Eng° Carlos Alberto de Paiva Carvalho (SIATT)

ENGENHARIA AEROESPACIAL

SÃO JOSÉ DOS CAMPOS
INSTITUTO TECNOLÓGICO DE AERONÁUTICA

2019

Dados Internacionais de Catalogação-na-Publicação (CIP)**Divisão de Informação e Documentação**

Soares de Lima, Hilton Alex
Desenvolvimento de uma simulação Monte Carlo para míssil ar-ar guiado a laser / Hilton Alex
Soares de Lima.

São José dos Campos, 2019.

Número de folhas no formato 99f.

Trabalho de Graduação – Curso de Engenharia Aeroespacial – Instituto Tecnológico de Aeronáutica, 2019. Orientador: Prof. Dr Luís Eduardo Vergueiro Loures da Costa. Co-orientador: Engº Carlos Alberto de Paiva Carvalho.

1. Mísseis. 2. Monte Carlo. 3. Simulação. 4. Engenharia aeroespacial. 5. Inserção de erros. 6. Kill probability. I. Instituto Tecnológico de Aeronáutica. II. Desenvolvimento de uma simulação Monte Carlo para míssil ar-ar guiado a laser

REFERÊNCIA BIBLIOGRÁFICA

SOARES DE LIMA, Hilton Alex; **Desenvolvimento de uma simulação Monte Carlo para míssil ar-ar guiado a laser**. 2019. 99f. Total de folhas. Trabalho de Conclusão de Curso. (Graduação em Engenharia Aeroespacial) – Instituto Tecnológico de Aeronáutica, São José dos Campos.

CESSÃO DE DIREITOS

NOME DO AUTOR: Hilton Alex Soares de Lima

TÍTULO DO TRABALHO: Desenvolvimento de uma simulação Monte Carlo para míssil ar-ar guiado a laser.

TIPO DO TRABALHO/ANO: Graduação / 2019

É concedida ao Instituto Tecnológico de Aeronáutica permissão para reproduzir cópias deste trabalho de graduação e para emprestar ou vender cópias somente para propósitos acadêmicos e científicos. O autor reserva outros direitos de publicação e nenhuma parte deste trabalho de graduação pode ser reproduzida sem a autorização do autor.

Hilton Alex Soares de Lima
Rua H9B, 504
12228-611, São José dos Campos - SP

DESENVOLVIMENTO DE UMA SIMULAÇÃO MONTE CARLO PARA MÍSSIL AR-AR GUIADO A LASER

Essa publicação foi aceita como Relatório Final de Trabalho de Graduação

Hilton Alex Soares de Lima

Autor

Prof. Dr Luís Eduardo Vergueiro Loures da Costa (ITA)

Orientador

Eng^o Carlos Alberto de Paiva Carvalho (SIATT)

Coorientador

Prof.^a Dr Cristiane Aparecida Martins

Coordenadora do Curso de Engenharia Aeroespacial

São José dos Campos, 21 de novembro de 2019.

Dedico este trabalho a todos que
colaboraram e acreditaram em mim nesta
jornada.

Agradecimentos

Agradeço primeiramente à minha família e minha esposa, Ana Carolina de Souza Alves, pelo apoio incondicional mesmo durante as longas noites de estudo.

Agradeço aos professores que tive o privilégio de ser aluno, em especial aos professores do curso de engenharia aeroespacial.

Agradeço ao professor Doutor Loures pelo empenho e preocupação não apenas comigo mais com todos os seus orientados, o Brasil precisa de mais professores com uma visão similar.

Agradeço ao Carlos Alberto por todo tempo investido e conhecimento transmitido a mim. Aproveito para agradecer a equipe da SIATT que sempre foi solícita quando precisei.

Agradeço também meus amigos da Marinha e do ITA pelo companheirismo ao longo da graduação.

"O Vencedor é apenas um sonhador que não DESISTIU".

(Autor Desconhecido)

Resumo

O objetivo desse trabalho é desenvolver uma simulação Monte Carlo para um míssil ar-ar guiado a laser com a finalidade de determinar os requisitos dos principais subsistemas do míssil sem inviabilizar a construção do mesmo e ao mesmo tempo garantir uma confiabilidade razoável (elevada kill probability), levando-se em consideração erros e ruídos oriundos das leituras de sensores e erros na execução de atuadores e na propulsão do míssil.

Foi utilizado como base para o desenvolvimento da simulação o trabalho de graduação [1] desta maneira foram implementadas algumas das sugestões sugeridas pelo trabalho citado.

A simulação levou em consideração a inserção de erros do tipo *Run-to-Run* e *in-Run* nos seguintes subsistemas: aerodinâmica, propulsão, navegação, autodiretor, atuador; e detecção da posição e velocidade iniciais do alvo. Além disso foi estudado os casos em que há limitações de leitura de dados como posição e velocidade de rastreamento do alvo pelo autodiretor e da distância de saturação do autodiretor, tais limitações modelam o que ocorre num voo real.

Abstract

The objective of this work is to develop a Monte Carlo Simulation for a laser guided air-to-air missile in order to determine the requirements of the main missile subsystems without making its construction unfeasible while at the same time ensuring a reasonable kill probability, taking into account errors and disturbances from sensor readings and errors in actuator execution and missile propulsion.

It was used as basis for the development of the simulation the undergraduate work [1] in this way some of the suggestions suggested by the mentioned work were implemented.

The simulation considered the insertion of Run-to-Run and in-Run errors in the following subsystems: aerodynamics, propulsion, navigation, self-steering, actuator; and detecting the initial position and speed of the target. In addition, we have studied cases where there are limitations of data reading such as position and target tracking speed by the auditor and the saturation distance of the auditor, such limitations model what happens in a real flight.

Lista de Figuras

FIGURA 1.1 – Exemplo de erro do tipo <i>Run-to-Run</i> ou <i>Offset</i> . Gráfico em azul é altura real do alvo e o gráfico em laranja é o valor passado pelo sensor, erro de 23 metros na leitura do altura do alvo.	21
FIGURA 1.2 – Exemplo de erro do tipo <i>in-Run</i> . Gráficos da leitura dos ângulos da linha de visada do alvo com inserção de ruídos.	22
Figura 1.3 – Método de processo de desenvolvimento do Ciclo em V (ALBERTO, 2014)....	22
FIGURA 5.1 – Diagrama de bloco completo.	31
FIGURA 5.2 – Diagrama de bloco do Autodiretor, Guiamento, Autopiloto, Vclosing (bloco que calcula a velocidade do alvo) e CheckFim (bloco responsável por encerrar a simulação quando o míssil não acerta o alvo).	32
FIGURA 5.3 – Diagrama de bloco dos Atuadores, Dinâmica 6DOF, Navegação e Alvo (bloco que calcula a posição, velocidade e aceleração do alvo).	32
FIGURA 5.4 – Diagrama de bloco do interior do bloco de Dinâmica 6DOF.	33
FIGURA 5.5 – Diagrama de bloco da aerodinâmica com variáveis aleatórias inseridas.	34
FIGURA 5.6 – Diagrama de bloco da aerodinâmica ampliado apenas onde foram inseridos as variáveis aleatórias.	34
FIGURA 5.7 – Diagrama de bloco da propulsão com variáveis aleatórias inseridas.	35
FIGURA 5.8 – Diagrama de bloco da navegação.	36
FIGURA 5.9 – Diagrama de bloco do Acelerômetro com variáveis aleatórias inseridas.	36
FIGURA 5.10 – Diagrama de bloco do Giroscópio com variáveis aleatórias inseridas.	37
FIGURA 5.11 – Diagrama de bloco do Autodiretor com variáveis aleatórias inseridas.	38
FIGURA 5.12 – Diagrama de bloco dos Atuadores com variáveis aleatórias inseridas.	38

FIGURA 5.13 – Diagrama de bloco da cinemática do alvo virtual com variáveis aleatórias inseridas.	39
FIGURA 6.1 – Gráficos das distâncias mínimas versus distância de saturação e ganho aceleração lateral.	42
FIGURA 6.2 - Gráficos das distâncias máximas versus distância de saturação e ganho aceleração lateral.	42
FIGURA 6.3 – Envelope de voo para uma altura de 500 metros e ganho de aceleração lateral de 1.1, alvo não realiza manobra evasiva. Alvo movendo-se com velocidade 120 m/s, na direção 180°.	44
FIGURA 6.4 – Envelope de voo para uma altura de 1500 metros, ganho de aceleração lateral de 1.1, alvo não realiza manobra evasiva. Alvo movendo-se com velocidade 120 m/s, na direção 180°.	44
FIGURA 6.5 – Envelope de voo para uma altura de 3000 metros e ganho de aceleração lateral de 1.1, alvo não realiza manobra evasiva. Alvo movendo-se com velocidade 120 m/s, na direção 180°.	45
FIGURA 6.6 – Envelope de voo para uma altura de 6000 metros e ganho de aceleração lateral de 1.1, alvo não realiza manobra evasiva. Alvo movendo-se com velocidade 120 m/s, na direção 180°.	45
FIGURA 6.7 – Envelope de voo para uma altura de 10000 metros e ganho de aceleração lateral de 1.1, alvo não realiza manobra evasiva. Alvo movendo-se com velocidade 120 m/s, na direção 180°.	46
FIGURA 6.8 – Envelope de voo para uma altura de 500 metros, ganho de aceleração lateral de 1.1 e alvo manobrando evasivamente em direção ao solo. Alvo movendo-se com velocidade 120 m/s, na direção 180°.	47

FIGURA 6.9 – Envelope de voo para uma altura de 1500 metros, ganho de aceleração lateral de 1.1 e alvo manobrando evasivamente em direção ao solo. Alvo movendo-se com velocidade 120 m/s, na direção 180°.....	47
FIGURA 6.10 – Envelope de voo para uma altura de 3000 metros, ganho de aceleração lateral de 1.1 e alvo manobrando evasivamente em direção ao solo. Alvo movendo-se com velocidade 120 m/s, na direção 180°.....	48
FIGURA 6.11 – Envelope de voo para uma altura de 6000 metros, ganho de aceleração lateral de 1.1 e alvo manobrando evasivamente em direção ao solo. Alvo movendo-se com velocidade 120 m/s, na direção 180°.....	48
FIGURA 6.12 – Envelope de voo para uma altura de 10000 metros, ganho de aceleração lateral de 1.1 e alvo manobrando evasivamente em direção ao solo. Alvo movendo-se com velocidade 120 m/s, na direção 180°.....	49
FIGURA 6.13 – Envelope de voo para uma altura de 3000 metros, ganho de aceleração lateral de 1.5 e alvo manobrando evasivamente em direção ao solo. Alvo movendo-se com velocidade 120 m/s, na direção 180°.....	50
FIGURA 6.14 – Envelope de voo para uma altura de 6000 metros, ganho de aceleração lateral de 1.5 e alvo manobrando evasivamente em direção ao solo. Alvo movendo-se com velocidade 120 m/s, na direção 180°.....	50
FIGURA 6.15 – Envelope de voo para uma altura de 10000 metros, ganho de aceleração lateral de 1.5 e alvo manobrando evasivamente em direção ao solo. Alvo movendo-se com velocidade 120 m/s, na direção 180°.....	51
FIGURA 6.16 – Envelope de voo para uma altura de 10000 metros, ganho de aceleração lateral de 2.5 e alvo manobrando evasivamente em direção ao solo. Alvo movendo-se com velocidade 120 m/s, na direção 180°.....	52

Lista de Tabelas

TABELA 4.1 – Valores das variáveis aleatórias do bloco da Aerodinâmica.....	26
TABELA 4.2 – Valores das variáveis aleatórias do bloco da Propulsão.	27
TABELA 4.3 – Valores das variáveis aleatórias do bloco da Navegação.	28
TABELA 4.4 – Valores das variáveis aleatórias do bloco do Autodiretor.	29
TABELA 4.5 – Valores das variáveis aleatórias do bloco dos atuadores.	29
TABELA 4.6 – Valores variáveis aleatórias do bloco de detecção da posição e velocidade iniciais do Alvo.....	30
TABELA 6.1 – Valores dos principais parâmetros do autodiretor e o respectivo nível de robustez.....	40
TABELA 7.1 – Simulação Monte Carlo para distância de lançamento de 1000, 5000 e 10000 metros, alvo aproximando-se do lançador e realizando manobra evasiva.	54
TABELA 7.2 – Simulação Monte Carlo para distância de lançamento de 1000, 5000 e 10000 metros, alvo afastando-se do lançador e realizando manobra evasiva.	57
TABELA 7.3 – Simulação Monte Carlo para distância de lançamento de 1000, 5000 e 10000 metros, alvo cruzando a proa do lançador perpendicularmente e realizando manobra evasiva.	60
TABELA 7.4 – Valores otimizados das variáveis aleatórias do conjunto de erros do bloco da Navegação.	63
TABELA 7.5 – Valor final da vetor de fator multiplicativo das incertezas de cada bloco de erros dos subsistemas.....	65
TABELA 7.6 – Simulação Monte Carlo para distância de lançamento de 1000, 5000 e 10000 metros, alvo cruzando a proa do lançador perpendicularmente e realizando manobra evasiva.	65
TABELA 8.1 – Valores dos parâmetros do subsistema da Aerodinâmica.....	69

TABELA 8.2 – Valores dos parâmetros do subsistema de Propulsão.	70
TABELA 8.3 – Valores dos parâmetros do subsistema de Navegação.....	70
TABELA 8.4 – Valores finais dos parâmetros do subsistema do Autodiretor.....	71
TABELA 8.5 – Valores dos parâmetros do subsistema do Autodiretor.	71
TABELA 8.6 – Valores dos parâmetros do subsistema dos atuadores.	72
TABELA 8.7 – Valores dos parâmetros do subsistema de detecção da posição e velocidade iniciais do Alvo.....	72
TABELA 8.8 – Kill Probability (Chnace de acerto) para um míssil ar-ar guiado a laser com os parâmetros e desvios padrões citados lançado de uma distância de 1000, 5000 e 10000 metros, alvo cruzando a proa do lançador perpendicularmente e realizando manobra evasiva.....	72

Lista de Abreviaturas e Siglas

6DOF	6 Graus de Liberdade
AD	Autodiretor
AERO	Aerodinâmica
AT	Atuadores
CG	Centro de Gravidade
CONOPS	Conceito de Operações
KAcIn	Ganho de Aceleração Lateral
NAV	Navegação
PROP	Propulsão

Lista de Símbolos

sen	função seno
cos	função cosseno
ω	velocidade angular
θ	ângulo da linha de visada no plano xz
ψ	ângulo da linha de visada no plano xy

Sumário

1	INTRODUÇÃO	19
1.1	Motivação	19
1.2	Hipóteses.....	19
1.3	Objetivo	20
1.4	Metodologia	20
2	REVISÃO BIBLIOGRÁFICA	24
3	CONCEITO DE OPERAÇÕES E ARQUITETURA OPERACIONAL DO SISTEMA	25
4	DEFINIÇÃO DAS VARIÁVEIS ALEATÓRIAS PARA A SIMULAÇÃO.....	26
4.1	Variáveis aleatórias da Aerodinâmica	26
4.2	Variáveis aleatórias da Propulsão.....	27
4.3	Variáveis aleatórias da Navegação.....	27
4.4	Variáveis aleatórias do Autodiretor	28
4.5	Variáveis aleatórias dos Atuadores.....	29
4.6	Variáveis aleatórias na detecção da posição e velocidade iniciais do Alvo.....	30
5	IMPLEMENTAÇÃO DAS VARIÁVEIS ALEATÓRIAS NA SIMULAÇÃO COM	
6	GRAUS DE LIBERDADE	31
5.1	Aerodinâmica	33
5.2	Propulsão	35
5.3	Navegação.....	35
5.4	Autodiretor	37
5.5	Atuadores	38
5.6	Posição e Velocidade do Alvo	39
6	DEFINIÇÃO DOS PARÂMETROS INICIAIS DA SIMULAÇÃO	40
6.1	Envelopes de Voo	43
6.1.1	Envelopes de voo considerando que o alvo não realiza manobra evasiva	43
6.1.2	Envelopes de voo considerando que o alvo realiza manobra de evasão	46
7	TESTE DAS VARIÁVEIS ALEATÓRIAS IMPLEMENTADAS	53
7.1	Simulação Monte Carlo.....	53
7.1.1	Simulação Monte Carlo para alvo aproximando-se do lançador.....	54

7.1.2	Simulação Monte Carlo para alvo afastando-se do lançador	56
7.1.3	Simulação Monte Carlo para alvo cruzando a proa do lançador	59
7.2	Identificação das Variáveis Aleatórios Mais Sensíveis/Críticas	62
7.2.1	Subsistema mais crítico: Navegação	62
7.2.2	Testes dos demais subsistemas	64
7.3	Teste Final das Variáveis Aleatórias.....	65
8	DEFINIÇÃO DOS REQUISITOS DOS PRINCIPAIS SUBSISTEMAS.....	69
8.1	Parâmetros do Subsistema da Aerodinâmica	69
8.2	Parâmetros do Subsistema de Propulsão	69
8.3	Parâmetros do Subsistema de Navegação	70
8.4	Parâmetros do Subsistema do Autodiretor	71
8.5	Parâmetros do Subsistema dos Atuadores	71
8.6	Variáveis aleatórias na detecção da posição e velocidade iniciais do Alvo.....	72
9	CONCLUSÃO.....	73
	REFERÊNCIAS.....	74
	APÊNDICE A – EXEMPLO DE APÊNDICE.....ERRO! INDICADOR NÃO DEFINIDO.	
A.1	Exemplo de Subseção do Apêndice A	Erro! Indicador não definido.

1 Introdução

1.1 Motivação

Uma das tarefas mais importantes de engenharia de concepção de mísseis é determinar requisitos dos principais subsistemas do míssil sem inviabilizar a construção do mesmo e ao mesmo tempo garantir uma elevada kill probability (chance de acerto), levando-se em consideração erros e ruídos oriundos das leituras dos sensores e erros na execução dos comandos nas superfícies de controle e na desvios de empuxo na propulsão do míssil.

Tais erros podem levar a desestabilização do míssil e, conseqüentemente, derrubar o míssil antes que atinja o alvo. Os principais problemas observados em sensores embarcados são:

- 1) Erros em degrau (o sensor já apresenta um valor de desvio em relação ao valor real); e
- 2) Suscetibilidade a ruídos.

Enquanto que nas superfícies de controle o principal problema é a diferença entre a ação comandada e a executada.

A solução de compromisso de um míssil sempre é tão complexa quanto se queira, porém, os recursos como em qualquer outro projeto são finitos e, portanto, é prudente não gerar requisitos superdimensionados de forma a tornar o míssil caro demais. O *stakeholder*, supostamente, não estará inclinado a pagar mais por um projeto similar já existente.

1.2 Hipóteses

A hipótese fundamental deste trabalho é a existência de uma combinação de requisitos dos principais subsistemas do míssil que otimizem ao máximo o voo e a probabilidade de acerto do míssil (kill probability) considerando flutuações (perturbações) nas leituras de sensores embarcados, montagem e execução das superfícies de controle do míssil, entre outros mostrados que serão detalhados posteriormente.

Para tal será utilizado inicialmente valores de flutuações reais nas leituras dos sensores, ou seja, hipoteticamente será considerado que não haja flutuações maiores do que as já observadas em voos reais realizados e valores reais de erros nos comandos para a superfície de controle.

1.3 Objetivo

Este trabalho possui o objetivo de desenvolver uma simulação Monte Carlo para um míssil ar-ar guiado a laser e como resultado definir os requisitos dos principais subsistemas do míssil:

- a) Autodiretor (responsável pelo rastreamento do alvo);
- b) Atuador (responsável pela deflexão das superfícies de controle);
- c) Aerodinâmica (responsável pela determinação do movimento do míssil);
- d) Propulsão (responsável pelo empuxo do míssil);
- e) Acelerômetro e Giroscópio (responsáveis pela navegação); e
- f) Dados de inicialização como posição e velocidade estimada do alvo.

1.4 Metodologia

Os requisitos dos subsistemas serão determinados a partir da análise dos resultados das simulações variando-se os parâmetros dos sensores e atuadores que serão estudados na simulação Monte Carlo, utilizando como plataforma de simulação computacional o software *MATLAB® R2018b*. Os parâmetros serão agrupados de acordo com o subsistema a qual pertencem. Ou seja, erros e ruídos dos acelerômetros estarão agrupados no mesmo conjunto, chamado de bloco de erros. Essa metodologia foi definida desta maneira por duas razões:

- 1) O tempo de processamento necessário para fazer uma simulação com 6 graus de liberdade inserindo-se os erros nas leituras de todos os sensores de uma única vez. Levaria muitas horas, talvez dias dependendo da discretização adotada; e
- 2) Inserir erros na leitura de muitos sensores e atuadores de uma única vez impossibilitaria a correta observação de qual oscilação afeta mais a estabilidade do míssil.

Portanto, a inserção dos erros será feita através de uma combinação de blocos de erros seguindo uma ordem binária, enquanto alguns blocos estão ligados outros não estão, de forma que possa ser possível verificar qual subsistema é mais suscetível à erros, e posteriormente faz-se a simulação com apenas o bloco de erros que demonstrou ser mais o mais crítico para a estabilidade, isolando os erros um a um e encontrar qual(is) dos erros e ruídos desse subsistema está(ão) interferindo na kill probability desejada.

Para simular a instabilidade serão inseridos dois tipos de perturbações que representem os erros, incertezas e ruídos observados num voo real, são eles:

- Erro do tipo *Run-to-Run*, erro gerado aleatoriamente a parte de uma distribuição predeterminada (normal ou uniforme) no início de cada simulação e permanece constante durante a simulação onde foi gerada;
- Erro do tipo *In-Run*, erro gerado aleatoriamente a partir de uma distribuição normal com média e incertas predeterminadas e seu valor altera-se a cada instante da simulação.

O erro do tipo *Run-to-Run* é utilizado para simular erros de alinhamentos de sensores, de posicionamento do CG, atuação das superfícies de controle, entre outros, este erro é mais conhecido como erro de offset, conforme Figura 1.1.

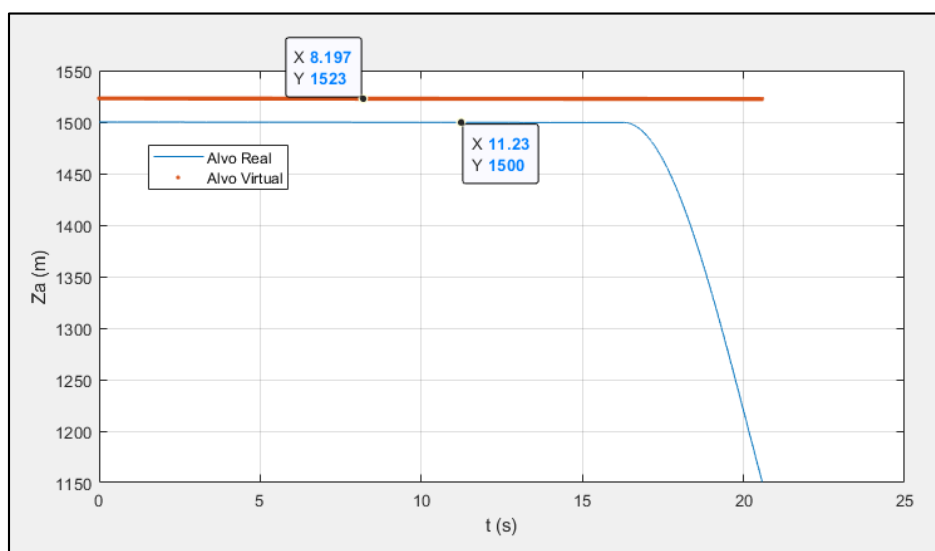


FIGURA 1.1 – Exemplo de erro do tipo *Run-to-Run* ou *Offset*. Gráfico em azul é altura real do alvo e o gráfico em laranja é o valor passado pelo sensor, erro de 23 metros na leitura do altura do alvo.

Enquanto que o erro do tipo *In-Run* é utilizado para simular ruídos na leitura dos sensores e na atuação de superfícies de controle, tipicamente chamado de ruído, conforme na Figura 1.2.

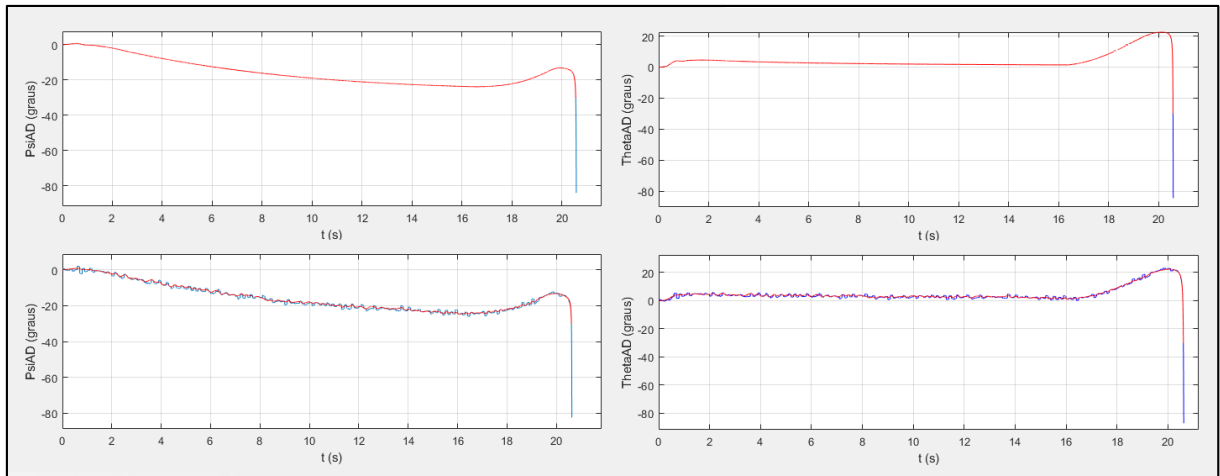


FIGURA 1.2 – Exemplo de erro do tipo *in-Run*. Gráficos da leitura dos ângulos da linha de visada do alvo com inserção de ruídos.

Para verificação dos passos deste trabalho será utilizado no processo de desenvolvimento o método do ciclo em V, conforme mostrado na Figura1.3.

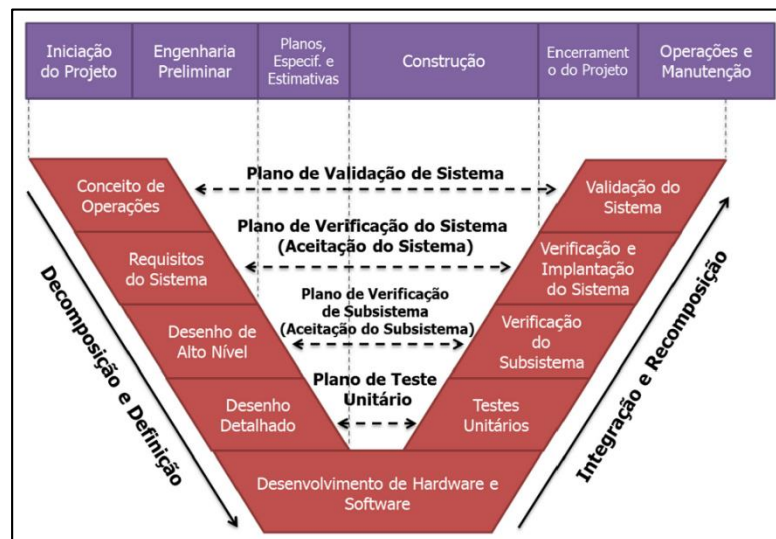


Figura 1.3 – Método de processo de desenvolvimento do Ciclo em V (ALBERTO, 2014).

Ou seja, após a realização de cada passo em direção à definição dos requisitos dos subsistemas serão realizados testes, verificação e validação para certificar-se que os valores apresentados atendem aos requisitos de mais alto nível (extremo esquerdo do V) que é os requisitos do CONOPS (Conceito de Operações).

Este trabalho será dividido nos seguintes capítulos para que haja uma melhor compreensão do passo a passo adotado. Haja vista que na concepção de um míssil existem milhares de

variáveis aleatórias e, portanto, a orientação de um especialista na área (papel esse exercido pelo co-orientador deste trabalho) é requisitada de forma a tornar os resultados relevantes e que realmente representem um sistema real:

- Revisão Bibliográfica.
- Conceito de Operações e Arquitetura Operacional do Sistema.
- Definição das Variáveis Aleatórias.
- Implementação das Variáveis Aleatórias na Simulação com 6 Graus de Liberdade.
- Definição dos Parâmetros Iniciais da Simulação.
- Teste das Variáveis Aleatórias Implementadas
- Definição dos Requisitos dos Principais Subsistemas
- Conclusão.

2 Revisão Bibliográfica

Na concepção de míssil modernos não há tanta bibliografia disponível por razões óbvias. Grandes empresas de defesa guardam a sete chaves os principais requisitos de sistema de mísseis dos seus armamentos.

No entanto para este trabalho foi utilizado como base teórica o trabalho de conclusão de curso de graduação [1].

O trabalho [1] abordou todos aspectos aerodinâmicos necessários para uma simulação completo de míssil ar-ar guiado a laser. No entanto os resultados obtidos levam em consideração que todos os sensores embarcados e superfícies de controle são ideais, ou seja, não apresentam erros de leitura, montagem ou execução. Os resultados apresentados no trabalho são claros e oportunos de modo que é possível considerar todos os parâmetros já cálculos em [1], como determinação das dimensões do míssil, cálculo dos coeficientes aerodinâmicos e ganhos das malhas de controle. Portanto todos parâmetros básicos para se iniciar as simulações já estão fornecidas para o tipo de míssil que este trabalho abordará.

Para as simulações Monte Carlo será analisado quais das variáveis que são mais sensíveis numa simulação Monte Carlo.

3 Conceito de Operações e Arquitetura Operacional do Sistema

De forma a garantir uma continuidade aceitável do trabalho [1] foram mantidas iguais todas as características do conceito de operações e arquitetura operacional do sistema como cenário de missão, diagrama de contexto e demais características do míssil concebido excetuando-se a direção inicial de voo do alvo e das manobras evasivas adotadas pelo alvo. Ainda em [1], foi adotada a melhor condição possível para manter uma kill probability que é quando o alvo está aproximando-se em linha reta do lançador sem efetuar manobras evasivas.

Porém neste trabalho foi adotado uma configuração mais severa para o acerto do alvo que é quando este cruza a proa do lançador perpendicularmente (fazendo um ângulo de 90°) efetuando manobra evasiva descendente quando o míssil está a uma distância menor que 1000 metros. A adoção dessa configuração será justificado no decorrer deste trabalho.

4 Definição das Variáveis Aleatórias para a Simulação

Para a definição das variáveis aleatórias relevantes para o sistema estudado é de suma importância a experiência de um especialista no assunto portanto foi realizado uma pesquisa de campo com o Eng. Carlos Alberto para conhecer os principais erros observados num voo real.

Após tal estudo foram levantados 6 conjuntos de variáveis aleatórias que representam as perturbações reais de um míssil do tipo ar-ar com seus respectivos nomes (nomenclatura utilizado no código e diagramas de blocos do *MATLAB®*), tipo de unidade (metros, segundos, etc.), tipo de erro (*Run-to-Run* ou *in-Run*), médias e desvios padrão da distribuição de cada subsistema (para a realização das simulações foi utilizado a distribuição normal).

4.1 Variáveis aleatórias da Aerodinâmica

Abrange os erros referente aos cálculos de forças e momentos que representam as interações aerodinâmicas que ocorrem durante o voo do míssil. Na tabela 4.1 observamos tais variáveis.

TABELA 4.1 – Valores das variáveis aleatórias do bloco da Aerodinâmica.

Variável (nomenclatura) / Unidade	Tipo de Erro	Média da Distribuição	Desvio Padrão
Força em X (erro_Fx) / Porcentagem %	<i>Run-to-Run</i>	100	$\frac{10}{3}$
Força em Y (erro_Fy) / Porcentagem %	<i>Run-to-Run</i>	100	$\frac{10}{3}$
Força em Z (erro_Fz) / Porcentagem %	<i>Run-to-Run</i>	100	$\frac{10}{3}$
Momento L (erro_L) / Porcentagem %	<i>Run-to-Run</i>	100	5
Momento M (erro_M) / Porcentagem %	<i>Run-to-Run</i>	100	5
Momento N (erro_N) / Porcentagem %	<i>Run-to-Run</i>	100	5

4.2 Variáveis aleatórias da Propulsão

Abrange os erros referente à confiabilidade das características do motor do míssil tal como força propulsiva, desvio do empuxo na tubeira, massa do míssil e do propelente e na definição do CG do míssil. Na tabela 4.2 observamos tais variáveis.

TABELA 4.2 – Valores das variáveis aleatórias do bloco da Propulsão.

Erro / Unidade	Tipo de Erro	Média da Distribuição	Desvio Padrão
Força Propulsiva (erro_Fprop) / Porcentagem %	<i>Run-to-Run</i>	100	$\frac{10}{3}$
Desvio de Empuxo em Y (desvio_FpropY) / graus °	<i>Run-to-Run</i>	0	0.1
Desvio de Empuxo em Z (desvio_FpropZ) / graus °	<i>Run-to-Run</i>	0	0.1
Massa Final do Míssil (erro_Mf) / Porcentagem %	<i>Run-to-Run</i>	100	1
Massa de Propelente (erro_Mprop) / Porcentagem %	<i>Run-to-Run</i>	100	1
CG em X (erro_CGx) / metro m	<i>Run-to-Run</i>	0	0.0147
CG em Y (erro_CGy) / metro m	<i>Run-to-Run</i>	0	0.0147
CG em Z (erro_CGz) / metro m	<i>Run-to-Run</i>	0	0.0147

4.3 Variáveis aleatórias da Navegação

Abrange os erros referente à confiabilidade dos sensores utilizados no subsistema de navegação do míssil como o erro e ruído do acelerômetro e giroscópio gerados durante o voo. Tais ruídos quando não levados em consideração podem gerar uma condição de instabilidade do voo. Na tabela 4.3 observamos tais variáveis.

TABELA 4.3 – Valores das variáveis aleatórias do bloco da Navegação.

Erro / Unidade	Tipo de Erro	Média da Distribuição	Desvio Padrão
Acelerômetro (erro_Acel) / aceleração m/s^2	<i>Run-to-Run</i>	0	0.1*g
Ruído no acelerômetro (ruído_Acel) / aceleração m/s^2	<i>In-Run</i>	0	0.02*g
Ganho do Giroscópio (erro_Kgiro) / porcentagem %	<i>Run-to-Run</i>	100	1
BIAS do Giroscópio (erro_BIASgiro) / graus °	<i>Run-to-Run</i>	0	0.01
Ruído do Giroscópio (ruído_giro) / graus °	<i>In-Run</i>	0	0.01
Valores iniciais dos ângulos de Euler (erro_euler_nav0) / graus °	<i>Run-to-Run</i>	0	0.5
Valores iniciais da velocidade do míssil (erro_vb_nav0) / velocidade m/s	<i>Run-to-Run</i>	0	3
Valores iniciais da posição do míssil (erro_Xe_nav0) / metro m	<i>Run-to-Run</i>	0	20

4.4 Variáveis aleatórias do Autodiretor

Abrange os erros referente à confiabilidade dos sensores utilizados no subsistema do Autodiretor tais como erros na medição da direção e velocidade angular do alvo e também leva em consideração o tempo que o Seeker pode levar para procurar e rastrear o alvo após ser acionado o laser. O Seeker é o subsistema responsável por receber as emissões IR provenientes do alvo, rastrear e acompanhar o alvo. Na tabela 4.4 observamos tais variáveis.

TABELA 4.4 – Valores das variáveis aleatórias do bloco do Autodiretor.

Erro	Tipo de Erro	Média da Distribuição	Desvio Padrão
Velocidade angular Y (erro_OmegaY) / graus/s	<i>Run-to-Run</i>	0	0.01
Ruido na velocidade angular Y (ruido_OmegaY) / graus/s	<i>In-Run</i>	0	0.005
Velocidade angular Z (erro_OmegaZ) / graus/s	<i>Run-to-Run</i>	0	0.01
Ruido na velocidade angular Z (ruido_OmegaZ) / graus/s	<i>In-Run</i>	0	0.005
Ângulo θ (erro_ThetaAD) / graus °	<i>Run-to-Run</i>	0	0.2
Ruido no Ângulo θ (ruido_ ThetaAD) / graus °	<i>In-Run</i>	0	0.01
Ângulo ψ (erro_PsiAD) / graus °	<i>Run-to-Run</i>	0	0.2
Ruido no ψ (ruido_ PsiAD) / graus °	<i>In-Run</i>	0	0.01

4.5 Variáveis aleatórias dos Atuadores

Abrange os erros referente à montagem e execução dos atuadores quando acionados. Na tabela 4.5 observamos tais variáveis.

TABELA 4.5 – Valores das variáveis aleatórias do bloco dos atuadores.

Erro / Unidade	Tipo de Erro	Média da Distribuição	Desvio Padrão
Delta do atuador direito (erro_Delta1) / graus °	<i>Run-to-Run</i>	0	0.5
Delta do atuador esquerdo (erro_Delta2) / graus °	<i>Run-to-Run</i>	0	0.5

4.6 Variáveis aleatórias na detecção da posição e velocidade iniciais do Alvo

Abrange os erros referente à inicialização das informações do alvo repassadas do lançador para o míssil. Inicialmente o subsistema de navegação considerará um alvo virtual com as características cinemáticas passadas para ele antes do lançamento. Na tabela 4.6 observamos tais variáveis.

TABELA 4.6 – Valores variáveis aleatórias do bloco de detecção da posição e velocidade iniciais do Alvo.

Erro / Unidade	Tipo de Erro	Média da Distribuição	Desvio Padrão
Valores iniciais da velocidade do alvo (erro_Va0) / velocidade m/s	<i>Run-to-Run</i>	0	3
Valores iniciais da posição do alvo (erro_a0) / metro m	<i>Run-to-Run</i>	0	20

5 Implementação das Variáveis Aleatórias na Simulação com 6 Graus de Liberdade

A simulação com 6 graus de liberdade utilizado nesse trabalho consiste de blocos integrados para ser possível realizar a simulação do voo do míssil porém diferentemente do modelo utilizado em [1] nesta simulação foram introduzidos os erros e ruídos (variáveis aleatórias) dos dois tipos citados anteriormente, *Run-to-Run* e *in-Run*.

Na Figura 5.1 temos uma visão geral do diagrama dos subsistemas do míssil que são responsáveis pela simulação. Os erros e incertezas foram introduzidos dentro de cada subsistema como será a apresentado a seguir.

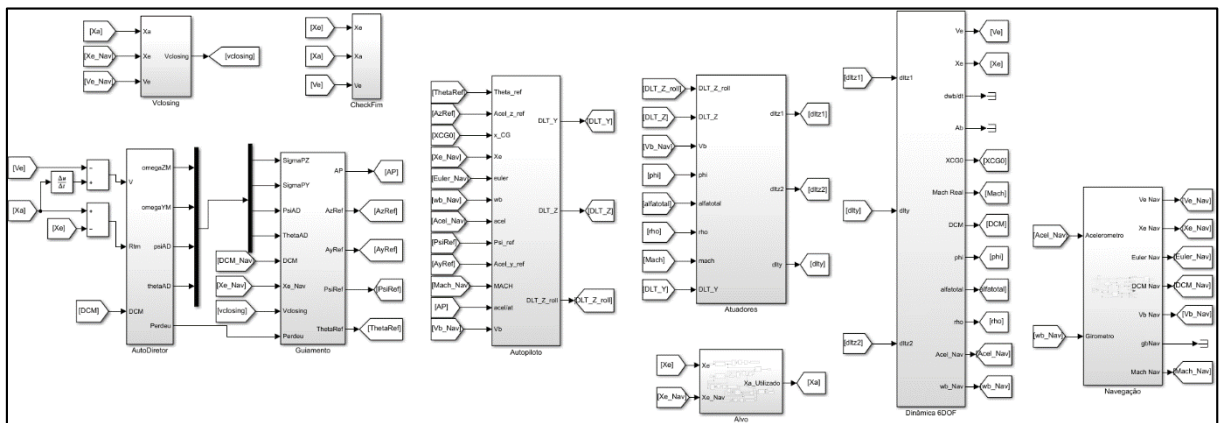


FIGURA 5.1 – Diagrama de bloco completo.

Nas Figuras 5.2 e 5.3 temos uma melhor visualização dos subsistemas do míssil.

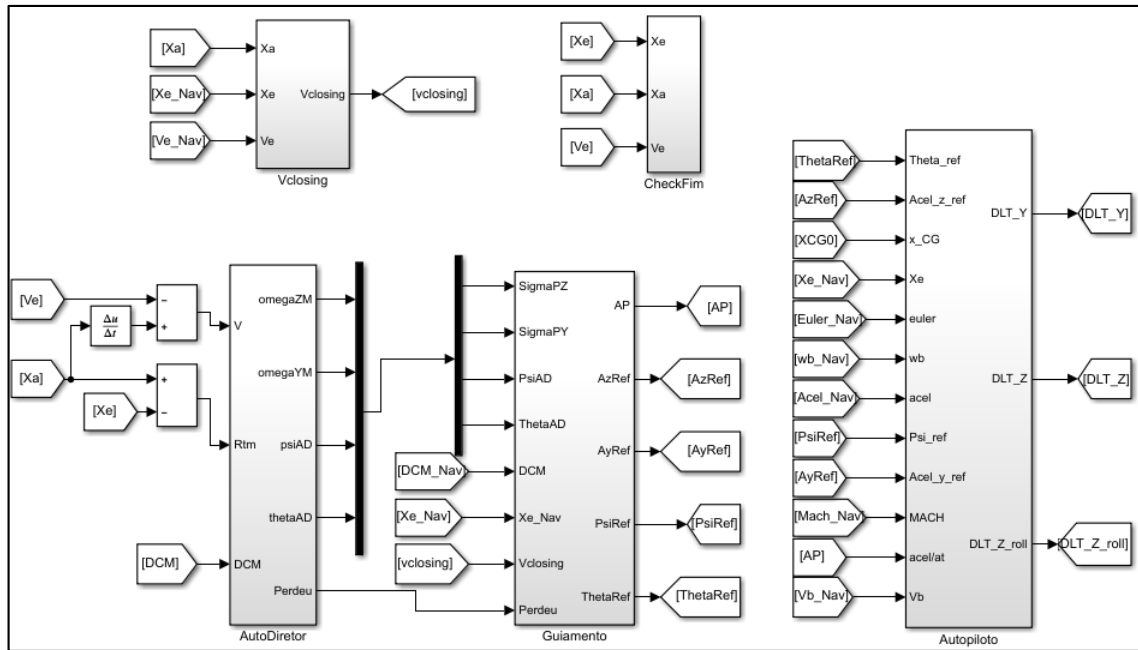


FIGURA 5.2 – Diagrama de bloco do Autodiretor, Guiamento, Autopiloto, Vclosing (bloco que calcula a velocidade do alvo) e CheckFim (bloco responsável por encerrar a simulação quando o míssil não acerta o alvo).

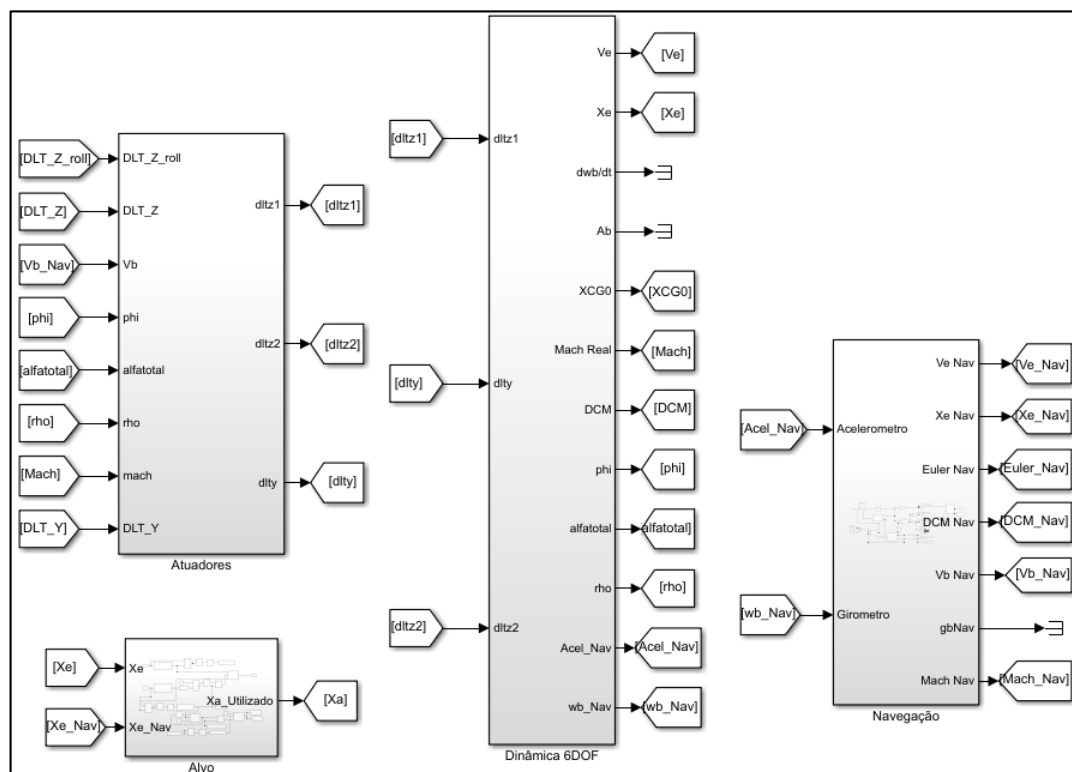


FIGURA 5.3 – Diagrama de bloco dos Atuadores, Dinâmica 6DOF, Navegação e Alvo (bloco que calcula a posição, velocidade e aceleração do alvo).

Já na Figura 5.4 observamos o interior do bloco de Dinâmica 6DOF que consiste dos blocos da Aerodinâmica, Propulsão, Acelerômetro (faz parte do conjunto de erros da navegação) e o bloco do 6DOF (efetua os cálculos dos valores dinâmicos reais do sistema, valores exatos do voo).

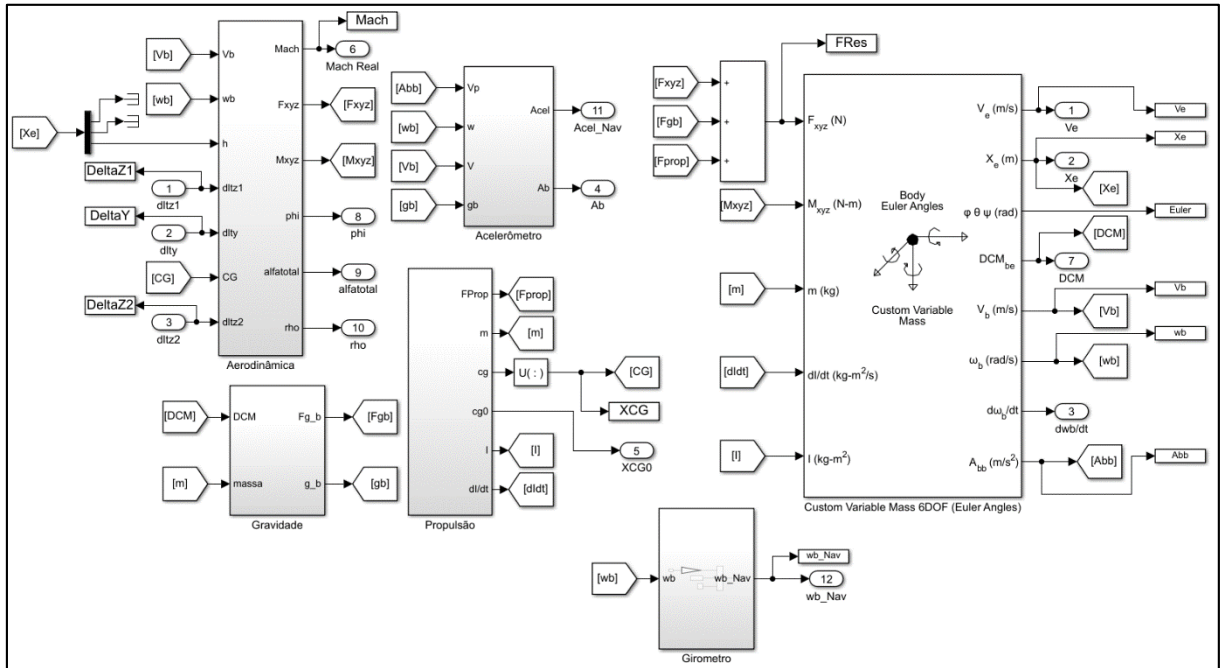


FIGURA 5.4 – Diagrama de bloco do interior do bloco de Dinâmica 6DOF.

A seguir será mostrado como foi realizado a implementação das variáveis aleatórias em cada um dos blocos que sofreram modificações. Serão mostrados apenas os blocos que sofreram modificações, os demais blocos permanecem iguais ao de [1].

5.1 Aerodinâmica

Na Figura 5.5 observamos o bloco responsável pelos cálculos da aerodinâmica do míssil já com as variáveis aleatórias implementadas de acordo com as equações 5.1 a 5.6.

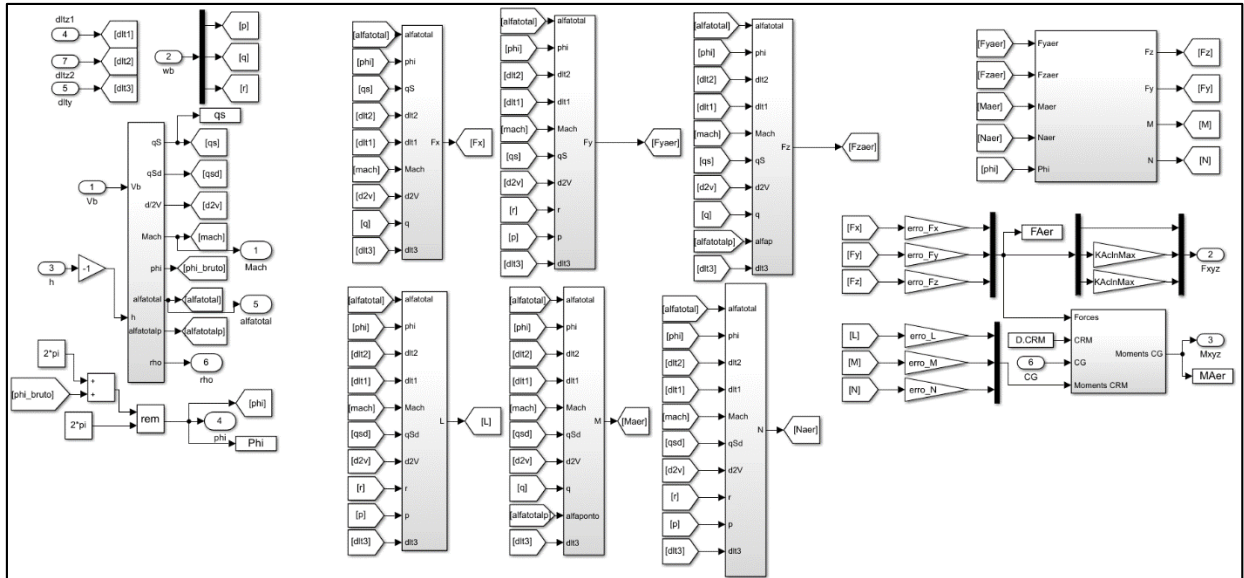


FIGURA 5.5 – Diagrama de bloco da aerodinâmica com variáveis aleatórias inseridas.

$$F_{x_real} = F_x * erro_Fx \quad (5.1)$$

$$F_{y_real} = F_y * erro_Fy \quad (5.2)$$

$$F_{z_real} = F_z * erro_Fz \quad (5.3)$$

$$L_real = L * erro_L \quad (5.4)$$

$$M_real = M * erro_M \quad (5.5)$$

$$N_real = M * erro_N \quad (5.6)$$

Na Figura 5.6 temos uma melhor visualização de como foi feita a implementação das seis variáveis aleatórias. Observamos também um ganho multiplicando as componentes em y e z da força aerodinâmica calculada, este será explicado posteriormente.

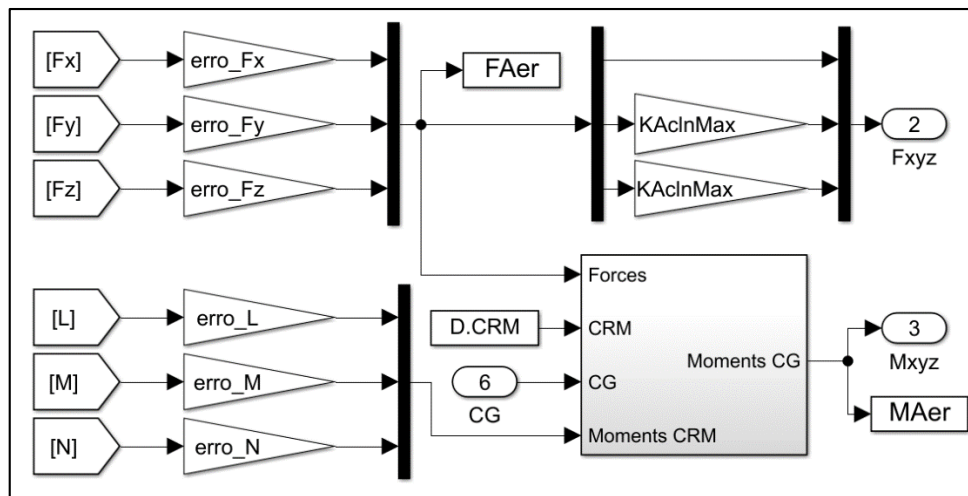


FIGURA 5.6 – Diagrama de bloco da aerodinâmica ampliado apenas onde foram inseridas variáveis aleatórias.

5.2 Propulsão

Na Figura 5.7 observamos o bloco da propulsão com as oito variáveis aleatórias implementadas conforme as Equações 5.7 a 5.14.

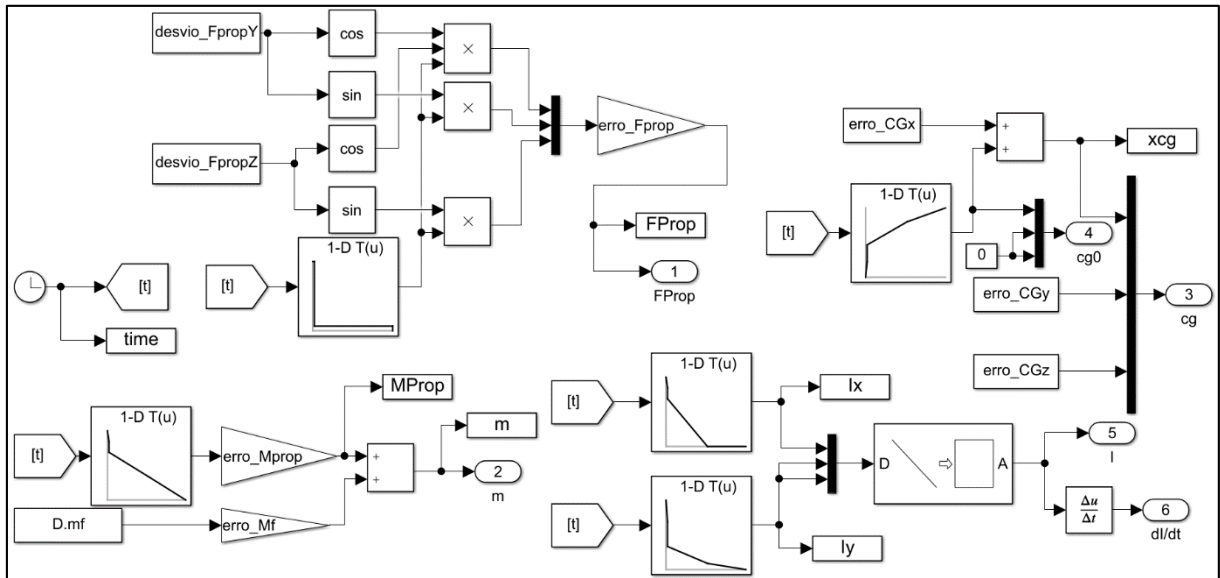


FIGURA 5.7 – Diagrama de bloco da propulsão com variáveis aleatórias inseridas.

$$Fpropx_real = Fprop * \cos(desvio_FpropY) * \cos(desvio_FpropY) * erro_Fprop \quad (5.7)$$

$$Fpropy_real = Fprop * \sin(desvio_FpropY) * erro_Fprop \quad (5.8)$$

$$Fpropz_real = Fprop * \sin(desvio_FpropZ) * erro_Fprop \quad (5.9)$$

$$Mf_real = Mf * erro_Mf \quad (5.10)$$

$$Mprop_real = Mprop * erro_Mprop \quad (5.11)$$

$$CGx_real = CGx + erro_CGx \quad (5.12)$$

$$CGy_real = erro_CGy \quad (5.13)$$

$$CGz_real = erro_CGz \quad (5.14)$$

5.3 Navegação

O subsistema de Navegação do míssil é composto pelos blocos de Navegação, Acelerômetro e giroscópio apresentados nas Figuras 5.8 a 5.10, respectivamente. As variáveis

aleatórias do bloco da navegação são variáveis aleatórias de inicialização da navegação inercial mostradas na Tabela 4.3.

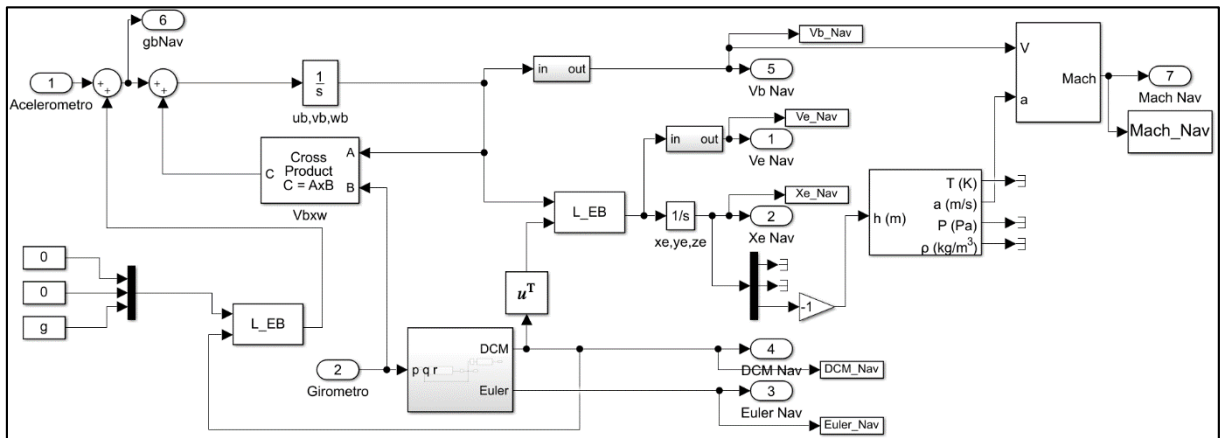


FIGURA 5.8 – Diagrama de bloco da navegação.

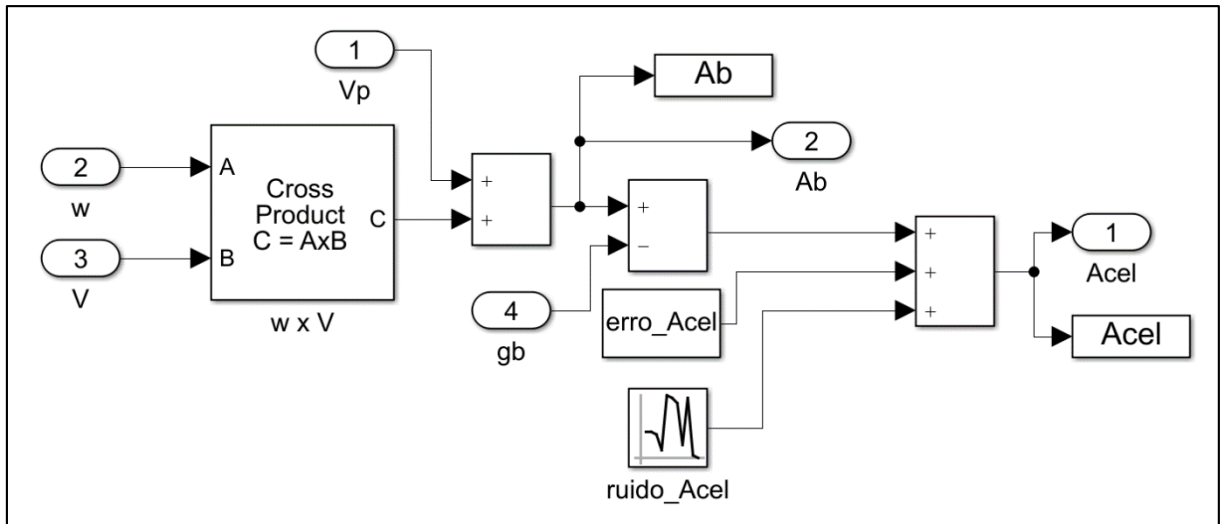


FIGURA 5.9 – Diagrama de bloco do Acelerômetro com variáveis aleatórias inseridas.

A Equação 5.15 representa como foram implementadas as variáveis aleatórias do bloco do acelerômetro.

$$\text{Acel_real} = \text{Acel} + \text{erro_Acel} + \text{ruído_Acel} \quad (5.15)$$

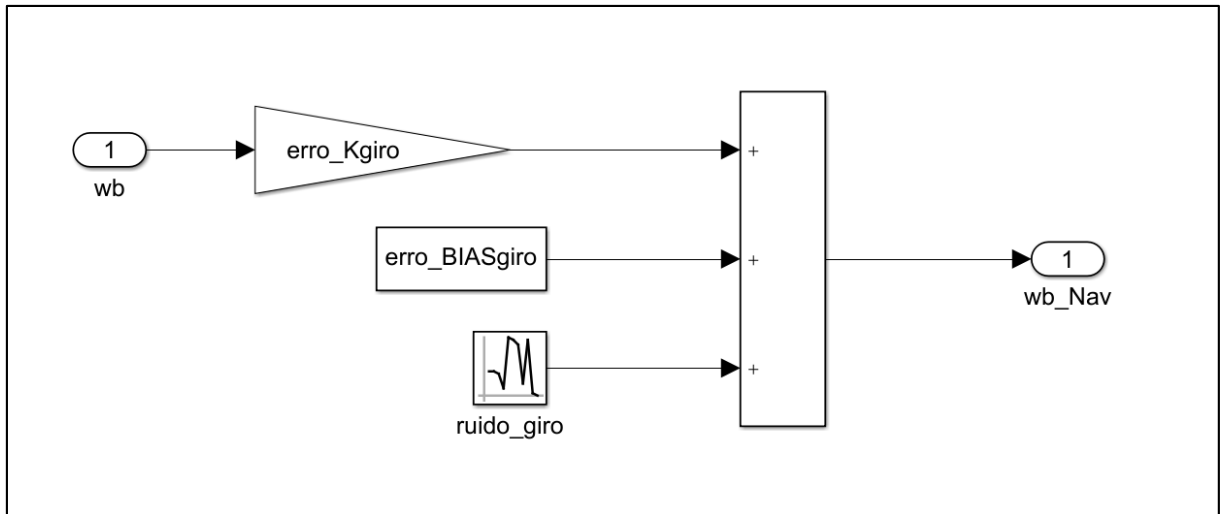


FIGURA 5.10 – Diagrama de bloco do Giroscópio com variáveis aleatórias inseridas.

A Equação 5.16 representa como foram implementadas as variáveis aleatórias do bloco do giroscópio.

$$\omega_{b_real} = \omega_b * \text{erro_Kgiro} + \text{erro_BIASgiro} + \text{ruído_giro} \quad (5.16)$$

5.4 Autodiretor

Na Figura 5.11 observamos o bloco do autodiretor subsistema responsável pela obtenção da posição e velocidade angular do alvo com a inserção das variáveis aleatórias conforme as Equações 5.17 a 5.18.

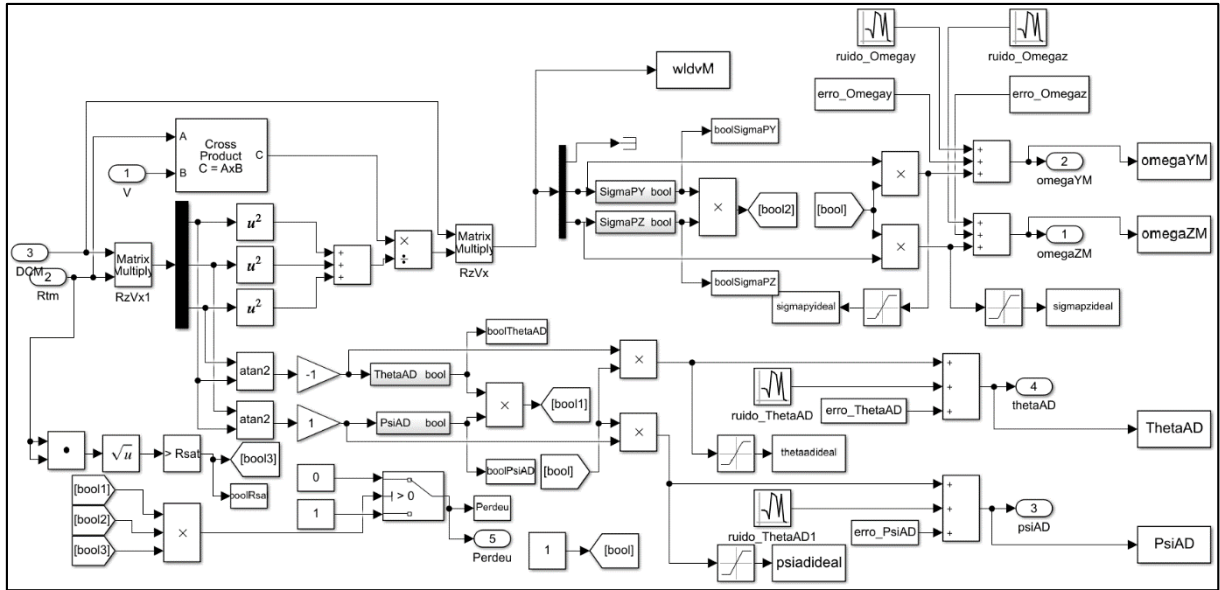


FIGURA 5.11 – Diagrama de bloco do Autodiretor com variáveis aleatórias inseridas.

$$\omega_{y_real} = \omega_y + \text{erro_}\omega_y + \text{ruído_}\omega_y \quad (5.17)$$

$$\omega_{z_real} = \omega_z + \text{erro_}\omega_z + \text{ruído_}\omega_z \quad (5.18)$$

$$\theta_{AD_real} = \theta_{AD} + \text{erro_}\theta_{AD} + \text{ruído_}\theta_{AD} \quad (5.17)$$

$$\psi_{AD_real} = \psi_{AD} + \text{erro_}\psi_{AD} + \text{ruído_}\psi_{AD} \quad (5.18)$$

5.5 Atuadores

Na Figura 5.12 observamos o bloco dos atuadores subsistema responsável por enviar o comando de deflexão aos atuadores. As variáveis aleatórias foram implementadas conforme as Equações 5.19 e 5.20.

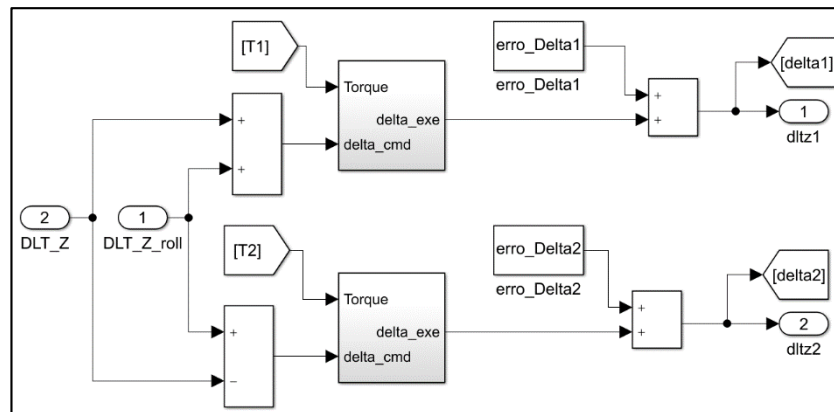


FIGURA 5.12 – Diagrama de bloco dos Atuadores com variáveis aleatórias inseridas.

$$\delta 1_{\text{real}} = \delta 1 + \text{erro_}\delta 1 \quad (5.19)$$

$$\delta 2_{\text{real}} = \delta 2 + \text{erro_}\delta 2 \quad (5.20)$$

5.6 Posição e Velocidade do Alvo

Na Figura 5.13 observamos o bloco responsável por calcular a posição e velocidade do alvo enquanto o míssil está na condição de navegação inercial, ou seja, o alvo está fora do alcance do laser do míssil. Para resolver tal situação foi criado um bloco para simular um alvo virtual ao qual o subsistema de guiamento vai perseguir até que a condição de alcance do laser será cumprida que é o alvo virtual estar a uma distância menor do que 6000 metros do míssil. Caso o míssil seja lançado a uma distância menor do que essa o seeker já estará ligado antes mesmo do lançamento. Isto significa que as variáveis aleatórias desse bloco só serão relevantes se o alvo estiver a uma distância maior do que o alcance laser.

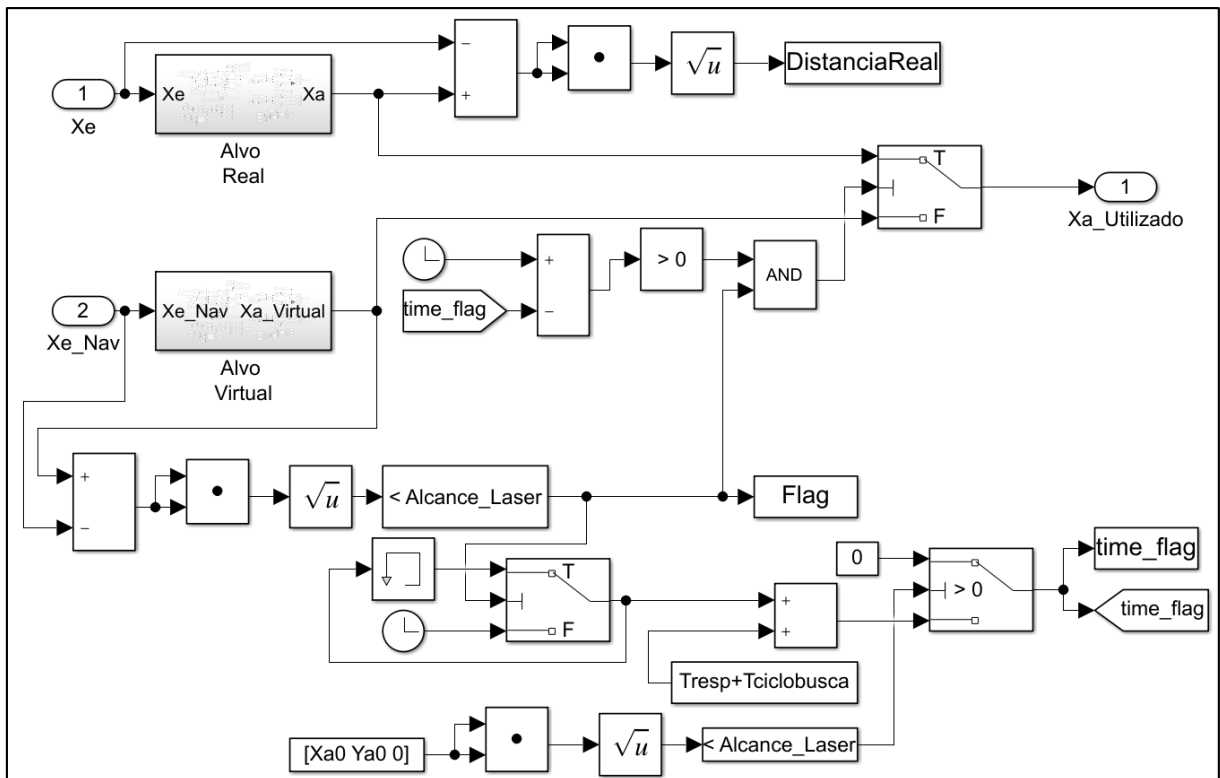


FIGURA 5.13 – Diagrama de bloco da cinemática do alvo virtual com variáveis aleatórias inseridas.

6 Definição dos Parâmetros Iniciais da Simulação

Para dar-se início as simulações do voo do míssil, primeiramente, foi necessário definir os principais parâmetros do autodiretor pois este influenciaria todos os demais além disso o impacto de um autodiretor com parâmetros muito robusto encareceria muito o custo total do míssil. Abaixo está descrito quais parâmetros foram escolhidos e na Tabela 6.1 está descrito os valores para cada parâmetro normalmente encontrado no mercado, do menos robusto ao mais. Quanto mais robusto o sensor maior é sua capacidade de detecção e maior é seu custo.

a) θ_{ADmax} (θ_{ADmax}): é o valor máximo de ângulo de visada no plano xy do corpo do míssil que o Seeker é capaz de captar as emissões IR.

b) ψ_{ADmax} (ψ_{ADmax}): é o valor máximo de ângulo de visada no plano xz do corpo do míssil que o Seeker é capaz de captar as emissões IR.

c) $\text{SigmapontoYmaxAD} = \omega_{yADmax}$ (ω_{yADmax}): é o valor máximo de velocidade angular no plano xy do corpo do míssil que o Seeker é capaz de rastrear.

d) $\text{SigmapontoZmaxAD} = \omega_{zADmax}$ (ω_{zADmax}): é o valor máximo de velocidade angular no plano xz do corpo do míssil que o Seeker é capaz de rastrear.

e) R_{sat} : é a distância de saturação de recepção das emissões IR.

TABELA 6.1 – Valores dos principais parâmetros do autodiretor e o respectivo nível de robustez.

Parâmetro / Unidade	Nível de Robustez (quanto maior, maior o custo)			
	Baixo	Médio	Alto	Muito Alto
θ_{ADmax} / graus	30	45	60	90
ψ_{ADmax} / graus	30	45	60	90
ω_{yADmax} / graus/s	17	40	40	-
ω_{zADmax} / graus/s	17	40	40	-
R_{sat} / metros m	200	150	100	< 50

Estes são os parâmetros básicos de um autodiretor e portanto essenciais que sejam definidos de acordo com as necessidades para atingir ou passar no máximo à 7 metros do alvo para que este possa ser atingindo pela detonação da cabeça de combate, definido como distância de passagem).

Ao simular um alvo cruzando a proa do lançador perpendicularmente e realizando manobra evasiva quando o míssil está a uma distância menor que 1000 metros do alvo, constatou-se que para as condições de voo e tipo de manobra adotada pelo alvo o míssil concebido em [1] não é capaz de cumprir o pré-requisito de distância de passagem do alvo por falta de capacidade de manobra lateral tanto no eixo y quando no eixo z. Devido tal acontecimento foi necessário a implementação de mais um parâmetro chamado de ganho de aceleração lateral (K_{Acln}) que representa uma correção na aerodinâmica do míssil (como por exemplo: empenas maiores para K_{Acln} maior que 1).

Portanto foram estudados os casos para os seguintes ganhos de aceleração lateral, 1, 1.1, 1.3, 1.5, 2.5 e 4, onde 1.1 representa uma pequena modificação na estrutura o míssil e 4 representa uma grande modificação.

Após a introdução do K_{Acln} , novas simulações foram realizadas utilizando o código MATLAB apresentado no Anexo deste trabalho. Observou-se que os parâmetros de limitação da leitura dos ângulos do alvo (θ_{ADmax} e ψ_{ADmax}) e as velocidades angulares de rastreo (ω_{yADmax} e ω_{zADmax}) não são fatores limitantes para o cumprimento do requisito de distância de passagem do alvo menor ou igual à 7 metros, pois o míssil é sempre lançado com o autodiretor alinhado com o alvo. Isto ocorre porque o míssil é para ser lançado de helicóptero e com helicóptero, para este tipo de alvo (aeronave voando a 120 m/s realizando manobra com fator de carga de 4g), é possível alinhar o míssil com a linha de visada antes do lançamento. Se o lançamento fosse por aeronave de asa fixa, que não consegue mudar a proa rapidamente, o ângulo máximo do autodiretor seria muito importante. O mesmo vale para velocidade angular da linha de visada (velocidade de rastreo).

Portanto para manter o menor custo possível foram escolhidos os parâmetros de menor nível para os ângulos máximos e velocidade de rastreo do autodiretor, restando apenas escolher os valores de distância de saturação do Seeker e o ganho de aceleração lateral.

Nas Figuras 6.1 e 6.2 observamos as distâncias mínimas e máximas respectivamente para cada uma das distâncias de saturação estudadas.

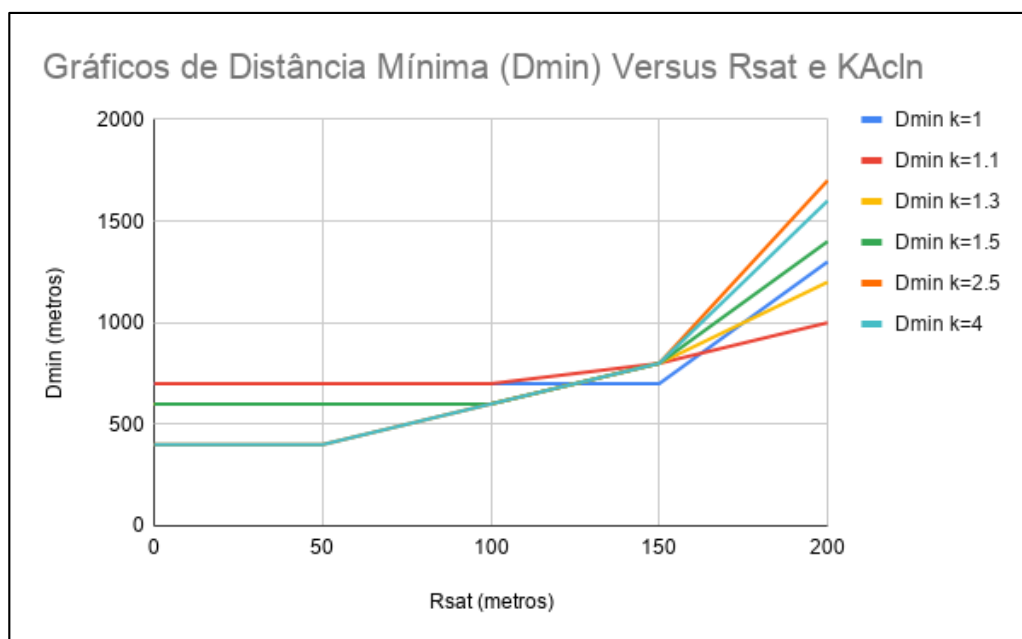


FIGURA 6.1 – Gráficos das distâncias mínimas versus distância de saturação e ganho aceleração lateral.

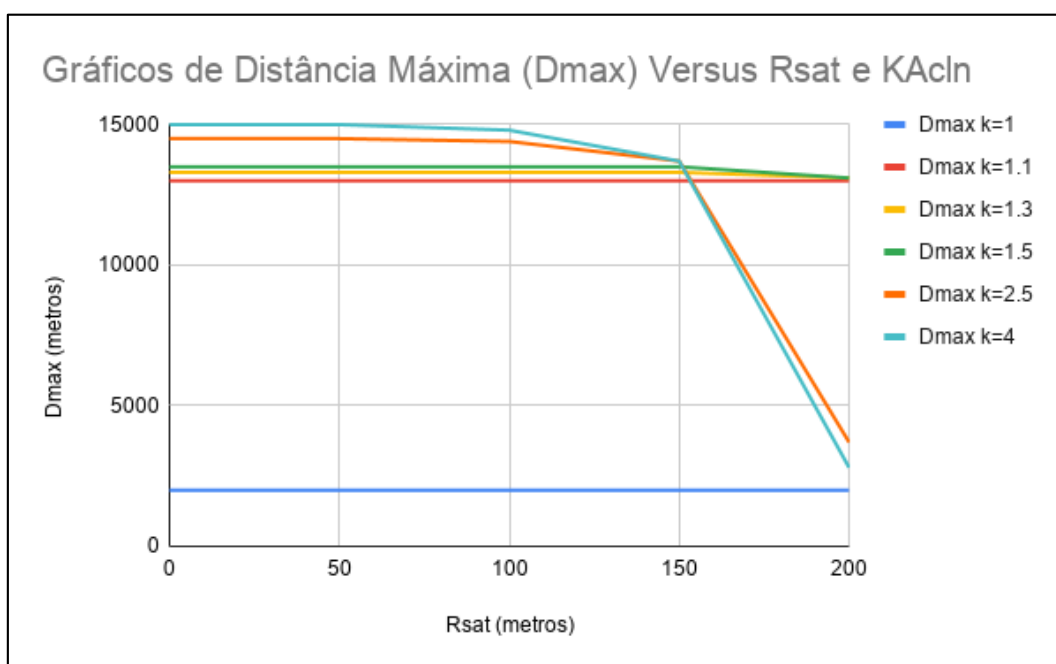


FIGURA 6.2 - Gráficos das distâncias máximas versus distância de saturação e ganho aceleração lateral.

A partir dos gráficos apresentados nas Figuras 6.1 e 6.2 concluímos que o melhor custo-benefício de distância de saturação é um $R_{sat} = 150$ metros, pois para $R_{sat} = 200$ metros as distâncias mínimas e máximas pioram muito e para valores menores de R_{sat} não há ganho para

a distância mínima e a melhora que ocorre para a distância máxima já encontra-se além do requisito de alcance de 10000 metros. Enquanto que para valores de K_{Acln} entre 1.1 a 1.5 satisfaz os requisitos. Portanto, inicialmente, é escolhido o menor ganho pois este representa uma menor alteração na estrutura do míssil.

6.1 Envelopes de Voo

O envelope de voo permite a visualização da região de kill probability do míssil em todas as direções que o alvo possa adotar, ou pelo menos todas as direções que foram simuladas. Com os envelopes de voo a tarefa de verificação dos parâmetros escolhidos do autodiretor foram assertivos ou não, conforme veremos a seguir.

Para a construção dos envelopes de voo estudados foi considerado que o alvo sempre está na mesma altitude do lançador e para os casos em que o alvo realiza manobra evasiva essa manobra é uma manobra de descida com fator de carga igual a 4g (4 vezes a gravidade) quando o míssil encontra-se à 1000 metros de distância do alvo. A direção do alvo inicial do alvo em todos os envelopes de voo é sempre a mesma, direção 180°.

Para a geração de dados e criação das imagens dos envelopes de voo foram utilizados os códigos de MATLAB presentes no Apêndice A.

6.1.1 Envelopes de voo considerando que o alvo não realiza manobra evasiva

Esse seria o cenário ideal pois o alvo não efetuando uma manobra evasiva não é exigido grandes acelerações laterais ao míssil. Conforme os requisitos estabelecidos pelo CONOPS em [1] a aeronave lançadora deve ser capaz de lançar o míssil entre as altitudes de 100 a 10700 metros foram realizados simulações para as alturas de 500, 1500, 3000, 6000 e 10000 metros.

Alvo encontra-se sempre no centro das figuras voando com velocidade de 120 m/s na direção 180°.

Nas Figuras 6.3 a 6.7 observamos os envelopes de voo quando o ganho para aceleração lateral é de apenas 1.1.

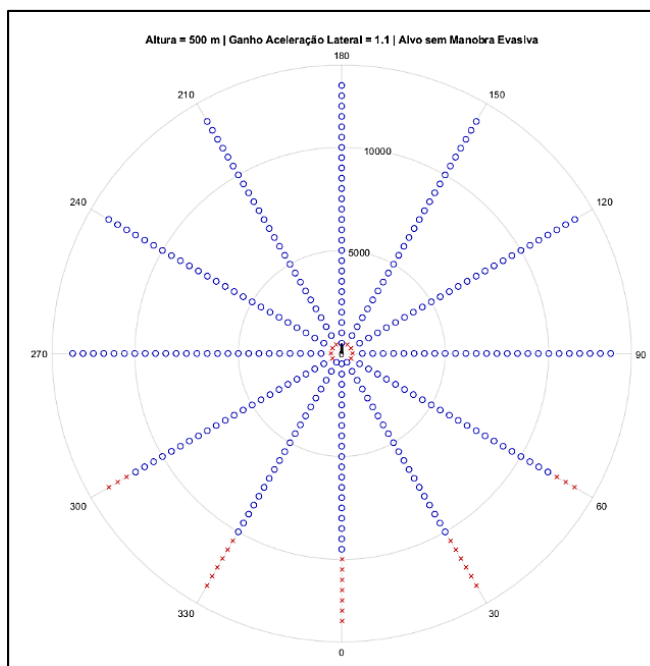


FIGURA 6.3 – Envelope de voo para uma altura de 500 metros e ganho de aceleração lateral de 1.1, alvo não realiza manobra evasiva. Alvo movendo-se com velocidade 120 m/s, na direção 180°.

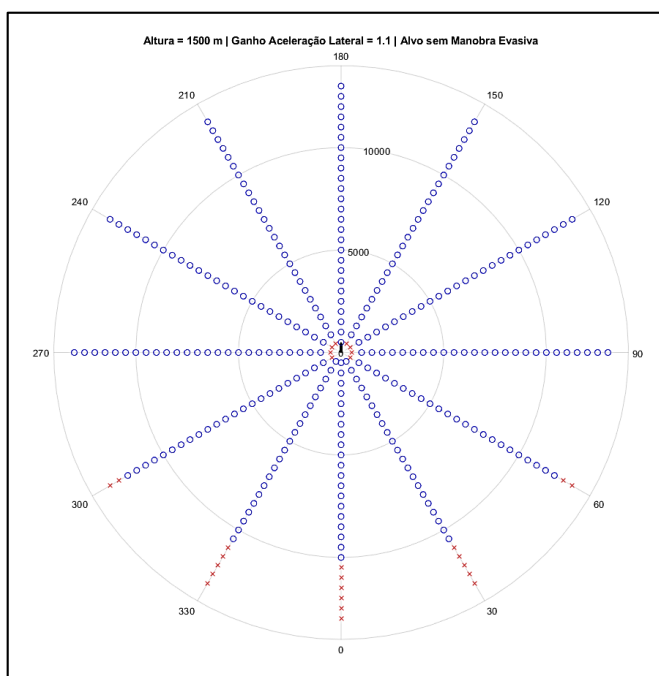


FIGURA 6.4 – Envelope de voo para uma altura de 1500 metros, ganho de aceleração lateral de 1.1, alvo não realiza manobra evasiva. Alvo movendo-se com velocidade 120 m/s, na direção 180°.

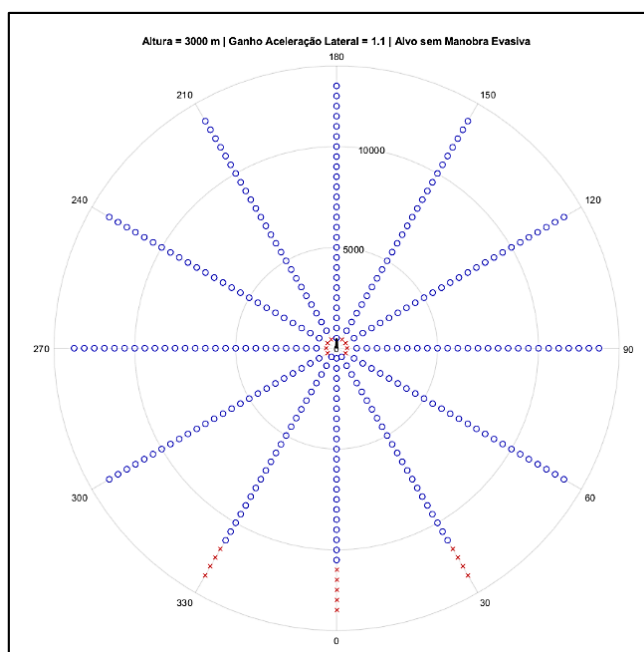


FIGURA 6.5 – Envelope de voo para uma altura de 3000 metros e ganho de aceleração lateral de 1.1, alvo não realiza manobra evasiva. Alvo movendo-se com velocidade 120 m/s, na direção 180°.

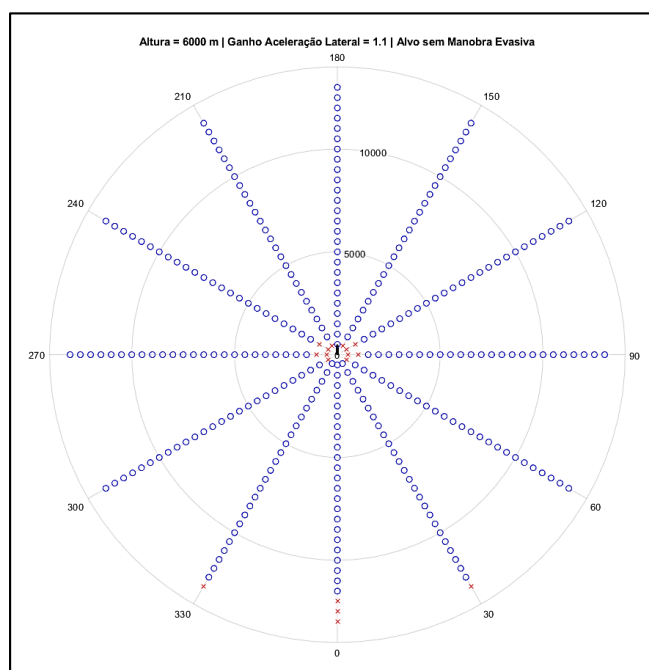


FIGURA 6.6 – Envelope de voo para uma altura de 6000 metros e ganho de aceleração lateral de 1.1, alvo não realiza manobra evasiva. Alvo movendo-se com velocidade 120 m/s, na direção 180°.

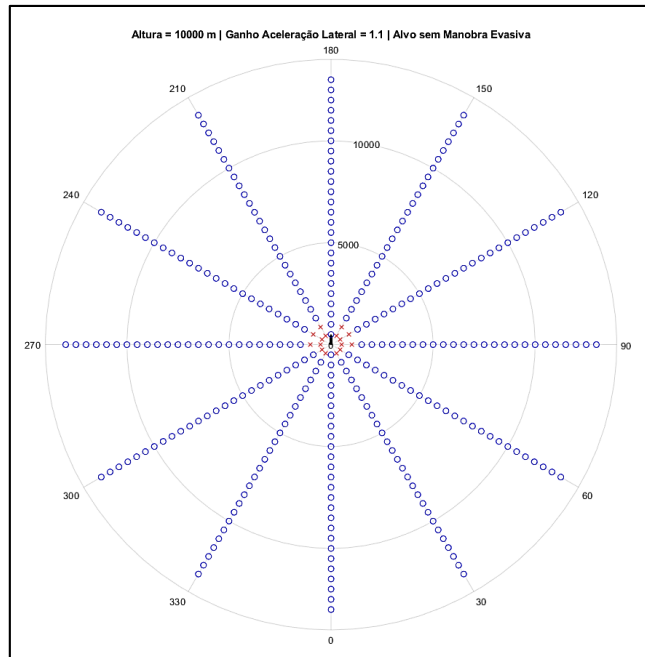


FIGURA 6.7 – Envelope de voo para uma altura de 10000 metros e ganho de aceleração lateral de 1.1, alvo não realiza manobra evasiva. Alvo movendo-se com velocidade 120 m/s, na direção 180°.

Concluimos que considerando a situação onde o alvo não tenta evadir-se do míssil e a não inserção das variáveis aleatórias os parâmetros iniciais escolhidos atendem aos pré-requisitos de altura e alcance.

Porém este cenário é muito otimista e não representa bem a realidade.

6.1.2 Envelopes de voo considerando que o alvo realiza manobra de evasão

Ao simular o envelope de voo quando o alvo realiza uma manobra evasiva descendente quando o míssil encontra-se a 1000 metros de distância do alvo os resultados foram bem diferentes conforme mostrado nas Figuras 6.8 a 6.12.

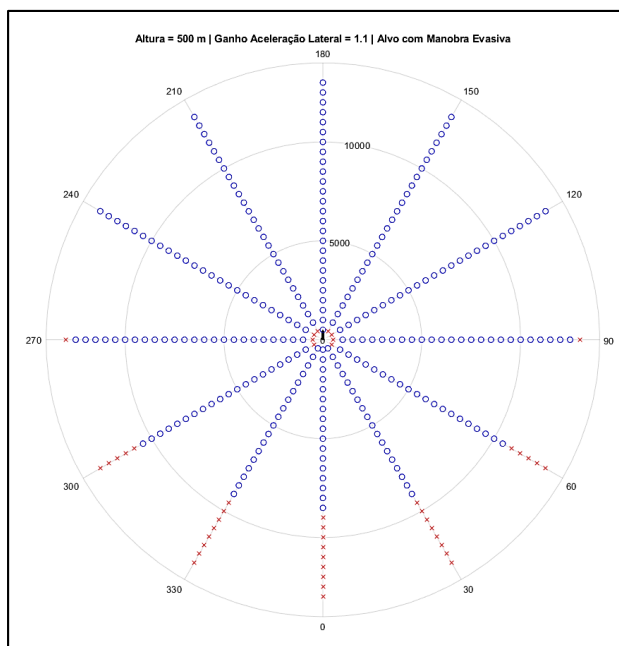


FIGURA 6.8 – Envelope de voo para uma altura de 500 metros, ganho de aceleração lateral de 1.1 e alvo manobrando evasivamente em direção ao solo. Alvo movendo-se com velocidade 120 m/s, na direção 180°.

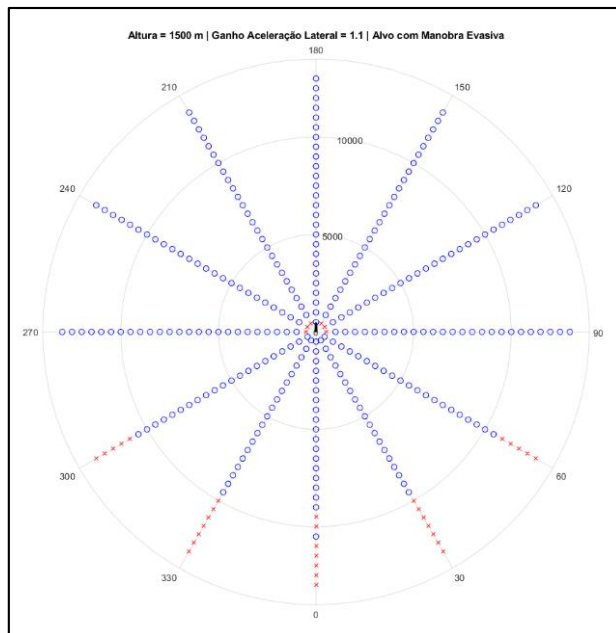


FIGURA 6.9 – Envelope de voo para uma altura de 1500 metros, ganho de aceleração lateral de 1.1 e alvo manobrando evasivamente em direção ao solo. Alvo movendo-se com velocidade 120 m/s, na direção 180°.

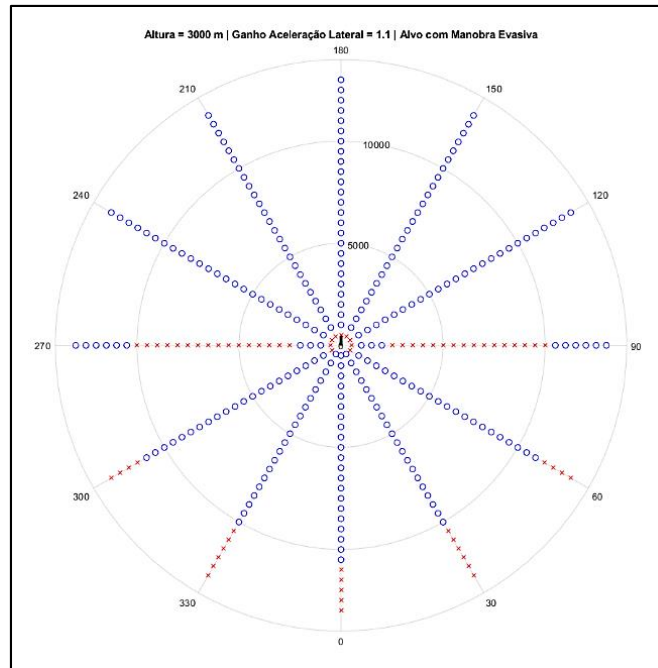


FIGURA 6.10 – Envelope de voo para uma altura de 3000 metros, ganho de aceleração lateral de 1.1 e alvo manobrando evasivamente em direção ao solo. Alvo movendo-se com velocidade 120 m/s, na direção 180°.

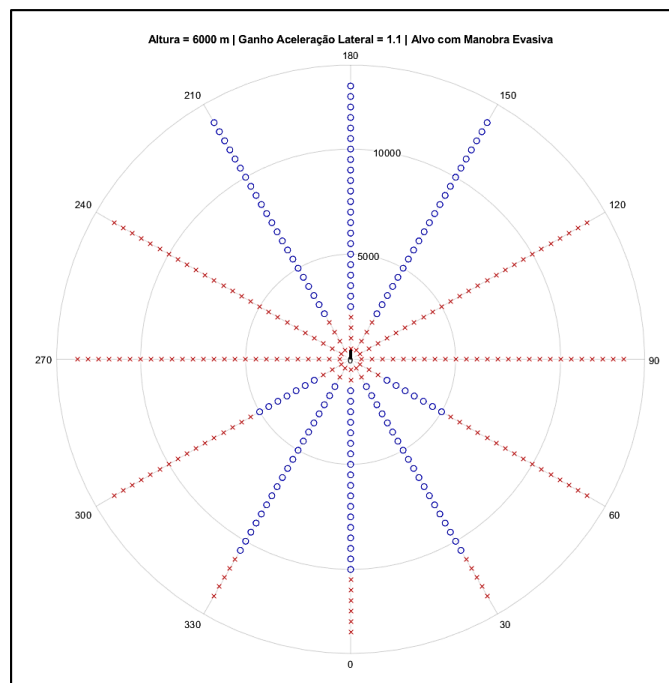


FIGURA 6.11 – Envelope de voo para uma altura de 6000 metros, ganho de aceleração lateral de 1.1 e alvo manobrando evasivamente em direção ao solo. Alvo movendo-se com velocidade 120 m/s, na direção 180°.

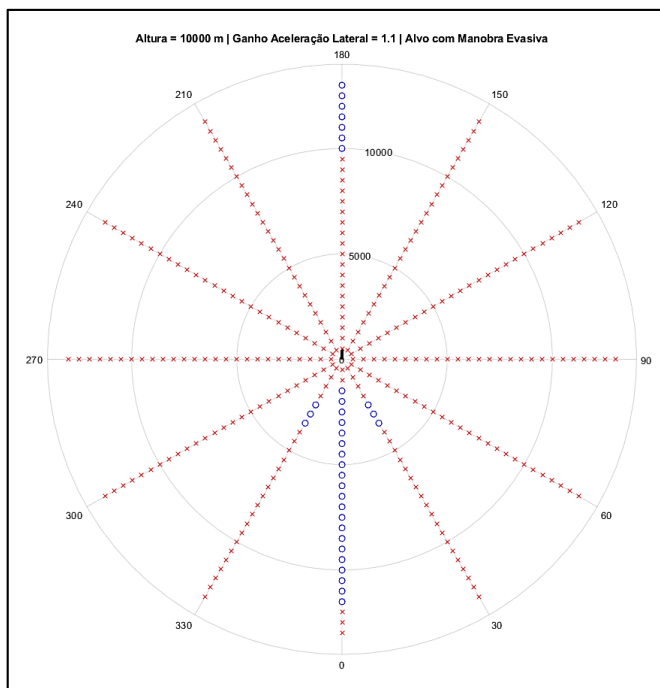


FIGURA 6.12 – Envelope de voo para uma altura de 10000 metros, ganho de aceleração lateral de 1.1 e alvo manobrando evasivamente em direção ao solo. Alvo movendo-se com velocidade 120 m/s, na direção 180°.

Portanto basta observamos as Figuras 6.10 a 6.12 para perceber que o ganho de aceleração lateral escolhido de 1.1 não é suficiente para gerar a aceleração lateral necessária para atingir o alvo. Ou seja, para alturas a partir de 3000 metros ou mais o ganho de aceleração lateral de 1.1 não é suficiente para o míssil não acerta o alvo quando este está passa perpendicularmente ao lançador.

Para solucionar esse problema foi necessário elevar o valor de K_{Acln} para 1.5 e assim gerar os envelopes de voo novamente, conforme Figuras 6.13 a 6.15. Foram gerados os envelopes de voo porém para nossa análise de envelope de voo serão apresentados apenas para as alturas de 3000, 6000 e 10000 metros. Pois para as alturas menores o ganho menor já satisfaz as condições, ganhos maiores tornariam o míssil mais manobrável.

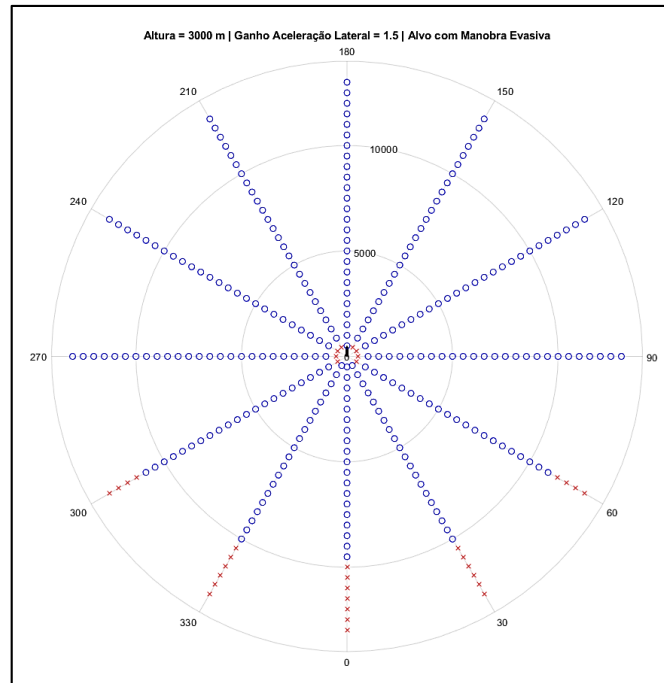


FIGURA 6.13 – Envelope de voo para uma altura de 3000 metros, ganho de aceleração lateral de 1.5 e alvo manobrando evasivamente em direção ao solo. Alvo movendo-se com velocidade 120 m/s, na direção 180°.

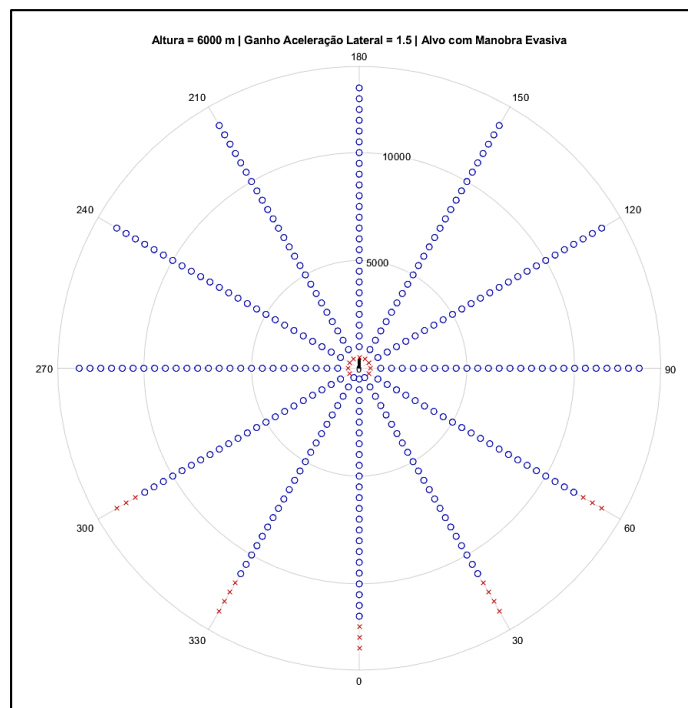


FIGURA 6.14 – Envelope de voo para uma altura de 6000 metros, ganho de aceleração lateral de 1.5 e alvo manobrando evasivamente em direção ao solo. Alvo movendo-se com velocidade 120 m/s, na direção 180°.

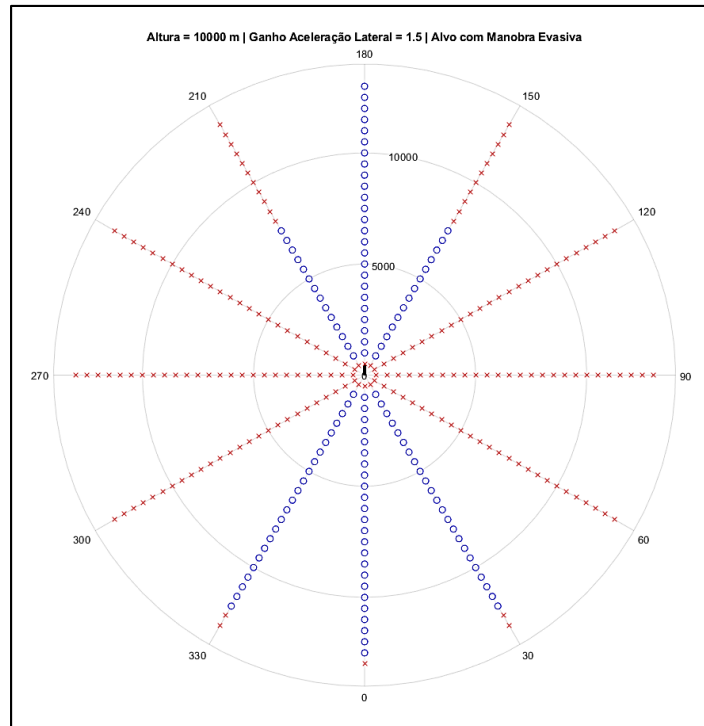


FIGURA 6.15 – Envelope de voo para uma altura de 10000 metros, ganho de aceleração lateral de 1.5 e alvo manobrando evasivamente em direção ao solo. Alvo movendo-se com velocidade 120 m/s, na direção 180°.

Na Figura 6.16 observamos o envelope de voo quando aplicando um ganho de aceleração lateral de 2.5.

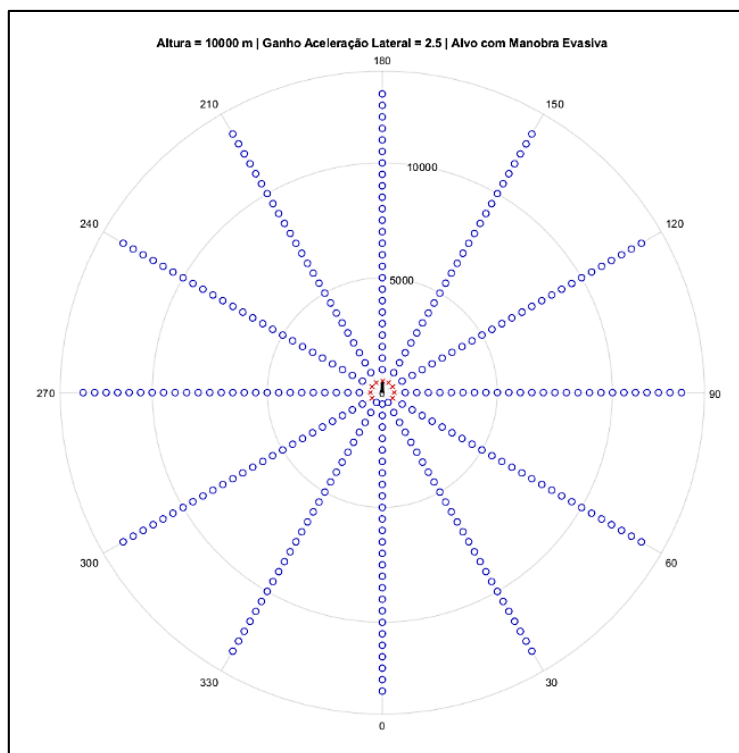


FIGURA 6.16 – Envelope de voo para uma altura de 10000 metros, ganho de aceleração lateral de 2.5 e alvo manobrando evasivamente em direção ao solo. Alvo movendo-se com velocidade 120 m/s, na direção 180°.

A modificação do KAcln de 1.1 para 1.5 resolveu a acurácia para as alturas de 3000 e 6000 metros mas não resolveu todos os problemas do envelope de voo para a altura de 10000 metros porém para solucionar todos os furos para essa altura seria necessário um KAcln de 2.5 porém essa mudança acarretaria uma modificação desnecessária na estrutura do míssil. De acordo com o especialista no assunto, Engº Carlos Alberto, modificar o KAcln para 2.5 não vale a pena pois uma altura de 10000 metros não é uma altura típica de lançamento e portanto é uma situação muito específica não compensando reprojetar todo o míssil para atender apenas essa faixa de altura. Assim tornando o custo projeto do míssil ainda mais elevado por demanda uma estrutura ainda mais robusta para suporta fatores de carga maiores.

7 Teste das Variáveis Aleatórias Implementadas

Finalmente vamos inserir as variáveis aleatórias que representam os erros e ruídos num voo real na simulação e utilizaremos o modelo de simulação de Monte Carlo para validar a escolha dos parâmetros.

7.1 Simulação Monte Carlo

Para cada simulação Monte Carlo os erros foram inseridos em 6 blocos, conforme mostrados anteriormente, e os erros foram sendo inseridos combinando cada um dos 6 blocos. Portanto, temos 63 combinações diferentes sem repetições de forma que seja possível identificar o conjunto de erros e ruídos mais sensíveis para a estabilidade do voo. Para cada combinação de blocos de erros a simulação Monte Carlo contou com 20 simulações do voo variando-se a semente (*seed*). Semente é a origem dos valores randomizados para as variáveis aleatórias dessas forma os resultados podem ser reproduzidos posteriormente. A semente utilizada segue a seguinte lógica, para a simulação do voo de número 1 a semente utilizada foi a 1, para a simulação do voo de número 2 a semente foi a 2, e assim por diante até o número estabelecido de simulações do voo para cada combinação de blocos de erro for atingido, ou seja, até a *seed* de número 20.

Para as simulações foi considerado como padrão a altura de 1500 metros, ou seja, é nessa altura que normalmente o lançamento desse time de armamento ocorre, para 3 distâncias de lançamento diferente, 1000, 5000 e 10000 metros, e para 3 posições do alvo diferente (afastando-se, aproximando-se e cruzando a proa do lançador perpendicularmente). Para fins acadêmicos poderíamos ter realizado a otimização dos requisitos dos subsistemas para os casos em que a distância de lançamento fosse igual a 1000 ou 5000 metros, porém é intuito perceber que para a distância de lançamento de 10000 metros os erros inseridos nos subsistemas representam maiores chances de diminuição da kill probability do míssil pois os erros e ruídos tem mais tempo para gerarem resultados cumulativos, principalmente na plataforma inercial que será responsável pela navegação do míssil por até 40% do tempo de voo, conforme veremos a seguir. As simulações Monte Carlo foram realizadas utilizando os códigos MATLAB do Apêndice B.

Vale relembrar que o valor desejado de kill probability é:

- 1) Maior ou igual a 80% para o chamado *objective* (objetivo);

- 2) Maior ou igual a 70% e menor que 80% para o chamado *threshold* (mínimo aceitável); e
- 3) Menor que 70% será considerado como *miss* (falha).

7.1.1 Simulação Monte Carlo para alvo aproximando-se do lançador

Na Tabelas 7.1 observamos os resultados da simulação Monte Carlo para o caso em que o alvo está a 1000, 5000 e 10000 metros de distância e está aproximando-se do lançador.

Nas colunas de blocos de erros (6 primeiras colunas) observamos quais blocos estão com erros inseridos e quais não estão e na coluna de kill probability observamos a chance de acerto para cada distância inicial estudada.

TABELA 7.1 – Simulação Monte Carlo para distância de lançamento de 1000, 5000 e 10000 metros, alvo aproximando-se do lançador e realizando manobra evasiva.

						(continua)		
Blocos de Erros						Kill Probability		
AERO	PROP	NAV	AD	AT	ALVO	1000 m	5000 m	10000 m
					X	100%	100%	100%
				X		100%	100%	100%
				X	X	100%	100%	100%
			X			100%	100%	100%
			X		X	100%	100%	100%
			X	X		100%	100%	100%
			X	X	X	100%	100%	100%
		X				100%	80%	100%
		X			X	100%	80%	100%
		X		X		100%	80%	100%
		X		X	X	100%	80%	100%
		X	X			100%	80%	100%
		X	X		X	100%	80%	100%
		X	X	X		100%	80%	100%
		X	X	X	X	100%	80%	100%
	X					100%	100%	100%

TABELA 7.1 – Simulação Monte Carlo para distância de lançamento de 1000, 5000 e 10000 metros, alvo aproximando-se do lançador e realizando manobra evasiva.

(continuação)								
Blocos de Erros						Kill Probability		
AERO	PROP	NAV	AD	AT	ALVO	1000 m	5000 m	10000 m
	X				X	100%	100%	100%
	X			X		100%	100%	100%
	X			X	X	100%	100%	100%
	X		X			100%	100%	100%
	X		X		X	100%	100%	100%
	X		X	X		100%	100%	100%
	X		X	X	X	100%	100%	100%
	X	X				100%	100%	100%
	X	X			X	100%	100%	100%
	X	X		X		100%	100%	100%
	X	X		X	X	100%	100%	100%
	X	X	X			100%	100%	100%
	X	X	X		X	100%	100%	100%
	X	X	X	X		100%	90%	100%
	X	X	X	X	X	100%	90%	100%
X						100%	100%	100%
X					X	100%	100%	100%
X				X		100%	100%	100%
X				X	X	100%	100%	100%
X			X			100%	100%	100%
X			X		X	100%	100%	100%
X			X	X		100%	100%	100%
X			X	X	X	100%	100%	100%
X		X				100%	100%	100%
X		X			X	100%	100%	100%
X		X		X		100%	100%	100%
X		X		X	X	100%	100%	100%
X		X	X			100%	100%	100%

TABELA 7.1 – Simulação Monte Carlo para distância de lançamento de 1000, 5000 e 10000 metros, alvo aproximando-se do lançador e realizando manobra evasiva.

						(conclusão)		
Blocos de Erros						Kill Probability		
AERO	PROP	NAV	AD	AT	ALVO	1000 m	5000 m	10000 m
X		X	X		X	100%	100%	100%
X		X	X	X		100%	100%	100%
X		X	X	X	X	100%	100%	100%
X	X					100%	90%	100%
X	X				X	100%	90%	90%
X	X			X		100%	100%	100%
X	X			X	X	100%	100%	100%
X	X		X			100%	100%	100%
X	X		X		X	100%	100%	100%
X	X		X	X		100%	100%	100%
X	X		X	X	X	100%	100%	100%
X	X	X				100%	100%	90%
X	X	X			X	100%	100%	90%
X	X	X		X		100%	100%	100%
X	X	X		X	X	100%	100%	100%
X	X	X	X			100%	100%	100%
X	X	X	X		X	100%	100%	100%
X	X	X	X	X		100%	100%	100%
X	X	X	X	X	X	100%	100%	100%

Observamos que para o caso em que o alvo aproxima-se do lançador a inserção das perturbações afeta muito pouco (menos de 10% para cada combinação) a kill probability.

7.1.2 Simulação Monte Carlo para alvo afastando-se do lançador

Na Tabela 7.2 temos os resultados da simulação Monte Carlo para o caso em que o alvo está a 1000, 5000 e 10000 metros de distância e está afastando-se do lançador.

TABELA 7.2 – Simulação Monte Carlo para distância de lançamento de 1000, 5000 e 10000 metros, alvo afastando-se do lançador e realizando manobra evasiva.

(continua)

Blocos de Erros						Kill Probability		
AERO	PROP	NAV	AD	AT	ALVO	1000 m	5000 m	10000 m
					X	100%	100%	0%
				X		100%	100%	40%
				X	X	100%	100%	40%
			X			100%	100%	0%
			X		X	100%	100%	0%
			X	X		100%	100%	20%
			X	X	X	100%	100%	20%
		X				100%	50%	0%
		X			X	100%	50%	10%
		X		X		100%	60%	0%
		X		X	X	100%	60%	0%
		X	X			100%	50%	10%
		X	X		X	100%	50%	10%
		X	X	X		100%	60%	20%
		X	X	X	X	100%	60%	20%
	X					100%	100%	30%
	X				X	100%	100%	30%
	X			X		100%	100%	40%
	X			X	X	100%	100%	40%
	X		X			100%	100%	0%
	X		X		X	100%	100%	10%
	X		X	X		100%	100%	50%
	X		X	X	X	100%	100%	50%
	X	X				100%	80%	10%
	X	X			X	100%	80%	10%
	X	X		X		100%	70%	20%
	X	X		X	X	100%	70%	20%
	X	X	X			100%	80%	20%

TABELA 7.2 – Simulação Monte Carlo para distância de lançamento de 1000, 5000 e 10000 metros, alvo afastando-se do lançador e realizando manobra evasiva.

						(continuação)		
Blocos de Erros						Kill Probability		
AERO	PROP	NAV	AD	AT	ALVO	1000 m	5000 m	10000 m
	X	X	X		X	100%	80%	20%
	X	X	X	X		100%	100%	10%
	X	X	X	X	X	100%	100%	10%
X						100%	100%	0%
X					X	100%	100%	0%
X				X		100%	100%	20%
X				X	X	100%	100%	20%
X			X			100%	100%	0%
X			X		X	100%	100%	0%
X			X	X		100%	100%	40%
X			X	X	X	100%	100%	40%
X		X				100%	40%	20%
X		X			X	100%	40%	20%
X		X		X		100%	70%	10%
X		X		X	X	100%	70%	20%
X		X	X			100%	40%	10%
X		X	X		X	100%	40%	10%
X		X	X	X		100%	60%	0%
X		X	X	X	X	100%	60%	0%
X	X					100%	100%	20%
X	X				X	100%	100%	20%
X	X			X		100%	100%	20%
X	X			X	X	100%	100%	20%
X	X		X			100%	100%	20%
X	X		X		X	100%	100%	10%
X	X		X	X		100%	100%	10%
X	X		X	X	X	100%	100%	20%
X	X	X				100%	40%	50%

TABELA 7.2 – Simulação Monte Carlo para distância de lançamento de 1000, 5000 e 10000 metros, alvo afastando-se do lançador e realizando manobra evasiva.

						(conclusão)		
Blocos de Erros						Kill Probability		
AERO	PROP	NAV	AD	AT	ALVO	1000 m	5000 m	10000 m
X	X	X			X	100%	40%	30%
X	X	X		X		100%	40%	30%
X	X	X		X	X	100%	40%	30%
X	X	X	X			100%	40%	10%
X	X	X	X		X	100%	40%	10%
X	X	X	X	X		100%	40%	10%
X	X	X	X	X	X	100%	40%	10%

Já na Tabela 7.2 observamos uma diferença significativa em comparação com os dados apresentados na Tabela 7.1. Quando o alvo está afastando-se do lançador para o caso do alvo a 1000 metros de distância não ocorre mudanças na kill probability e 100% dos casos o alvo será atingido. Enquanto que para a distância de 5000 metros observamos várias combinações de blocos de erros que reduzem a kill probability para um valor menor que 70%; e conforme esperado para a distância de 10000 metros o míssil não é mais capaz de atingir o alvo conforme mostrado no envelope de voo na Figura 6.13.

7.1.3 Simulação Monte Carlo para alvo cruzando a proa do lançador

Por fim, na Tabela 7.3 observamos resultados da simulação Monte Carlo para o caso em que o alvo está a 1000, 5000 e 10000 metros de distância e alvo cruzando a proa do lançador perpendicularmente e realizando manobra evasiva.

TABELA 7.3 – Simulação Monte Carlo para distância de lançamento de 1000, 5000 e 10000 metros, alvo cruzando a proa do lançador perpendicularmente e realizando manobra evasiva.

						(continua)		
Blocos de Erros						Kill Probability		
AERO	PROP	NAV	AD	AT	ALVO	1000 m	5000 m	10000 m
					X	100%	100%	100%
				X		100%	100%	100%
				X	X	100%	100%	100%
			X			100%	100%	100%
			X		X	100%	100%	100%
			X	X		100%	80%	100%
			X	X	X	100%	80%	100%
		X				100%	30%	30%
		X			X	100%	30%	30%
		X		X		100%	10%	0%
		X		X	X	100%	10%	0%
		X	X			100%	30%	20%
		X	X		X	100%	30%	20%
		X	X	X		100%	20%	20%
		X	X	X	X	100%	20%	20%
	X					100%	90%	70%
	X				X	100%	90%	70%
	X			X		100%	90%	90%
	X			X	X	100%	90%	90%
	X		X			100%	100%	70%
	X		X		X	100%	100%	70%
	X		X	X		100%	90%	100%
	X		X	X	X	100%	90%	100%
	X	X				100%	50%	10%
	X	X			X	100%	50%	10%
	X	X		X		100%	30%	0%
	X	X		X	X	100%	30%	0%
	X	X	X			100%	50%	10%

TABELA 7.3 – Simulação Monte Carlo para distância de lançamento de 1000, 5000 e 10000 metros, alvo cruzando a proa do lançador perpendicularmente e realizando manobra evasiva.

						(continuação)		
Blocos de Erros						Kill Probability		
AERO	PROP	NAV	AD	AT	ALVO	1000 m	5000 m	10000 m
	X	X	X		X	100%	50%	10%
	X	X	X	X		100%	40%	0%
	X	X	X	X	X	100%	40%	0%
X						100%	100%	100%
X					X	100%	100%	100%
X				X		100%	100%	100%
X				X	X	100%	100%	100%
X			X			100%	100%	100%
X			X		X	100%	100%	100%
X			X	X		100%	70%	80%
X			X	X	X	100%	70%	80%
X		X				100%	30%	10%
X		X			X	100%	30%	10%
X		X		X		100%	20%	30%
X		X		X	X	100%	20%	30%
X		X	X			100%	20%	10%
X		X	X		X	100%	20%	10%
X		X	X	X		100%	0%	10%
X		X	X	X	X	100%	0%	10%
X	X					100%	100%	70%
X	X				X	100%	100%	70%
X	X			X		100%	80%	100%
X	X			X	X	100%	80%	100%
X	X		X			100%	80%	70%
X	X		X		X	100%	80%	70%
X	X		X	X		100%	90%	80%
X	X		X	X	X	100%	90%	80%
X	X	X				100%	30%	0%

TABELA 7.3 – Simulação Monte Carlo para distância de lançamento de 1000, 5000 e 10000 metros, alvo cruzando a proa do lançador perpendicularmente e realizando manobra evasiva.

								(conclusão)
Blocos de Erros					Kill Probability			
AERO	PROP	NAV	AD	AT	ALVO	1000 m	5000 m	10000 m
X	X	X			X	100%	30%	0%
X	X	X		X		100%	40%	0%
X	X	X		X	X	100%	40%	0%
X	X	X	X			100%	20%	0%
X	X	X	X		X	100%	20%	0%
X	X	X	X	X		100%	20%	0%
X	X	X	X	X	X	100%	20%	0%

A partir das Tabelas 7.1, 7.2 e 7.3 é possível identificar quais os blocos de erros diminuem a chances de acerto para as condições de voo estabelecida. É fácil notar que a inserção de erros no subsistema de navegação já reduz a kill probability pela metade demonstrando ser o bloco mais sensível isoladamente e ao inserir outros erros que isoladamente não fariam a chance de acerto diminui para menos de 70% passa a reduzir ainda mais essa chance quando associado aos erros do bloco de navegação.

Ou seja, para a condição de voo mais crítica que é lançamento igual a 10000 metros, alvo cruzando a proa do míssil perpendicularmente e o alvo manobrando para baixo quando o míssil encontra-se à 1000 metros dele temos que o subsistema mais crítico é o de navegação apresentando uma kill probability menor que o threshold (mínimo esperado de 70%), apresentando uma chance de acerto de 50% quando o erro desse bloco está isolando e menor do 50% quando associado à erros de outros blocos.

7.2 Identificação das Variáveis Aleatórios Mais Sensíveis/Críticas

7.2.1 Subsistema mais crítico: Navegação

Para resolver tal situação foi necessário isolar os erros do bloco de navegação um a um e identificar qual deles eram os mais sensíveis, após várias rodadas de simulação Monte Carlo

foi possível observar que dos erros apresentados na Tabela 4.3 foi encontrado a seguinte ordem de criticidade:

- 1) Erro no Acelerômetro;
- 2) Erro nos valores iniciais de velocidade e ângulos de Euler do míssil; e
- 3) Ruído do Giroscópio.

Demais erros e ruídos do bloco de navegação não afetaram a chance de acerto e portanto foi decidido manter os valores originais pois correspondem à desvios padrão encontrados em componentes já utilizado no mercado. Na Tabela 7.4 observamos os valores de erros encontrados que isoladamente satisfazem os pré-requisitos de kill probability do míssil. Na coluna das incertezas estão em negrito os valores que foram alterados.

TABELA 7.4 – Valores otimizados das variáveis aleatórias do conjunto de erros do bloco da Navegação.

Erro / Unidade	Tipo de Erro	Média da Distribuição	Desvio Padrão
Acelerômetro / aceleração m/s^2	<i>Run-to-Run</i>	0	$0.005 * g$
Ruido no acelerômetro / aceleração m/s^2	<i>In-Run</i>	0	$0.02 * g$
Ganho do Giroscópio / porcentagem %	<i>Run-to-Run</i>	100	1
BIAS do Giroscópio / graus °	<i>Run-to-Run</i>	0	0.01
Ruido do Giroscópio / graus °	<i>In-Run</i>	0	$\frac{0.01}{3}$
Valores iniciais dos ângulos de Euler / graus °	<i>Run-to-Run</i>	0	$\frac{0.5}{2}$
Valores iniciais da velocidade do míssil / velocidade m/s	<i>Run-to-Run</i>	0	$\frac{3}{7}$
Valores iniciais da posição do míssil / metro m	<i>Run-to-Run</i>	0	20

7.2.2 Testes dos demais subsistemas

Porém encontrar o valores limítrofes para o bloco da navegação para se ter uma kill probability de pelo menos 70% não é o suficiente para garantir a chance de acerto para os demais casos de estudo. Portanto foi necessário 48 horas de simulações para encontrar todos parâmetros para que permitisse alcançar a kill probability desejada. Foi utilizado a seguinte metodologia para encontrar os valores das variáveis aleatórias que satisfizesse as condições de kill probability:

- 1) Para cada uma das 63 combinações de erros foi realizado a simulação Monte Carlo novamente e quando o resultado da kill probability é menor do que 70% de acerto o fator multiplicativo de cada erro que no início é igual a 1, ou seja 100%, é reduzido em 10% para cada um dos erros do bloco em estudo. Portanto o valor final do erro e da incerteza será 90% do valor inicial.
- 2) Após a redução do fator uma nova simulação Monte Carlo é realizada para a última combinação de erros anterior. Por exemplo na combinação de número 8 temos o caso quando o erro é inserido apenas no bloco da navegação, o fator referente ao bloco de navegação no início está igual à 100% após o passo 1 ele é reduzido para 90% e uma nova simulação Monte Carlo é executada. Caso o mesmo bloco de erro resulte numa kill probability menor que 70% uma nova diminuição de 10% será efetuada no fator.
- 3) Os passos 1 e 2 serão repetidos até todas as combinações tiverem sido abordadas e pôr fim a última combinação que contempla todos os erros inseridos na mesma simulação será testado e novamente os fatores podem ser diminuídos até que os requisitos de acerto sejam atingidos.

Portanto como resultado da execução do código apresentado no apêndice B obtivemos o seguinte resultado para a vetor de fatores multiplicativos dos blocos, conforme mostrado na Tabela 7.5

TABELA 7.5 – Valor final da vetor de fator multiplicativo das incertezas de cada bloco de erros dos subsistemas.

Bloco de Erro	Fator multiplicativo dos desvios padrões
Aerodinâmica	0.6 (60%)
Propulsão	1.0 (100%)
Navegação	0.3 (30%)
Autodiretor	0.9 (90%)
Atuador	0.8 (80%)
Informações do Alvo	1.0 (100%)

A Tabela 7.5 mostra que mesmo após a redução as incertezas aplicadas ao conjunto de erros do bloco de navegação ainda será necessário reduzir para 30% os valores dessas incertezas demonstrando mais uma vez que o subsistema de navegação é o mais crítico para esse míssil. Não é por coincidência que esse subsistema é um dos principais responsáveis pelo valor final do míssil.

Após encontrar o vetor de fatores apresentados na Tabela 7.5, foi realizado uma nova simulação Monte Carlo com todas as modificações e obtivemos os resultados para as condição mais crítica que é quando o alvo cruza a proa do míssil.

7.3 Teste Final das Variáveis Aleatórias

Na Tabela 7.6 observamos que os parâmetros escolhidos atendem o pré-requisito de kill probability para todas as combinações para uma altura de 1500 metros e distância inicial do alvo de 1000, 5000 e 10000 metros.

TABELA 7.6 – Simulação Monte Carlo para distância de lançamento de 1000, 5000 e 10000 metros, alvo cruzando a proa do lançador perpendicularmente e realizando manobra evasiva.

						(continua)		
Blocos de Erros						Kill Probability		
AERO	PROP	NAV	AD	AT	ALVO	1000 m	5000 m	10000 m
					X	100%	100%	100%
				X		100%	90%	100%

TABELA 7.6 – Simulação Monte Carlo para distância de lançamento de 1000, 5000 e 10000 metros, alvo cruzando a proa do lançador perpendicularmente e realizando manobra evasiva.

						(continuação)		
Blocos de Erros						Kill Probability		
AERO	PROP	NAV	AD	AT	ALVO	1000 m	5000 m	10000 m
				X	X	100%	90%	100%
			X			100%	100%	100%
			X		X	100%	100%	100%
			X	X		100%	80%	100%
			X	X	X	100%	80%	100%
		X				100%	100%	70%
		X			X	100%	100%	70%
		X		X		100%	80%	70%
		X		X	X	100%	80%	70%
		X	X			100%	100%	90%
		X	X		X	100%	100%	90%
		X	X	X		100%	100%	90%
		X	X	X	X	100%	100%	70%
	X					100%	90%	70%
	X				X	100%	90%	70%
	X			X		100%	90%	90%
	X			X	X	100%	90%	90%
	X		X			100%	100%	70%
	X		X		X	100%	100%	70%
	X		X	X		100%	90%	80%
	X		X	X	X	100%	90%	80%
	X	X				100%	90%	70%
	X	X			X	100%	90%	70%
	X	X		X		100%	100%	100%
	X	X		X	X	100%	100%	90%
	X	X	X			100%	100%	70%
	X	X	X		X	100%	100%	70%
	X	X	X	X		100%	80%	70%

TABELA 7.6 – Simulação Monte Carlo para distância de lançamento de 1000, 5000 e 10000 metros, alvo cruzando a proa do lançador perpendicularmente e realizando manobra evasiva.

						(continuação)		
Blocos de Erros						Kill Probability		
AERO	PROP	NAV	AD	AT	ALVO	1000 m	5000 m	10000 m
	X	X	X	X	X	100%	80%	70%
X						100%	100%	100%
X					X	100%	100%	100%
X				X		100%	80%	100%
X				X	X	100%	80%	100%
X			X			100%	100%	100%
X			X		X	100%	100%	100%
X			X	X		100%	70%	70%
X			X	X	X	100%	70%	70%
X		X				100%	100%	90%
X		X			X	100%	100%	90%
X		X		X		100%	70%	80%
X		X		X	X	100%	70%	80%
X		X	X			100%	100%	80%
X		X	X		X	100%	100%	80%
X		X	X	X		100%	90%	70%
X		X	X	X	X	100%	90%	70%
X	X					100%	100%	80%
X	X				X	100%	100%	80%
X	X			X		100%	70%	90%
X	X			X	X	100%	70%	90%
X	X		X			100%	80%	70%
X	X		X		X	100%	80%	80%
X	X		X	X		100%	90%	80%
X	X		X	X	X	100%	90%	80%
X	X	X				100%	100%	70%
X	X	X			X	100%	100%	70%
X	X	X		X		100%	100%	70%

TABELA 7.6 – Simulação Monte Carlo para distância de lançamento de 1000, 5000 e 10000 metros, alvo cruzando a proa do lançador perpendicularmente e realizando manobra evasiva.

						(conclusão)		
Blocos de Erros						Kill Probability		
AERO	PROP	NAV	AD	AT	ALVO	1000 m	5000 m	10000 m
X	X	X		X	X	100%	100%	70%
X	X	X	X			100%	80%	70%
X	X	X	X		X	100%	80%	70%
X	X	X	X	X		100%	90%	70%
X	X	X	X	X	X	100%	90%	70%

8 Definição dos Requisitos dos Principais Subsistemas

Após verificar que os pré-requisitos foram atendidos e para que não haja a necessidade de se carregar o vetor de fatores adiante optou-se por apresentar os valores dos requisitos dos principais subsistemas já com seus valores de média e incertezas otimizados, através da multiplicação das incertezas com os valores da Tabela 7.5.

No entanto a partir deste momento tais valores não são mais variáveis aleatórias ou perturbações ou erros de uma simulação. Os valores apresentados a seguir são as desvios padrão (requisitos) de cada subsistema.

8.1 Parâmetros do Subsistema da Aerodinâmica

Na Tabela 8.1 observamos os valores finais das desvios padrão para o subsistema da aerodinâmica do míssil. O fator multiplicativo apresentado na Tabela 7.5 foi de 60% para esse subsistema.

TABELA 8.1 – Valores dos parâmetros do subsistema da Aerodinâmica.

Parâmetro / Unidade	Desvio Padrão
Força em X (eixo x) / Porcentagem %	2
Força em Y (eixo y) / Porcentagem %	2
Força em Z (eixo z) / Porcentagem %	2
Momento L (ao longo do eixo x) / Porcentagem %	3
Momento M (ao longo do eixo y) / Porcentagem %	3
Momento N (ao longo do eixo z) / Porcentagem %	3

Além disso foi necessário incluir o ganho de aceleração lateral (K_{Acln}) igual a 1.5 conforme apresentado capítulo 7 deste trabalho.

8.2 Parâmetros do Subsistema de Propulsão

Na Tabela 8.2 observamos os valores finais das desvios padrão para o subsistema de propulsão do míssil. O fator multiplicativo apresentado na Tabela 7.5 foi de 100% para esse

subsistema, ou seja não houve necessidade de alterar as desvios padrão já previstas para o subsistema de propulsão.

TABELA 8.2 – Valores dos parâmetros do subsistema de Propulsão.

Parâmetro / Unidade	Desvio Padrão
Força Propulsiva / Porcentagem %	$\frac{10}{3}$
Desvio de Empuxo em Y / graus °	0.1
Desvio de Empuxo em Z / graus °	0.1
Massa Final do Míssil / Porcentagem %	1
Massa de Propelente / Porcentagem %	1
CG em X / metro m	0.0147
CG em Y / metro m	0.0147
CG em Z / metro m	0.0147

8.3 Parâmetros do Subsistema de Navegação

Na Tabela 8.3 observamos os valores finais das desvios padrão para o subsistema de Navegação do míssil. O fator multiplicativo apresentado na Tabela 7.5 foi de 30% para esse subsistema. Portanto mesmo com as mudanças realizadas durante os testes foi necessário reduzir ainda mais as desvios padrão do subsistema de Navegação. Isto demonstra qual dependente de um bom sistema de navegação é o míssil moderno.

TABELA 8.3 – Valores dos parâmetros do subsistema de Navegação.

Parâmetro / Unidade	Desvio Padrão
Acelerômetro / aceleração m/s²	0.0015*g
Ruido no acelerômetro / aceleração m/s²	0.006*g
Ganho do Giroscópio / porcentagem %	0.3
BIAS do Giroscópio / graus °	0.003
Ruido do Giroscópio / graus °	0.001
Valores iniciais dos ângulos de Euler / graus °	0.075
Valores iniciais da velocidade do míssil / velocidade m/s	$\frac{0.9}{7}$
Valores iniciais da posição do míssil / metro m	6

8.4 Parâmetros do Subsistema do Autodiretor

Na Tabela 8.4 observamos os parâmetros definidos para o subsistema do Autodiretor do míssil.

TABELA 8.4 – Valores finais dos parâmetros do subsistema do Autodiretor.

Parâmetro / Unidade	Valor do Parâmetro
θ_{ADmax} / graus	30
ψ_{ADmax} / graus	30
ω_{yADmax} / graus/s	17
ω_{zADmax} / graus/s	17
Rsat / metros m	150

Na Tabela 8.5 observamos os valores finais das desvios padrão para o subsistema do Autodiretor do míssil. O fator multiplicativo apresentado na Tabela 7.5 foi de 90% para esse subsistema.

TABELA 8.5 – Valores dos parâmetros do subsistema do Autodiretor.

Parâmetro / Unidade	Desvio Padrão
Velocidade angular Y / graus °/s	0.009
Ruido na velocidade angular Y / graus °/s	0.0045
Velocidade angular Z / graus °/s	0.009
Ruido na velocidade angular Z / graus °/s	0.0045
Ângulo θ (theta) / graus °	0.18
Ruido no Ângulo θ (theta) / graus °	0.009
Ângulo ψ (psi) / graus °	0.18
Ruido no ψ (psi) / graus °	0.009

8.5 Parâmetros do Subsistema dos Atuadores

Na Tabela 8.6 observamos os valores finais das desvios padrão para o subsistema dos Atuadores do míssil. O fator multiplicativo apresentado na Tabela 7.5 foi de 80% para esse subsistema.

TABELA 8.6 – Valores dos parâmetros do subsistema dos atuadores.

Parâmetro / Unidade	Desvio Padrão
Delta do atuador 1 (lado direito) / graus °	0.4
Delta do atuador 2 (lado esquerdo) / graus °	0.4

8.6 Parâmetros do Subsistema de detecção da posição e velocidade iniciais do Alvo

Na Tabela 8.7 observamos os valores finais das desvios padrão para o subsistema de detecção da posição e velocidade iniciais do alvo. O fator multiplicativo apresentado na Tabela 7.5 foi de 100% para esse subsistema, ou seja não houve necessidade de alterar as desvios padrão já previstas para este subsistema.

TABELA 8.7 – Valores dos parâmetros do subsistema de detecção da posição e velocidade iniciais do Alvo.

Parâmetro / Unidade	Desvio Padrão
Valores iniciais da velocidade do alvo / velocidade m/s	3
Valores iniciais da posição do alvo / metro m	20

Utilizando os valores de parâmetros e desvios padrões mostrados neste capítulo é possível atingir a uma Kill probability conforme Tabela 8.8.

TABELA 8.8 – Kill Probability (Chance de acerto) para um míssil ar-ar guiado a laser com os parâmetros e desvios padrões citados lançado de uma distância de 1000, 5000 e 10000 metros, alvo cruzando a proa do lançador perpendicularmente e realizando manobra evasiva.

Blocos de Erros						Kill Probability		
AERO	PROP	NAV	AD	AT	ALVO	1000 m	5000 m	10000 m
					X	100%	100%	100%
				X		100%	90%	100%
			X			100%	100%	100%
		X				100%	100%	70%
	X					100%	90%	70%
X						100%	100%	100%
X	X	X	X	X	X	100%	90%	70%

9 Conclusão

Portanto foi encontrado um conjunto de valores de variáveis aleatórias (parâmetros/desvios padrão) que otimizam ao máximo o voo e acerto do míssil (kill probability de pelo menos 70%) considerando as perturbações nas leituras de sensores embarcados, montagem e execução das superfícies de controle do míssil.

Assim temos os requisitos dos principais subsistemas estabelecidos de forma a garantir a kill probability para míssil ar-ar guiado a laser para alvos aproximando-se e afastando-se em linha reta do lançador e para uma condição mais severa de detecção, rastreamento e acerto que é quando o alvo cruza a proa do lançador.

As Recomendações para a continuação deste trabalho são:

- Realizar as simulações Monte Carlo para outras configurações de voo do alvo, por exemplo alvo aproximando ou afastando-se com um ângulo de 30°, 45°, 60° em relação a proa do lançador;
- Realizar as simulações Monte Carlo para alvo realizando outras manobras evasivas e até mesmo considerar a distância na qual o alvo vai iniciar uma manobra evasiva como mais uma variável aleatória;
- Utilizar um computador com capacidade computacional elevada para rodar simulações para todas as distâncias de lançamento e direção alvo possíveis, assim seriam gerados envelopes de voo com zonas de kill probability bem definidas;
- Inserir modelo não-linear do atuador, como descrito em [4]; e
- Estudar o processo de localização do alvo quando o laser é ligado, pois isto pode simplificar os requisitos da plataforma inercial/navegação. Na localização existe um trade-off entre o zoom que o operador do laser está usando e a sua capacidade de visualizar e achar o alvo.

Referências

- [1] PEIXOTO, Luiz; **Análise de desempenho operacional de mísseis ar-ar de curto alcance para aeronaves de asas rotativas**, 2018. 168 f. Trabalho de Graduação (Graduação em Engenharia Aeronáutica) – Instituto Tecnológico de Aeronáutica, São José dos Campos.
- [2] ALBERTO, CARLOS; **Programa Jovem Construtor: Processo de Engenharia de Sistemas**. São José dos Campos, SP: Odebrecht, 2014. 71 p. (PJC Turma 2).
- [3] ALBERTO, CARLOS; Notas de aula PRJ-75, versão 2.2, 2019.
- [4] BORGES, João; **Projeto de controlador para simulação HIL**, 2019. 141 f. Trabalho de Graduação (Graduação em Engenharia Eletrônica) – Instituto Tecnológico de Aeronáutica, São José dos Campos.

Apêndice A – Códigos MATLAB Utilizados para Gerar os Envelopes de Voo

A.1 Código MATLAB para gerar os dados do envelope de voo

```

clear all;
close all;

Altura0 = 1500;           % Altura de lançamento
passtheta = 30*D2R;      % Discretização em ângulo do envelope de voo
passX = 500;             % Discretização em distância de lançamento
dist_manobra = 1000;     % distância em que o alvo inicia a manobra evasiva
V0 = 120;                % (m/s) Modulo da Velocidade inicial do alvo

manobra = 5;             % Tipo de manobra evasiva: 4 -> Subir 5 -> Descer

ng = 4;                  % fator de carga

g = 9.80665;             % gravidade

D2R = pi/180;            % Constante graus -> rad
R2D = 1/D2R;

Tfim = 100;              % Tempo de simulação (s)

Dlt_max = 10*pi/180;     % Deflexão máxima do atuador
Dlt_max_roll = 2*pi/180;

D.Gama0 = 0*D2R;         % Angulo inicial
D.Alt0 = Altura0;        % (m) altitude inicial
D.Mach0 = 0.05;          % Velocidade de lançamento

p0 = 0*D2R;
q0 = 0*D2R;
r0 = 0*D2R;

phi0 = 0;
theta0 = 0;
psi0 = 0;
eul_0 = [phi0 theta0 psi0];
eul_statename = '';

% Alvo
Ya0 = 0;
Za0 = -Altura0;          % Za0 negativo acima do solo

[~,vsom,~,rho] = atmosisa(D.Alt0);

D.U = D.Mach0*vsom;      % Velocidade inicial
q = 1/2*rho*(D.U)^2;     % Pressão dinâmica

% Dados do míssil

```

```

D.dref = 0.147;           % Diâmetro (m)
D.c = D.dref;            % Comprimento de referência
D.S = pi*D.dref^2/4;     % Área de referência
D.R = D.dref/2;          % Raio (m)

% Características e Restrições (Parâmetros) do AutoDiretor
thetamaxAD = 30*D2R;
psimaxAD = 30*D2R;
sigmapYmaxAD = 17*D2R; % graus/s
sigmapZmaxAD = 17*D2R; % graus/s
Rsat = 150;              % Distancia em metros em que o radar fica cego
% Fator de correção na carga lateral, Fy e Fz
KAclnMax = 1.5;          % Ganho Aceleração lateral
Tciclobusca = 4;         % (s) tempo para o seeker encontrar o alvo
Alcance_Laser = 6000;    % Alcance Laser, antes dessa distância o míssil
persegue o alvo virtual
% Inserção de variáveis unitárias nos blocos (não alteram o valor de base)
% Variações aleatórias adicionadas no bloco de Aerodinâmica
erro_Fx = 1;
erro_Fy = 1;
erro_Fz = 1;
erro_L = 1;
erro_M = 1;
erro_N = 1;
% Variações aleatórias adicionadas no bloco de Propulsão
erro_Fprop = 1;
desvio_FpropY = 0;
desvio_FpropZ = 0;
erro_Mf = 1;
erro_Mprop = 1;
erro_CGx = 0;
erro_CGy = 0;
erro_CGz = 0;
% Variações adicionadas no bloco de Navegação
erro_Acel = 0;
ruído_Acel = [0 0];
erro_Kgiro = 1;
erro_BIASgiro = 0;
ruído_giro = [0 0];
erro_euler_nav0x = 0;
erro_euler_nav0y = 0;
erro_euler_nav0z = 0;
erro_vb_nav0x = 0;
erro_vb_nav0y = 0;
erro_vb_nav0z = 0;
erro_Xe_nav0x = 0;
erro_Xe_nav0y = 0;
erro_Xe_nav0z = 0;
% Variações aleatórias adicionadas no bloco do Autodiretor
erro_Omegay = 0;
ruído_Omegay = [0 0];
erro_Omegaz = 0;
ruído_Omegaz = [0 0];
erro_ThetaAD = 0;
ruído_ThetaAD = [0 0];
erro_PsiAD = 0;
ruído_PsiAD = [0 0];
% Variações aleatórias adicionadas no bloco do Atuador
erro_Delta1 = 0;
erro_Delta2 = 0;
% Variações aleatórias nos dados de posição e velocidade iniciais do alvo

```

```

erro_Xa0 = 0;
erro_Ya0 = 0;
erro_Za0 = 0;
erro_Vxa0 = 0;
erro_Vya0 = 0;
erro_Vza0 = 0;

% Cálculo do motor-foguete (queima radial)
% Booster
Phi_extB = D.dref - 0.005;           % Diâmetro externo do propelente (m)
Phi_intB = Phi_extB*0.715;           % Diâmetro interno do propelente (m)
VqueimaB = 0.0125;                   % Velocidade de queima do propelente (m/s)
Dens_propB = 1540;                   % Densidade do propelente (kg/m3)
IspbB = 230;                         % Impulso específico do propelente (s)
LpropB = 0.5;                        % Comprimento do propelente (m)
Vol_propB = pi/4*(Phi_extB^2 - Phi_intB^2)*LpropB; % Volume (m3)
m_prop0B = Vol_propB*Dens_propB;     % Massa (kg)
T_propB = (Phi_extB - Phi_intB)/(2*VqueimaB); % Tempo de queima (s)
F_propB = m_prop0B*g*IspbB/T_propB;  % Empuxo (N)
dmdtB = m_prop0B/T_propB;            % taxa de variação da massa (kg/s)

% Sustainer
Phi_extS = D.dref - 0.005;           % Diâmetro externo do propelente (m)
Phi_intS = 0;                        % Diâmetro interno do propelente (m)
VqueimaS = 0.010;                   % Velocidade de queima do propelente (m/s)
Dens_propS = 1540;                   % Densidade do propelente (kg/m3)
IspbS = 210;                         % Impulso específico do propelente (s)
LpropS = 0.507;                      % Comprimento do propelente (m)
Vol_propS = pi/4*(Phi_extS^2 - Phi_intS^2)*LpropS; % Volume (m3)
m_prop0S = Vol_propS*Dens_propS;     % Massa (kg)
T_propS = (LpropS)/(VqueimaS);       % Tempo de queima (s)
F_propS = m_prop0S*g*IspbS/T_propS;  % Empuxo (N)
dmdtS = m_prop0S/T_propS;            % taxa de variação da
massa (kg/s)

% Subsistemas
L_AD = 0.24;                         % (m) Comprimento do AD
m_AD = 5;                            % (kg) Massa do AD
L_CDG = 0.0954+0.1284;               % (m) Comprimento da cabeça de guerra + espoleta
m_CDG = 5+2;                         % (kg) Massa da cabeça de guerra + espoleta
L_ELE = 0.4209;                      % (m) Comprimento da Eletrônica + atuadores
m_ELE = 9;                           % (kg) Massa da Eletrônica + atuadores
L_MOT = LpropB + .1 + LpropS;         % (m) Comprimento do Propulsor
m_MOTB = m_prop0B/0.7;               % (kg) Massa do Propulsor + massa da
estrutura
m_MOTS = m_prop0S/0.8+5;

D.m = m_AD + m_CDG + m_ELE + m_MOTB + m_MOTS; % (kg) Massa do míssi
D.L = L_AD + L_CDG + L_ELE + L_MOT;      % (m) Comprimento do míssil

D.m_prop0 = m_prop0B+m_prop0S; % (kg) Massa de propelente

D.T_propB = T_propB;
D.T_propS = T_propS;

D.F_propB = F_propB;
D.F_propS = F_propS;
D.dmdtB = dmdtB;
D.dmdtS = dmdtS;
D.mf = D.m - D.m_prop0;               % (kg) Massa após a queima

```

```

% Cálculo do CG
xcg_AD = -(L_AD/2);
xcg_ELE = -(L_AD + L_ELE/2);
xcg_CDG = -(L_AD + L_CDG/2 + L_ELE);
xcg_MOTS = -(L_AD + L_CDG + L_ELE + LpropS/2);
xcg_MOTB = -(L_AD + L_CDG + L_ELE + LpropS + LpropB/2);

D.xcg0 = (xcg_AD*m_AD + xcg_CDG*m_CDG + xcg_ELE*m_ELE + xcg_MOTB*m_MOTB +
xcg_MOTS*m_MOTS)/D.m;
ycg0 = 0;
zcg0 = 0;

D.xcgBm = (xcg_AD*m_AD + xcg_CDG*m_CDG + xcg_ELE*m_ELE + ...
xcg_MOTB*(m_MOTB-m_prop0B/2) + xcg_MOTS*m_MOTS)/(D.m-m_prop0B/2);

D.xcgBf = (xcg_AD*m_AD + xcg_CDG*m_CDG + xcg_ELE*m_ELE + ...
xcg_MOTB*(m_MOTB-m_prop0B) + xcg_MOTS*m_MOTS)/(D.m-m_prop0B);

D.xcgSm = (xcg_AD*m_AD + xcg_CDG*m_CDG + xcg_ELE*m_ELE + ...
xcg_MOTB*(m_MOTB-m_prop0B) + xcg_MOTS*(m_MOTS-m_prop0S) + ...
(xcg_MOTS+LpropS/4)*m_prop0S/2)/(D.m-m_prop0B-m_prop0S/2);

D.xcgf = (xcg_AD*m_AD + xcg_CDG*m_CDG + xcg_ELE*m_ELE + ...
xcg_MOTB*(m_MOTB-m_prop0B)+xcg_MOTS*(m_MOTS-m_prop0S))/D.mf;

ycgf = 0;
zcgf = 0;

D.CRM = [D.xcgf 0 0]' - 0*[D.dref 0 0]';

% Cálculo das matrizes de inércia Ixx
D.Ix0 = D.m*D.R^2/2; % Inércia Ixx0 (kg.m2)
D.IxBm = (D.m-m_prop0B/2)*D.R^2/2;
D.IxBf = (D.m-m_prop0B)*D.R^2/2;
D.IxSm = (D.m-m_prop0B-m_prop0S)*D.R^2/2;
D.Ixf = D.mf*D.R^2/2; % Inércia Ixxf (kg.m2)

% Cálculo de Iyy de cada subsistema em relação ao CG inicial
Iy0_AD = (D.R^2/4 + L_AD^2/12 + (D.xcg0 - xcg_AD)^2)*m_AD;
Iy0_CDG = (D.R^2/4 + L_CDG^2/12 + (D.xcg0 - xcg_CDG)^2)*m_CDG;
Iy0_ELE = (D.R^2/4 + L_ELE^2/12 + (D.xcg0 - xcg_ELE)^2)*m_ELE;
Iy0_MOTB = (D.R^2/4 + LpropB^2/12 + (D.xcg0 - xcg_MOTB)^2)*m_MOTB;
Iy0_MOTS = (D.R^2/4 + LpropS^2/12 + (D.xcg0 - xcg_MOTS)^2)*m_MOTS;
D.Iy0 = Iy0_AD + Iy0_CDG + Iy0_ELE + Iy0_MOTB + Iy0_MOTS;

IyBm_AD = (D.R^2/4 + L_AD^2/12 + (D.xcgBm - xcg_AD)^2)*m_AD;
IyBm_CDG = (D.R^2/4 + L_CDG^2/12 + (D.xcgBm - xcg_CDG)^2)*m_CDG;
IyBm_ELE = (D.R^2/4 + L_ELE^2/12 + (D.xcgBm - xcg_ELE)^2)*m_ELE;
IyBm_MOTB = (D.R^2/4 + LpropB^2/12 + (D.xcgBm - xcg_MOTB)^2)*(m_MOTB-
m_prop0B/2);
IyBm_MOTS = (D.R^2/4 + LpropS^2/12 + (D.xcgBm - xcg_MOTS)^2)*m_MOTS;
D.IyBm = IyBm_AD + IyBm_CDG + IyBm_ELE + IyBm_MOTB + IyBm_MOTS;

IyBf_AD = (D.R^2/4 + L_AD^2/12 + (D.xcgBf - xcg_AD)^2)*m_AD;
IyBf_CDG = (D.R^2/4 + L_CDG^2/12 + (D.xcgBf - xcg_CDG)^2)*m_CDG;
IyBf_ELE = (D.R^2/4 + L_ELE^2/12 + (D.xcgBf - xcg_ELE)^2)*m_ELE;
IyBf_MOTB = (D.R^2/4 + LpropB^2/12 + (D.xcgBf - xcg_MOTB)^2)*(m_MOTB-
m_prop0B);

```

```

IyBf_MOTS = (D.R^2/4 + LpropS^2/12 + (D.xcgBf - xcg_MOTS)^2)*m_MOTS;
D.IyBf = IyBf_AD + IyBf_CDG + IyBf_ELE + IyBf_MOTB + IyBf_MOTS;

IySm_AD = (D.R^2/4 + L_AD^2/12 + (D.xcgSm - xcg_AD)^2)*m_AD;
IySm_CDG = (D.R^2/4 + L_CDG^2/12 + (D.xcgSm - xcg_CDG)^2)*m_CDG;
IySm_ELE = (D.R^2/4 + L_ELE^2/12 + (D.xcgSm - xcg_ELE)^2)*m_ELE;
IySm_MOTB = (D.R^2/4 + LpropB^2/12 + (D.xcgSm - xcg_MOTB)^2)*(m_MOTB -
m_prop0B);
IySm_MOTS = (D.R^2/4 + LpropS^2/12 + (D.xcgSm - xcg_MOTS)^2)*(m_MOTS -
m_prop0S) + (D.R^2/4 + (LpropS/2)^2/12 + (D.xcgSm -
(xcg_MOTS+LpropS/4))^2)*m_prop0S/2;
D.IySm = IySm_AD + IySm_CDG + IySm_ELE + IySm_MOTB + IySm_MOTS;

% Cálculo de Iyy de cada subsistema em relação ao CG final
Iyf_AD = (D.R^2/4 + L_AD^2/12 + (D.xcgf - xcg_AD)^2)*m_AD;
Iyf_CDG = (D.R^2/4 + L_CDG^2/12 + (D.xcgf - xcg_CDG)^2)*m_CDG;
Iyf_ELE = (D.R^2/4 + L_ELE^2/12 + (D.xcgf - xcg_ELE)^2)*m_ELE;
Iyf_MOTB = (D.R^2/4 + LpropB^2/12 + (D.xcgf - xcg_MOTB)^2)*(m_MOTB -
m_prop0B);
Iyf_MOTS = (D.R^2/4 + LpropS^2/12 + (D.xcgf - xcg_MOTS)^2)*(m_MOTS -
m_prop0S);
D.Iyf = Iyf_AD + Iyf_CDG + Iyf_ELE + Iyf_MOTB + Iyf_MOTS;

D.I0 = [D.Ix0  0      0; ...
        0      D.Iy0  0; ...
        0      0      D.Iy0];

D.IBm = [D.IxBm  0      0; ...
        0      D.IyBm  0; ...
        0      0      D.IyBm];

D.IBf = [D.IxBf  0      0; ...
        0      D.IyBf  0; ...
        0      0      D.IyBf];

D.ISm = [D.IxSm  0      0; ...
        0      D.IySm  0; ...
        0      0      D.IySm];

D.If = [D.Ixf  0      0; ...
        0      D.Iyf  0; ...
        0      0      D.Iyf];

% Autodiretor
wn_AD = 2*pi*30;
qsi_AD = sqrt(2)/2;
Tresp = .1;

% Guiamento
AltCruz = Za0; % (m) Altitude cruzeiro
RampaMax = 20*D2R; % (rad) Rampa máxima de mudança de altitude
GanhoAtit = .43*D2R; % (rad/m) Ganho de controle de atitude
AcInMax = 1000;

anp = 3.5; % Ganho de navegação proporcional

% Atuador
AT.Km = 0.09*1.37;
AT.Rm = 2.29;

```

```

AT.Kt = 0.134;
AT.J = 31.5 * 10^-6;
AT.Bm = 3.53 * 10^-3;
AT.Kp = 5.97*2.35;
AT.N1 = 3.28;
AT.N2 = 4;
AT.At = 1.5;
AT.Ap = 50;
wn = sqrt(AT.Ap*AT.Kp*AT.Km/AT.J/AT.N1/AT.N2/AT.Rm);
qsi =
(AT.N1*(AT.Bm*AT.Rm+AT.Km^2)+AT.Km*AT.Kt*AT.At)/AT.J/AT.N1/AT.Rm/2/wn;

CTI11 = cos(psi0)*cos(theta0);
CTI12 = sin(psi0)*cos(theta0);
CTI13 = -sin(theta0);
CTI21 = cos(psi0)*sin(theta0)*sin(phi0)-sin(psi0)*cos(phi0);
CTI22 = sin(psi0)*sin(theta0)*sin(phi0)+cos(psi0)*cos(phi0);
CTI23 = cos(theta0)*sin(phi0);
CTI31 = cos(psi0)*sin(theta0)*cos(phi0)+sin(psi0)*sin(phi0);
CTI32 = sin(psi0)*sin(theta0)*cos(phi0)-cos(psi0)*sin(phi0);
CTI33 = cos(theta0)*cos(phi0);
CTI = [CTI11 CTI12 CTI13; CTI21 CTI22 CTI23; CTI31 CTI32 CTI33];

Xe_0 = [0 0 -D.Alt0]';
Xa_0 = [Xa0 Ya0 Za0]';
delta0 = CTI*(Xa_0 - Xe_0);
thetaAD0 = -atan2(delta0(3),delta0(1));
psiAD0 = atan2(delta0(2),delta0(1));

% Dados aerodinâmicos
load('M_aed_3d.mat');

% Autopiloto
load('autopiloto_final.mat');

TTrav = 0.50;

i=1;
thetaApos=0;
for thetaA = 0:passtheta:(360-passtheta)*D2R % Direção do Alvo
    thetaApos=thetaApos+1;

    Vxa0 = V0*cos(thetaA);
    Vya0 = -V0*sin(thetaA);
    Vza0 = 0;

    Xa0pos=0;
    for Xa0 = 500:passX:1000 % distância inicial do alvo
        Xa0pos=Xa0pos+1;

        euler_nav0 = eul_0 + [erro_euler_nav0x erro_euler_nav0y
erro_euler_nav0z];
        vb_nav0 = [D.U 0 0] + [erro_vb_nav0x erro_vb_nav0y erro_vb_nav0z];
        Xe_nav0 = [0 0 -D.Alt0] + [erro_Xe_nav0x erro_Xe_nav0y
erro_Xe_nav0z];
        Xa0_Virtual = Xa0 + erro_Xa0;
        Ya0_Virtual = Ya0 + erro_Ya0;
        Za0_Virtual = Za0 + erro_Za0;
        Vxa0_Virtual = Vxa0 + erro_Vxa0;
        Vya0_Virtual = Vya0 + erro_Vya0;
    }
end

```

```

Vza0_Virtual = Vza0 + erro_Vza0;

sim('Sim_6DOF_completo.slx');

MissDist = sqrt((Xe.Data(end,1) - Xa.Data(end,1))^2 + ...
    (Xe.Data(end,2) - Xa.Data(end,2))^2 + ...
    (Xe.Data(end,3) - Xa.Data(end,3))^2);

disp(['Final Distance: ' num2str(MissDist) ' m']);

if MissDist > 7 %distancia de passagem maior que 7 metros
    envelope(thetaApos,Xa0pos)=0;
else
    envelope(thetaApos,Xa0pos)=1;
end

end
end
envelopevoo; % Chama o código que plota o envelope de voo

```

B.2 Código MATLAB para plotar o envelope de voo

```

close all

resultant_direction = 180*D2R;
resultant_length = 500;
%%%arrow head %%%
arrowhead_length = resultant_length/5; % arrow head length relative to
resultant_length
num_arrowlines = 100;
arrowhead_angle = deg2rad(30); % degrees
%%%arrow tip coordinates %%%
t1 = repmat(resultant_direction,1,num_arrowlines);
r1 = repmat(resultant_length,1,num_arrowlines);
%%%arrow base coordinates %%%
b = arrowhead_length.*tan(linspace(0,arrowhead_angle,num_arrowlines/2));
theta = atan(b./(resultant_length-arrowhead_length));
pre_t2 = [theta, -theta];
r2 = (resultant_length-arrowhead_length)./cos(pre_t2);
t2 = t1(1)+pre_t2;
%%%plot %%%
ax = polaraxes;
figure(1)
polarplot(ax,[t1(1) t1(1)],[0 r1(1)-
0.9*arrowhead_length],'k','linewidth',3)
if dist_manobra > 0
    title(['Altura = ',num2str(Altura0),' m | Ganho Aceleração Lateral =
',num2str(KAclnMax),' | Alvo com Manobra Evasiva'])
else
    title(['Altura = ',num2str(Altura0),' m | Ganho Aceleração Lateral =
',num2str(KAclnMax),' | Alvo sem Manobra Evasiva'])
end
hold on
polarplot(ax,[t1; t2],[r1; r2],'k')

[max1 max2]=size(envelope);
for thetaApos=1:max1%max(thetaApos)

```

```

for Xa0pos=1:max2
    if envelope(thetaApos,Xa0pos)==1
        polarplot(ax,(thetaApos-1)*passtheta,Xa0pos*passX,'b-o')
        %
        %hold on
    else
        polarplot(ax,(thetaApos-1)*passtheta,Xa0pos*passX,'r-x')
        %
        %hold on
    end
end
end
ax.ThetaZeroLocation = 'bottom';

```

Apêndice B – Códigos MATLAB utilizados na Simulação Monte Carlo

B.1 Código MATLAB para inicialização as condições de voo da simulação Monte Carlo

```
clear all;
close all;

Altura0 = 1500           % Altura de lançamento
num_var = 6;            %números de blocos de variáveis aleatórias criadas

% Ativar simulação monte carlo -> 1
monte_carlo = 1;
% aciona otimizador do fator multiplicativo dos erros
% que atenda aos requisitos de kil probability
otimiza = 0;
qtd_sim = 25;           % quantidade de simulações por rodada
fator = ones(1,num_var); % inicializa o vetor de fatores multiplicativos
% [Aero Prop Nav AD AT Alvo]
%fator = [0.6 1.0 0.3 0.9 0.8 1.0];

% Teste e Debugging
teste = 0;               % Teste = 1 -> modo debugging ligado
seed_teste = 1;
qtd_teste = 1;
comb_teste = [1 1 1 1 1 1]; % Escolhe quais blocos de erros serão
acionados
fator_teste = [0.6 1.0 0.3 0.9 0.8 1.0]; % Fator multiplicativo
utilizado no teste

manobra = 5;             % 4 -> Subir 5 -> Descer

ng = 4;                  % fator de carga

g = 9.80665;             % gravidade

D2R = pi/180;            % Constante graus -> rad
R2D = 1/D2R;

Tfim = 100;              % Tempo de simulação (s)

Dlt_max = 10*pi/180;     % Deflexão máxima do atuador
Dlt_max_roll = 2*pi/180;

D.Gama0 = 0*D2R;         % Angulo inicial
D.Alt0 = Altura0;        % (m) altitude inicial
D.Mach0 = 0.05;          % Velocidade de lançamento

p0 = 0*D2R;
```

```

q0 = 0*D2R;
r0 = 0*D2R;

phi0 = 0;
theta0 = 0;
psi0 = 0;
eul_0 = [phi0 theta0 psi0];
eul_statename = '';

% Alvo
Xa0 = 5000;           % (m) Posição inicial do alvo
Ya0 = 0;
Za0 = -Altura0;       % Za0 negativo acima do solo
dist_manobra = 1000;  % distância em que o alvo inicia a manobra evasiva
V0 = 120;             % (m/s) Modulo da Velocidade inicial do alvo

thetaA = 0*D2R;       % Direção do Alvo

Vxa0 = V0*cos(thetaA);
Vya0 = -V0*sin(thetaA);
Vza0 = 0;

[~,vsom,~,rho] = atmosisa(D.Alt0);

D.U = D.Mach0*vsom;    % Velocidade inicial
q = 1/2*rho*(D.U)^2;   % Pressão dinâmica

% Dados do míssil
D.dref = 0.147;        % Diâmetro (m)
D.c = D.dref;          % Comprimento de referência
D.S = pi*D.dref^2/4;   % Área de referência
D.R = D.dref/2;        % Raio (m)

% Características e Restrições (Parâmetros) do AutoDiretor
thetamaxAD = 30*D2R;
psimaxAD = 30*D2R;
sigmapYmaxAD = 17*D2R; % graus/s
sigmapZmaxAD = 17*D2R; % graus/s
Rsat = 150;            % Distância em metros em que o radar fica cego
% Fator de correção na carga lateral, Fy e Fz
KAclnMax = 1.5;        % Ganho Aceleração lateral máxima
Tciclobusca = 4;       % (s) tempo para o seeker encontrar o alvo
Alcance_Laser = 6000;  % Alcance Laser, antes dessa distância o míssil
persegue o alvo virtual

disp(['Distância Inicial do Alvo: ' num2str(Xa0) ' m']);
disp(['thetamaxAD: ' num2str(thetamaxAD/D2R) ...
', psimaxAD: ' num2str(psimaxAD/D2R) ', sigmapYmaxAD: '
num2str(sigmapYmaxAD/D2R) ...
', sigmapZmaxAD: ' num2str(sigmapZmaxAD/D2R) ', Rsat: '
num2str(Rsat) ...
', KAclnMax: ' num2str(KAclnMax)]);

montecarlo; % Executa o código da simulação Monte Carlo

killprobability; % Plota a kill probability para cada combinação de erros

```

B.2 Código MATLAB da simulação Monte Carlo

```

i = num_var;
tam = 0;
while i>0
    i = i-1;
    tam = tam + 2^i;
end

comb = zeros(tam,num_var);
for p = 1:tam
    aux=p+1;
    for j=5:-1:0
        if aux>2^j
            comb(p,num_var-j)=1;
            aux=aux-2^j;
        end
    end
end

% Inicializando as várias do looping
pos_flag=zeros(tam,qtd_sim);
p=1;
completo=0;
while p<=tam
    if monte_carlo == 1
        if teste
            numsim = qtd_teste;
            comb = comb_teste;
            fator = fator_teste;
        else
            numsim = qtd_sim;    % Número de simulações do monte carlo
        end
        % Printar os blocos de variaveis que estão sendo inseridas
        % 0 0 0 0 0 1 varia ALVO
        % 0 0 0 0 1 0 varia AT
        % 0 0 0 1 0 0 varia AD
        % 0 0 1 0 0 0 varia NAV
        % 0 1 0 0 0 0 varia PROP
        % 1 0 0 0 0 0 varia AERO
        % 1 0 0 0 0 1 varia AERO e ALVO
        % assim por diante...

        if comb(p,1)==1 disp('AERO'); end    % Aerodinamica
        if comb(p,2)==1 disp('PROP'); end    % Propulsão
        if comb(p,3)==1 disp('NAV'); end     % Navegação
        if comb(p,4)==1 disp('AD'); end      % AutoDiretor
        if comb(p,5)==1 disp('AT'); end      % Atuador
        if comb(p,6)==1 disp('ALVO'); end    % Posição e Velocidade Alvo
    else
        numsim = 1;
        p=tam;
        disp('Simulação Nominal (Sem Erros inseridos)');
    end
    for i = 1:numsim
        if teste
            seed = seed_teste;

```

```

else seed = i;
end
% rng('shuffle'); Faz a variação das variáveis aleatórias
rng(seed);
% Variações aleatórias adicionadas no bloco de Aerodinâmica
erro_Fx = 1;
erro_Fy = 1;
erro_Fz = 1;
erro_L = 1;
erro_M = 1;
erro_N = 1;
% Variações aleatórias adicionadas no bloco de Propulsão
erro_Fprop = 1;
desvio_FpropY = 0;
desvio_FpropZ = 0;
erro_Mf = 1;
erro_Mprop = 1;
erro_CGx = 0;
erro_CGy = 0;
erro_CGz = 0;
% Variações adicionadas no bloco de Navegação
erro_Acel = 0;
ruído_Acel = [0 0];
erro_Kgiro = 1;
erro_BIASgiro = 0;
ruído_giro = [0 0];
erro_euler_nav0x = 0;
erro_euler_nav0y = 0;
erro_euler_nav0z = 0;
erro_vb_nav0x = 0;
erro_vb_nav0y = 0;
erro_vb_nav0z = 0;
erro_Xe_nav0x = 0;
erro_Xe_nav0y = 0;
erro_Xe_nav0z = 0;
% Variações aleatórias adicionadas no bloco do Autodiretor
erro_Omegay = 0;
ruído_Omegay = [0 0];
erro_Omegaz = 0;
ruído_Omegaz = [0 0];
erro_ThetaAD = 0;
ruído_ThetaAD = [0 0];
erro_PsiAD = 0;
ruído_PsiAD = [0 0];
% Variações aleatórias adicionadas no bloco do Atuador
erro_Delta1 = 0;
erro_Delta2 = 0;
% Variações aleatórias nos dados de posição e velocidade iniciais
do alvo
    erro_Xa0 = 0;
    erro_Ya0 = 0;
    erro_Za0 = 0;
    erro_Vxa0 = 0;
    erro_Vya0 = 0;
    erro_Vza0 = 0;

% Ativar Simulação Monte Carlo
if monte_carlo == 1
    % Variações aleatórias adicionadas no bloco de Aerodinâmica
    if comb(p,1) == 1
        erro_Fx = normrnd(1,0.1/3*fator(1));
    end
end

```

```

    erro_Fy = normrnd(1,0.1/3*fator(1));
    erro_Fz = normrnd(1,0.1/3*fator(1));
    erro_L = normrnd(1,0.15/3*fator(1));
    erro_M = normrnd(1,0.15/3*fator(1));
    erro_N = normrnd(1,0.15/3*fator(1));
end
% Variações aleatórias adicionadas no bloco de Propulsão
if comb(p,2) == 1
    erro_Fprop = normrnd(1,0.1/3*fator(2));
    desvio_FpropY = normrnd(0,0.1*D2R*fator(2));
    desvio_FpropZ = normrnd(0,0.1*D2R*fator(2));
    erro_Mf = normrnd(1,0.01*fator(2));
    erro_Mprop = normrnd(1,0.01*fator(2));
    erro_CGx = normrnd(0,D.dref/10*fator(2));
    erro_CGy = normrnd(0,D.dref/10*fator(2));
    erro_CGz = normrnd(0,D.dref/10*fator(2));
end
% Variações aleatórias adicionadas no bloco de Navegação
if comb(p,3) == 1
    erro_Acel = normrnd(0,0.1/20*g*fator(3)); %
    ruído_Acel = [0 0.02*g*fator(3)];
    erro_Kgiro = normrnd(1,0.01*fator(3));
    erro_BIASgiro = normrnd(0,0.01*D2R*fator(3));
    ruído_giro = [0 0.01/3*D2R*fator(3)];
    erro_euler_nav0x = normrnd(0,0.5/2*D2R*fator(3));
    erro_euler_nav0y = normrnd(0,0.5/2*D2R*fator(3));
    erro_euler_nav0z = normrnd(0,0.5/2*D2R*fator(3));
    erro_vb_nav0x = normrnd(0,3/7*fator(3));
    erro_vb_nav0y = normrnd(0,3/7*fator(3));
    erro_vb_nav0z = normrnd(0,3/7*fator(3));
    erro_Xe_nav0x = normrnd(0,20*fator(3));
    erro_Xe_nav0y = normrnd(0,20*fator(3));
    erro_Xe_nav0z = normrnd(0,20*fator(3));
end
% Variações aleatórias adicionadas no bloco do Autodiretor
if comb(p,4) == 1
    erro_Omegay = normrnd(0,0.01*D2R*fator(4));
    ruído_Omegay = [0 0.005*D2R*fator(4)];
    erro_Omegaz = normrnd(0,0.01*D2R*fator(4));
    ruído_Omegaz = [0 0.005*D2R*fator(4)];
    erro_ThetaAD = normrnd(0,0.2*D2R*fator(4));
    ruído_ThetaAD = [0 0.01*D2R*fator(4)];
    erro_PsiAD = normrnd(0,0.2*D2R*fator(4));
    ruído_PsiAD = [0 0.01*D2R*fator(4)];
    Tciclobusca = random('unif',0,Tciclobusca);
end
% Variações aleatórias adicionadas no bloco do Atuador
if comb(p,5) == 1
    erro_Delta1 = normrnd(0,0.5*D2R*fator(5));
    erro_Delta2 = normrnd(0,0.5*D2R*fator(5));
end
% Variações nos dados de posição e velocidade do alvo
if comb(p,6) == 1
    erro_Vxa0 = normrnd(0,3*fator(6));
    erro_Vya0 = normrnd(0,3*fator(6));
    erro_Vza0 = normrnd(0,3*fator(6));
    erro_Xa0 = normrnd(0,20*fator(6));
    erro_Ya0 = normrnd(0,20*fator(6));
    erro_Za0 = normrnd(0,20*fator(6));
end

```

```

end

euler_nav0 = eul_0 + [erro_euler_nav0x erro_euler_nav0y
erro_euler_nav0z];
vb_nav0 = [D.U 0 0] + [erro_vb_nav0x erro_vb_nav0y erro_vb_nav0z];
Xe_nav0 = [0 0 -D.Alt0] + [erro_Xe_nav0x erro_Xe_nav0y
erro_Xe_nav0z];
Xa0_Virtual = Xa0 + erro_Xa0;
Ya0_Virtual = Ya0 + erro_Ya0;
Za0_Virtual = Za0 + erro_Za0;
Vxa0_Virtual = Vxa0 + erro_Vxa0;
Vya0_Virtual = Vya0 + erro_Vya0;
Vza0_Virtual = Vza0 + erro_Vza0;

% Cálculo do motor-foguete (queima radial)
% Booster
Phi_extB = D.dref - 0.005;           % Diâmetro externo do propelente
(m)
Phi_intB = Phi_extB*0.715;           % Diâmetro interno do propelente
(m)
VqueimaB = 0.0125;                   % Velocidade de queima do
propelente (m/s)
Dens_propB = 1540;                   % Densidade do propelente (kg/m3)
IspbB = 230;                         % Impulso específico do propelente
(s)
LpropB = 0.5;                        % Comprimento do propelente (m)
Vol_propB = pi/4*(Phi_extB^2 - Phi_intB^2)*LpropB; % Volume (m3)
m_prop0B = Vol_propB*Dens_propB;     % Massa (kg)
T_propB = (Phi_extB - Phi_intB)/(2*VqueimaB); % Tempo de
queima (s)
F_propB = m_prop0B*g*IspbB/T_propB; % Empuxo (N)
dmdtB = m_prop0B/T_propB;           % taxa de variação da massa (kg/s)

% Sustainer
Phi_extS = D.dref - 0.005;           % Diâmetro externo do propelente
(m)
Phi_intS = 0;                        % Diâmetro interno do propelente
(m)
VqueimaS = 0.010;                   % Velocidade de queima do
propelente (m/s)
Dens_propS = 1540;                   % Densidade do propelente (kg/m3)
IspbS = 210;                         % Impulso específico do propelente
(s)
LpropS = 0.507;                     % Comprimento do propelente (m)
Vol_propS = pi/4*(Phi_extS^2 - Phi_intS^2)*LpropS; % Volume (m3)
m_prop0S = Vol_propS*Dens_propS;     % Massa (kg)
T_propS = (LpropS)/(VqueimaS);       % Tempo de
queima (s)
F_propS = m_prop0S*g*IspbS/T_propS; % Empuxo (N)
dmdtS = m_prop0S/T_propS;           % taxa de
variação da massa (kg/s)

% Subistemas
L_AD = 0.24;                         % (m) Comprimento do AD
m_AD = 5;                           % (kg) Massa do AD
L_CDG = 0.0954+0.1284;              % (m) Comprimento da cabeça de guerra +
espoleta
m_CDG = 5+2;                        % (kg) Massa da cabeça de guerra + espoleta
L_ELE = 0.4209;                     % (m) Comprimento da Eletrônica + atuadores
m_ELE = 9;                          % (kg) Massa da Eletrônica + atuadores
L_MOT = LpropB + .1 + LpropS;       % (m) Comprimento do Propulsor

```

```

    m_MOTB = m_prop0B/0.7;           % (kg) Massa do Propulsor + massa
da estrutura
    m_MOTS = m_prop0S/0.8+5;

    D.m = m_AD + m_CDG + m_ELE + m_MOTB + m_MOTS;   % (kg) Massa do
missil
    D.L = L_AD + L_CDG + L_ELE + L_MOT;             % (m) Comprimento
do míssil

    D.m_prop0 = m_prop0B+m_prop0S; % (kg) Massa de propelente

    D.T_propB = T_propB;
    D.T_propS = T_propS;

    D.F_propB = F_propB;
    D.F_propS = F_propS;
    D.dmdtB = dmdtB;
    D.dmdtS = dmdtS;
    D.mf = D.m - D.m_prop0;           % (kg) Massa após a queima

% Cálculo do CG
x_cg_AD = -(L_AD/2);
x_cg_ELE = -(L_AD + L_ELE/2);
x_cg_CDG = -(L_AD + L_CDG/2 + L_ELE);
x_cg_MOTS = -(L_AD + L_CDG + L_ELE + LpropS/2);
x_cg_MOTB = -(L_AD + L_CDG + L_ELE + LpropS + LpropB/2);

    D.x_cg0 = (x_cg_AD*m_AD + x_cg_CDG*m_CDG + x_cg_ELE*m_ELE +
x_cg_MOTB*m_MOTB + x_cg_MOTS*m_MOTS)/D.m;
    y_cg0 = 0;
    z_cg0 = 0;

    D.x_cgBm = (x_cg_AD*m_AD + x_cg_CDG*m_CDG + x_cg_ELE*m_ELE + ...
x_cg_MOTB*(m_MOTB-m_prop0B/2) + x_cg_MOTS*m_MOTS)/(D.m-
m_prop0B/2);

    D.x_cgBf = (x_cg_AD*m_AD + x_cg_CDG*m_CDG + x_cg_ELE*m_ELE + ...
x_cg_MOTB*(m_MOTB-m_prop0B) + x_cg_MOTS*m_MOTS)/(D.m-m_prop0B);

    D.x_cgSm = (x_cg_AD*m_AD + x_cg_CDG*m_CDG + x_cg_ELE*m_ELE + ...
x_cg_MOTB*(m_MOTB-m_prop0B) + x_cg_MOTS*(m_MOTS-m_prop0S) + ...
(x_cg_MOTS+LpropS/4)*m_prop0S/2)/(D.m-m_prop0B-m_prop0S/2);

    D.x_cgf = (x_cg_AD*m_AD + x_cg_CDG*m_CDG + x_cg_ELE*m_ELE + ...
x_cg_MOTB*(m_MOTB-m_prop0B)+x_cg_MOTS*(m_MOTS-m_prop0S))/D.mf;

    y_cgf = 0;
    z_cgf = 0;

    D.CRM = [D.x_cgf  0  0]' - 0*[D.dref 0  0]';

% Cálculo das matrizes de inércia Ixx
    D.Ix0 = D.m*D.R^2/2;           % Inércia Ixx0 (kg.m2)
    D.IxBm = (D.m-m_prop0B/2)*D.R^2/2;
    D.IxBf = (D.m-m_prop0B)*D.R^2/2;
    D.IxSm = (D.m-m_prop0B-m_prop0S)*D.R^2/2;
    D.Ixf = D.mf*D.R^2/2;           % Inércia Ixxf (kg.m2)

% Cálculo de Iyy de cada subsistema em relação ao CG inicial

```

```

Iy0_AD = (D.R^2/4 + L_AD^2/12 + (D.xcg0 - xcg_AD)^2)*m_AD;
Iy0_CDG = (D.R^2/4 + L_CDG^2/12 + (D.xcg0 - xcg_CDG)^2)*m_CDG;
Iy0_ELE = (D.R^2/4 + L_ELE^2/12 + (D.xcg0 - xcg_ELE)^2)*m_ELE;
Iy0_MOTB = (D.R^2/4 + LpropB^2/12 + (D.xcg0 - xcg_MOTB)^2)*m_MOTB;
Iy0_MOTS = (D.R^2/4 + LpropS^2/12 + (D.xcg0 - xcg_MOTS)^2)*m_MOTS;
D.Iy0 = Iy0_AD + Iy0_CDG + Iy0_ELE + Iy0_MOTB + Iy0_MOTS;

IyBm_AD = (D.R^2/4 + L_AD^2/12 + (D.xcgBm - xcg_AD)^2)*m_AD;
IyBm_CDG = (D.R^2/4 + L_CDG^2/12 + (D.xcgBm - xcg_CDG)^2)*m_CDG;
IyBm_ELE = (D.R^2/4 + L_ELE^2/12 + (D.xcgBm - xcg_ELE)^2)*m_ELE;
IyBm_MOTB = (D.R^2/4 + LpropB^2/12 + (D.xcgBm -
xcg_MOTB)^2)*(m_MOTB-m_prop0B/2);
IyBm_MOTS = (D.R^2/4 + LpropS^2/12 + (D.xcgBm -
xcg_MOTS)^2)*m_MOTS;
D.IyBm = IyBm_AD + IyBm_CDG + IyBm_ELE + IyBm_MOTB + IyBm_MOTS;

IyBf_AD = (D.R^2/4 + L_AD^2/12 + (D.xcgBf - xcg_AD)^2)*m_AD;
IyBf_CDG = (D.R^2/4 + L_CDG^2/12 + (D.xcgBf - xcg_CDG)^2)*m_CDG;
IyBf_ELE = (D.R^2/4 + L_ELE^2/12 + (D.xcgBf - xcg_ELE)^2)*m_ELE;
IyBf_MOTB = (D.R^2/4 + LpropB^2/12 + (D.xcgBf -
xcg_MOTB)^2)*(m_MOTB-m_prop0B);
IyBf_MOTS = (D.R^2/4 + LpropS^2/12 + (D.xcgBf -
xcg_MOTS)^2)*m_MOTS;
D.IyBf = IyBf_AD + IyBf_CDG + IyBf_ELE + IyBf_MOTB + IyBf_MOTS;

IySm_AD = (D.R^2/4 + L_AD^2/12 + (D.xcgSm - xcg_AD)^2)*m_AD;
IySm_CDG = (D.R^2/4 + L_CDG^2/12 + (D.xcgSm - xcg_CDG)^2)*m_CDG;
IySm_ELE = (D.R^2/4 + L_ELE^2/12 + (D.xcgSm - xcg_ELE)^2)*m_ELE;
IySm_MOTB = (D.R^2/4 + LpropB^2/12 + (D.xcgSm -
xcg_MOTB)^2)*(m_MOTB-m_prop0B);
IySm_MOTS = (D.R^2/4 + LpropS^2/12 + (D.xcgSm -
xcg_MOTS)^2)*(m_MOTS-m_prop0S) + (D.R^2/4 + (LpropS/2)^2/12 + (D.xcgSm -
(xcg_MOTS+LpropS/4))^2)*m_prop0S/2;
D.IySm = IySm_AD + IySm_CDG + IySm_ELE + IySm_MOTB + IySm_MOTS;

% Cálculo de Iyy de cada subsistema em relação ao CG final
Iyf_AD = (D.R^2/4 + L_AD^2/12 + (D.xcgf - xcg_AD)^2)*m_AD;
Iyf_CDG = (D.R^2/4 + L_CDG^2/12 + (D.xcgf - xcg_CDG)^2)*m_CDG;
Iyf_ELE = (D.R^2/4 + L_ELE^2/12 + (D.xcgf - xcg_ELE)^2)*m_ELE;
Iyf_MOTB = (D.R^2/4 + LpropB^2/12 + (D.xcgf - xcg_MOTB)^2)*(m_MOTB
- m_prop0B);
Iyf_MOTS = (D.R^2/4 + LpropS^2/12 + (D.xcgf - xcg_MOTS)^2)*(m_MOTS
- m_prop0S);
D.Iyf = Iyf_AD + Iyf_CDG + Iyf_ELE + Iyf_MOTB + Iyf_MOTS;
D.I0 = [D.Ix0 0 0; ...
0 D.Iy0 0; ...
0 0 D.Iy0];
D.IBm = [D.IxBm 0 0; ...
0 D.IyBm 0; ...
0 0 D.IyBm];
D.IBf = [D.IxBf 0 0; ...
0 D.IyBf 0; ...
0 0 D.IyBf];
D.ISm = [D.IxSm 0 0; ...
0 D.IySm 0; ...
0 0 D.IySm];
D.If = [D.Ixf 0 0; ...
0 D.Iyf 0; ...
0 0 D.Iyf];
% Autodiretor
wn_AD = 2*pi*30;

```

```

qsi_AD = sqrt(2)/2;
Tresp = .1;

% Guiamento
AltCruz = Za0; % (m) Altitude cruzeiro
RampaMax = 20*D2R; % (rad) Rampa máxima de mudança de
altitude
GanhoAtit = .43*D2R; % (rad/m) Ganho de controle de atitude
AclnMax = 1000;

anp = 3.5; % Ganho de navegação proporcional
% Atuador
AT.Km = 0.09*1.37;
AT.Rm = 2.29;
AT.Kt = 0.134;
AT.J = 31.5 * 10^-6;
AT.Bm = 3.53 * 10^-3;
AT.Kp = 5.97*2.35;
AT.N1 = 3.28;
AT.N2 = 4;
AT.At = 1.5;
AT.Ap = 50;
wn = sqrt(AT.Ap*AT.Kp*AT.Km/AT.J/AT.N1/AT.N2/AT.Rm);
qsi =
(AT.N1*(AT.Bm*AT.Rm+AT.Km^2)+AT.Km*AT.Kt*AT.At)/AT.J/AT.N1/AT.Rm/2/wn;
CTI11 = cos(psi0)*cos(theta0);
CTI12 = sin(psi0)*cos(theta0);
CTI13 = -sin(theta0);
CTI21 = cos(psi0)*sin(theta0)*sin(phi0)-sin(psi0)*cos(phi0);
CTI22 = sin(psi0)*sin(theta0)*sin(phi0)+cos(psi0)*cos(phi0);
CTI23 = cos(theta0)*sin(phi0);
CTI31 = cos(psi0)*sin(theta0)*cos(phi0)+sin(psi0)*sin(phi0);
CTI32 = sin(psi0)*sin(theta0)*cos(phi0)-cos(psi0)*sin(phi0);
CTI33 = cos(theta0)*cos(phi0);
CTI = [CTI11 CTI12 CTI13; CTI21 CTI22 CTI23; CTI31 CTI32 CTI33];

Xe_0 = [0 0 -D.Alt0]';
Xa_0 = [Xa0 Ya0 Za0]';
delta0 = CTI*(Xa_0 - Xe_0);
thetaAD0 = -atan2(delta0(3),delta0(1));
psiAD0 = atan2(delta0(2),delta0(1));

% Dados aerodinâmicos
load('M_aed_3d.mat');

% Autopiloto
load('autopiloto_final.mat');

TTrav = 0.50;

sim('Sim_6DOF_completo.slx');

MissDist = sqrt((Xe.Data(end,1) - Xa.Data(end,1))^2 + ...
(Xe.Data(end,2) - Xa.Data(end,2))^2 + ...
(Xe.Data(end,3) - Xa.Data(end,3))^2);
if monte_carlo
    disp(['Final Distance: ' num2str(MissDist) ' m, seed: '
num2str(seed) ', p: ' num2str(p)]);
else disp(['Final Distance: ' num2str(MissDist) ' m']);
end

```

```

data_monte_carlo.MissDist(p,i) = MissDist;

end
flag=0;
if completo==0
    %Verifica no conjunto atual das numsim simulações se alguma teve
    %distância de passagem maior que 7 metros
    kill=0;
    for j=1:numsim
        if data_monte_carlo.MissDist(p,j) > 7 %distancia de passagem
maior que 7 metros
            %flag=1;
            pos_flag(p,j)=1;
            %envelope(thetaApos,Xa0pos)=0;
        else
            %envelope(thetaApos,Xa0pos)=1;
            kill = kill +1;
        end
    end

    end
    kill = kill*100/qtd_sim;
    if kill < 70
        flag=1;
    end

    %Procedimento quando a distância de passagem foi maior que o limite
    %estabelecido
    if otimiza>0
        if flag==1
            maior=0;
            %Procedimento para identificar qual fator de multiplicação
deve
            %ser alterado para se rodar a simulação novamente
            for i=1:num_var
                if comb(p,i)==1
                    if maior==0
                        maior=i;
                        %Critério escolhido para escolher qual fator
                        %alterar: Diminuir em 10% o fator multiplicativo
do
                        %maior fator entre os blocos envolvidos
                        elseif fator(i)>fator(maior)
                            maior=i;
                        end
                    end
                end
            end
            fator(maior)=fator(maior)-0.1;
        else p=p+1;
        end
    else p=p+1;
    end
end

if teste | monte_carlo == 0
    p = tam+1;
end

fator;

```

```

    if completo~=0
        p=tam+1;
    end
    if p==tam && completo~=0
        p=tam+1;
    end
end
end

```

B.3 Código MATLAB para plotar a tabela de kill probability

```

kill = zeros(tam,1);

for p=1:tam
    for j=1:qtd_sim
        if data_monte_carlo.MissDist(p,j)<=7
            kill(p) = kill(p) +1;
        end
    end
end
kill = kill*100/qtd_sim;

for p=1:tam
    if kill(p) >= 80
        disp([num2str(kill(p)), '% = objective']);
    elseif kill(p) >= 70
        disp([num2str(kill(p)), '% = threshold']);
    else disp([num2str(kill(p)), '% = miss']);
    end
end

for p=1:tam
    disp([num2str(kill(p)), '%']);
end
end

```

Anexo – Código MATLAB utilizado para Simular Voo sem Erros Inseridos (PEIXOTO, 2018)

```
%Simulação unitária do voo do míssil sem inserção de Erros

clear all;
close all;

Altura0 = 1500;           % Altura de lançamento
dist_manobra = 1000;     % distância em que o alvo inicia a manobra evasiva
V0 = 120;                 % (m/s) Modulo da Velocidade inicial do alvo

manobra = 5;              % Tipo de manobra evasiva: 4 -> Subir 5 ->
Descer

ng = 4;                   % fator de carga

g = 9.80665;              % gravidade

D2R = pi/180;             % Constante graus -> rad
R2D = 1/D2R;

Tfim = 100;               % Tempo de simulação (s)

Dlt_max = 10*pi/180;      % Deflexão máxima do atuador
Dlt_max_roll = 2*pi/180;

D.Gama0 = 0*D2R;          % Angulo inicial
D.Alt0 = Altura0;         % (m) altitude inicial
D.Mach0 = 0.05;           % Velocidade de lançamento

p0 = 0*D2R;
q0 = 0*D2R;
r0 = 0*D2R;

phi0 = 0;
theta0 = 0;
psi0 = 0;
eul_0 = [phi0 theta0 psi0];
eul_statename = '';

% Alvo
Xa0 = 1000;                % Distancia inicial do alvo
Ya0 = 0;
Za0 = -Altura0;           % Za0 negativo acima do solo
thetaA = 270*D2R;         % Direção do alvo (0 -> alvo afastando; 180*D2R ->
alvo aproximando
% 270*D2R -> alvo cruzando a proa do lançador perpendicularmente
Vxa0 = V0*cos(thetaA);
Vya0 = -V0*sin(thetaA);
Vza0 = 0;

[~,vsom,~,rho] = atmosisa(D.Alt0);
```

```

D.U = D.Mach0*vsom;      % Velocidade inicial
q = 1/2*rho*(D.U)^2;     % Pressão dinâmica

% Dados do míssil
D.dref = 0.147;          % Diâmetro (m)
D.c = D.dref;            % Comprimento de referência
D.S = pi*D.dref^2/4;     % Área de referência
D.R = D.dref/2;          % Raio (m)

% Características e Restrições (Parâmetros) do AutoDiretor
thetamaxAD = 30*D2R;
psimaxAD = 30*D2R;
sigmapYmaxAD = 17*D2R;   % graus/s
sigmapZmaxAD = 17*D2R;   % graus/s
Rsat = 150;              % Distancia em metros em que o radar fica cego
% Fator de correção na carga lateral, Fy e Fz
KAcInMax = 1.5;          % Ganho Aceleração lateral
Tciclobusca = 4;         % (s) tempo para o seeker encontrar o alvo
Alcance_Laser = 6000;    % Alcance Laser, antes dessa distância o míssil
                           persegue o alvo virtual

% Variações aleatórias adicionadas no bloco de Aerodinâmica
erro_Fx = 1;
erro_Fy = 1;
erro_Fz = 1;
erro_L = 1;
erro_M = 1;
erro_N = 1;
% Variações aleatórias adicionadas no bloco de Propulsão
erro_Fprop = 1;
desvio_FpropY = 0;
desvio_FpropZ = 0;
erro_Mf = 1;
erro_Mprop = 1;
erro_CGx = 0;
erro_CGy = 0;
erro_CGz = 0;
% Variações adicionadas no bloco de Navegação
erro_Acel = 0;
ruído_Acel = [0 0];
erro_Kgiro = 1;
erro_BIASgiro = 0;
ruído_giro = [0 0];
erro_euler_nav0x = 0;
erro_euler_nav0y = 0;
erro_euler_nav0z = 0;
erro_vb_nav0x = 0;
erro_vb_nav0y = 0;
erro_vb_nav0z = 0;
erro_Xe_nav0x = 0;
erro_Xe_nav0y = 0;
erro_Xe_nav0z = 0;
% Variações aleatórias adicionadas no bloco do Autodiretor
erro_Omegay = 0;
ruído_Omegay = [0 0];
erro_Omegaz = 0;
ruído_Omegaz = [0 0];
erro_ThetaAD = 0;
ruído_ThetaAD = [0 0];
erro_PsiAD = 0;

```

```

ruído_PsiAD = [0 0];
% Variações aleatórias adicionadas no bloco do Atuador
erro_Delta1 = 0;
erro_Delta2 = 0;
% Variações aleatórias nos dados de posição e velocidade iniciais do alvo
erro_Xa0 = 0;
erro_Ya0 = 0;
erro_Za0 = 0;
erro_Vxa0 = 0;
erro_Vya0 = 0;
erro_Vza0 = 0;

euler_nav0 = eul_0 + [erro_euler_nav0x erro_euler_nav0y erro_euler_nav0z];
vb_nav0 = [D.U 0 0] + [erro_vb_nav0x erro_vb_nav0y erro_vb_nav0z];
Xe_nav0 = [0 0 -D.Alt0] + [erro_Xe_nav0x erro_Xe_nav0y erro_Xe_nav0z];
Xa0_Virtual = Xa0 + erro_Xa0;
Ya0_Virtual = Ya0 + erro_Ya0;
Za0_Virtual = Za0 + erro_Za0;
Vxa0_Virtual = Vxa0 + erro_Vxa0;
Vya0_Virtual = Vya0 + erro_Vya0;
Vza0_Virtual = Vza0 + erro_Vza0;

% Cálculo do motor-foguete (queima radial)
% Booster
Phi_extB = D.dref - 0.005;           % Diâmetro externo do propelente (m)
Phi_intB = Phi_extB*0.715;           % Diâmetro interno do propelente (m)
VqueimaB = 0.0125;                   % Velocidade de queima do propelente (m/s)
Dens_propB = 1540;                   % Densidade do propelente (kg/m3)
IspbB = 230;                         % Impulso específico do propelente (s)
LpropB = 0.5;                        % Comprimento do propelente (m)
Vol_propB = pi/4*(Phi_extB^2 - Phi_intB^2)*LpropB; % Volume (m3)
m_prop0B = Vol_propB*Dens_propB;     % Massa (kg)
T_propB = (Phi_extB - Phi_intB)/(2*VqueimaB); % Tempo de queima (s)
F_propB = m_prop0B*g*IspbB/T_propB;  % Empuxo (N)
dmdtB = m_prop0B/T_propB;            % taxa de variação da massa (kg/s)

% Sustainer
Phi_extS = D.dref - 0.005;           % Diâmetro externo do propelente (m)
Phi_intS = 0;                        % Diâmetro interno do propelente (m)
VqueimaS = 0.010;                   % Velocidade de queima do propelente (m/s)
Dens_propS = 1540;                   % Densidade do propelente (kg/m3)
IspbS = 210;                         % Impulso específico do propelente (s)
LpropS = 0.507;                     % Comprimento do propelente (m)
Vol_propS = pi/4*(Phi_extS^2 - Phi_intS^2)*LpropS; % Volume (m3)
m_prop0S = Vol_propS*Dens_propS;     % Massa (kg)
T_propS = (LpropS)/(VqueimaS);       % Tempo de queima (s)
F_propS = m_prop0S*g*IspbS/T_propS;  % Empuxo (N)
dmdtS = m_prop0S/T_propS;            % taxa de variação da
massa (kg/s)

% Subistemas
L_AD = 0.24;                         % (m) Comprimento do AD
m_AD = 5;                            % (kg) Massa do AD
L_CDG = 0.0954+0.1284;               % (m) Comprimento da cabeça de guerra + espoleta
m_CDG = 5+2;                         % (kg) Massa da cabeça de guerra + espoleta
L_ELE = 0.4209;                      % (m) Comprimento da Eletrônica + atuadores
m_ELE = 9;                           % (kg) Massa da Eletrônica + atuadores
L_MOT = LpropB + .1 + LpropS;         % (m) Comprimento do Propulsor
m_MOTB = m_prop0B/0.7;                % (kg) Massa do Propulsor + massa da
estrutura
m_MOTS = m_prop0S/0.8+5;

```

```

D.m = m_AD + m_CDG + m_ELE + m_MOTB + m_MOTS;    % (kg) Massa do míssil
D.L = L_AD + L_CDG + L_ELE + L_MOT;              % (m) Comprimento do míssil

D.m_prop0 = m_prop0B+m_prop0S; % (kg) Massa de propelente

D.T_propB = T_propB;
D.T_propS = T_propS;

D.F_propB = F_propB;
D.F_propS = F_propS;
D.dmdtB = dmdtB;
D.dmdtS = dmdtS;
D.mf = D.m - D.m_prop0;          % (kg) Massa após a queima

% Cálculo do CG
xcg_AD = -(L_AD/2);
xcg_ELE = -(L_AD + L_ELE/2);
xcg_CDG = -(L_AD + L_CDG/2 + L_ELE);
xcg_MOTS = -(L_AD + L_CDG + L_ELE + LpropS/2);
xcg_MOTB = -(L_AD + L_CDG + L_ELE + LpropS + LpropB/2);

D.xcg0 = (xcg_AD*m_AD + xcg_CDG*m_CDG + xcg_ELE*m_ELE + xcg_MOTB*m_MOTB +
xcg_MOTS*m_MOTS)/D.m;
ycg0 = 0;
zcg0 = 0;

D.xcgBm = (xcg_AD*m_AD + xcg_CDG*m_CDG + xcg_ELE*m_ELE + ...
xcg_MOTB*(m_MOTB-m_prop0B/2) + xcg_MOTS*m_MOTS)/(D.m-m_prop0B/2);

D.xcgBf = (xcg_AD*m_AD + xcg_CDG*m_CDG + xcg_ELE*m_ELE + ...
xcg_MOTB*(m_MOTB-m_prop0B) + xcg_MOTS*m_MOTS)/(D.m-m_prop0B);

D.xcgSm = (xcg_AD*m_AD + xcg_CDG*m_CDG + xcg_ELE*m_ELE + ...
xcg_MOTB*(m_MOTB-m_prop0B) + xcg_MOTS*(m_MOTS-m_prop0S) + ...
(xcg_MOTS+LpropS/4)*m_prop0S/2)/(D.m-m_prop0B-m_prop0S/2);

D.xcgf = (xcg_AD*m_AD + xcg_CDG*m_CDG + xcg_ELE*m_ELE + ...
xcg_MOTB*(m_MOTB-m_prop0B)+xcg_MOTS*(m_MOTS-m_prop0S))/D.mf;

ycgf = 0;
zcgf = 0;

D.CRM = [D.xcgf  0  0]' - 0*[D.dref 0  0]';

% Cálculo das matrizes de inércia Ixx
D.Ix0 = D.m*D.R^2/2;          % Inércia Ixx0 (kg.m2)
D.IxBm = (D.m-m_prop0B/2)*D.R^2/2;
D.IxBf = (D.m-m_prop0B)*D.R^2/2;
D.IxSm = (D.m-m_prop0B-m_prop0S)*D.R^2/2;
D.Ixf = D.mf*D.R^2/2;          % Inércia Ixxf (kg.m2)

% Cálculo de Iyy de cada subsistema em relação ao CG inicial
Iy0_AD = (D.R^2/4 + L_AD^2/12 + (D.xcg0 - xcg_AD)^2)*m_AD;
Iy0_CDG = (D.R^2/4 + L_CDG^2/12 + (D.xcg0 - xcg_CDG)^2)*m_CDG;
Iy0_ELE = (D.R^2/4 + L_ELE^2/12 + (D.xcg0 - xcg_ELE)^2)*m_ELE;
Iy0_MOTB = (D.R^2/4 + LpropB^2/12 + (D.xcg0 - xcg_MOTB)^2)*m_MOTB;
Iy0_MOTS = (D.R^2/4 + LpropS^2/12 + (D.xcg0 - xcg_MOTS)^2)*m_MOTS;
D.Iy0 = Iy0_AD + Iy0_CDG + Iy0_ELE + Iy0_MOTB + Iy0_MOTS;

```

```

IyBm_AD = (D.R^2/4 + L_AD^2/12 + (D.xcgBm - xcg_AD)^2)*m_AD;
IyBm_CDG = (D.R^2/4 + L_CDG^2/12 + (D.xcgBm - xcg_CDG)^2)*m_CDG;
IyBm_ELE = (D.R^2/4 + L_ELE^2/12 + (D.xcgBm - xcg_ELE)^2)*m_ELE;
IyBm_MOTB = (D.R^2/4 + LpropB^2/12 + (D.xcgBm - xcg_MOTB)^2)*(m_MOTB-
m_prop0B/2);
IyBm_MOTS = (D.R^2/4 + LpropS^2/12 + (D.xcgBm - xcg_MOTS)^2)*m_MOTS;
D.IyBm = IyBm_AD + IyBm_CDG + IyBm_ELE + IyBm_MOTB + IyBm_MOTS;

```

```

IyBf_AD = (D.R^2/4 + L_AD^2/12 + (D.xcgBf - xcg_AD)^2)*m_AD;
IyBf_CDG = (D.R^2/4 + L_CDG^2/12 + (D.xcgBf - xcg_CDG)^2)*m_CDG;
IyBf_ELE = (D.R^2/4 + L_ELE^2/12 + (D.xcgBf - xcg_ELE)^2)*m_ELE;
IyBf_MOTB = (D.R^2/4 + LpropB^2/12 + (D.xcgBf - xcg_MOTB)^2)*(m_MOTB-
m_prop0B);
IyBf_MOTS = (D.R^2/4 + LpropS^2/12 + (D.xcgBf - xcg_MOTS)^2)*m_MOTS;
D.IyBf = IyBf_AD + IyBf_CDG + IyBf_ELE + IyBf_MOTB + IyBf_MOTS;

```

```

IySm_AD = (D.R^2/4 + L_AD^2/12 + (D.xcgSm - xcg_AD)^2)*m_AD;
IySm_CDG = (D.R^2/4 + L_CDG^2/12 + (D.xcgSm - xcg_CDG)^2)*m_CDG;
IySm_ELE = (D.R^2/4 + L_ELE^2/12 + (D.xcgSm - xcg_ELE)^2)*m_ELE;
IySm_MOTB = (D.R^2/4 + LpropB^2/12 + (D.xcgSm - xcg_MOTB)^2)*(m_MOTB-
m_prop0B);
IySm_MOTS = (D.R^2/4 + LpropS^2/12 + (D.xcgSm - xcg_MOTS)^2)*(m_MOTS-
m_prop0S) + (D.R^2/4 + (LpropS/2)^2/12 + (D.xcgSm -
(xcg_MOTS+LpropS/4))^2)*m_prop0S/2;
D.IySm = IySm_AD + IySm_CDG + IySm_ELE + IySm_MOTB + IySm_MOTS;

```

% Cálculo de Iyy de cada subsistema em relação ao CG final

```

Iyf_AD = (D.R^2/4 + L_AD^2/12 + (D.xcgf - xcg_AD)^2)*m_AD;
Iyf_CDG = (D.R^2/4 + L_CDG^2/12 + (D.xcgf - xcg_CDG)^2)*m_CDG;
Iyf_ELE = (D.R^2/4 + L_ELE^2/12 + (D.xcgf - xcg_ELE)^2)*m_ELE;
Iyf_MOTB = (D.R^2/4 + LpropB^2/12 + (D.xcgf - xcg_MOTB)^2)*(m_MOTB -
m_prop0B);
Iyf_MOTS = (D.R^2/4 + LpropS^2/12 + (D.xcgf - xcg_MOTS)^2)*(m_MOTS -
m_prop0S);
D.Iyf = Iyf_AD + Iyf_CDG + Iyf_ELE + Iyf_MOTB + Iyf_MOTS;

```

```

D.IO = [D.Ix0 0 0; ...
0 D.Iy0 0; ...
0 0 D.Iy0];

```

```

D.IBm = [D.IxBm 0 0; ...
0 D.IyBm 0; ...
0 0 D.IyBm];

```

```

D.IBf = [D.IxBf 0 0; ...
0 D.IyBf 0; ...
0 0 D.IyBf];

```

```

D.ISm = [D.IxSm 0 0; ...
0 D.IySm 0; ...
0 0 D.IySm];

```

```

D.If = [D.Ixf 0 0; ...
0 D.Iyf 0; ...
0 0 D.Iyf];

```

% Autodiretor

```

wn_AD = 2*pi*30;

```

```

qsi_AD = sqrt(2)/2;
Tresp = .1;

% Guiamento
AltCruz = Za0; % (m) Altitude cruzeiro
RampaMax = 20*D2R; % (rad) Rampa máxima de mudança de altitude
GanhoAtit = .43*D2R; % (rad/m) Ganho de controle de attitude
AcInMax = 1000;

anp = 3.5; % Ganho de navegação proporcional
% Atuador
AT.Km = 0.09*1.37;
AT.Rm = 2.29;
AT.Kt = 0.134;
AT.J = 31.5 * 10^-6;
AT.Bm = 3.53 * 10^-3;
AT.Kp = 5.97*2.35;
AT.N1 = 3.28;
AT.N2 = 4;
AT.At = 1.5;
AT.Ap = 50;
wn = sqrt(AT.Ap*AT.Kp*AT.Km/AT.J/AT.N1/AT.N2/AT.Rm);
qsi =
(AT.N1*(AT.Bm*AT.Rm+AT.Km^2)+AT.Km*AT.Kt*AT.At)/AT.J/AT.N1/AT.Rm/2/wn;

CTI11 = cos(psi0)*cos(theta0);
CTI12 = sin(psi0)*cos(theta0);
CTI13 = -sin(theta0);
CTI21 = cos(psi0)*sin(theta0)*sin(phi0)-sin(psi0)*cos(phi0);
CTI22 = sin(psi0)*sin(theta0)*sin(phi0)+cos(psi0)*cos(phi0);
CTI23 = cos(theta0)*sin(phi0);
CTI31 = cos(psi0)*sin(theta0)*cos(phi0)+sin(psi0)*sin(phi0);
CTI32 = sin(psi0)*sin(theta0)*cos(phi0)-cos(psi0)*sin(phi0);
CTI33 = cos(theta0)*cos(phi0);
CTI = [CTI11 CTI12 CTI13; CTI21 CTI22 CTI23; CTI31 CTI32 CTI33];

Xe_0 = [0 0 -D.Alt0]';
Xa_0 = [Xa0 Ya0 Za0]';
delta0 = CTI*(Xa_0 - Xe_0);
thetaAD0 = -atan2(delta0(3),delta0(1));
psiAD0 = atan2(delta0(2),delta0(1));

% Dados aerodinâmicos
load('M_aed_3d.mat');

% Autopiloto
load('autopiloto_final.mat');

TTrav = 0.50;

i=1;
sim('Sim_6DOF_completo.slx');

MissDist = sqrt((Xe.Data(end,1) - Xa.Data(end,1))^2 + ...
(Xe.Data(end,2) - Xa.Data(end,2))^2 + ...
(Xe.Data(end,3) - Xa.Data(end,3))^2);

disp(['Final Distance: ' num2str(MissDist) ' m']);

```

FOLHA DE REGISTRO DO DOCUMENTO			
1. CLASSIFICAÇÃO/TIPO TC	2. DATA 18 de novembro de 2019	3. REGISTRO Nº DCTA/ITA/TC-096/2019	4. Nº DE PÁGINAS 99
5. TÍTULO E SUBTÍTULO: Desenvolvimento de uma Simulação Monte Carlo para míssil ar-ar guiado a laser			
6. AUTOR(ES): Hilton Alex Soares de Lima			
7. INSTITUIÇÃO(ÕES)/ÓRGÃO(S) INTERNO(S)/DIVISÃO(ÕES): Instituto Tecnológico de Aeronáutica – ITA			
8. PALAVRAS-CHAVE SUGERIDAS PELO AUTOR: 1. Mísseis. 2. Monte Carlo. 3. Simulação.			
9. PALAVRAS-CHAVE RESULTANTES DE INDEXAÇÃO: Mísseis guiados; Método de Monte Carlo; Simulação computadorizada; Sensores; Engenharia aeroespacial.			
10. APRESENTAÇÃO: X Nacional Internacional ITA, São José dos Campos. Curso de Graduação em Engenharia Aeroespacial. Orientador: Prof. Dr. Luís Eduardo Vergueiro Loures da Costa; coorientador: Engº Carlos Alberto de Paiva Carvalho. Publicado em 2019.			
11. RESUMO: O objetivo desse trabalho é desenvolver uma simulação monte carlo para um míssil ar-ar guiado a laser com a finalidade de determinar os requisitos dos principais subsistemas do míssil sem inviabilizar a construção do mesmo e ao mesmo tempo garantir uma confiabilidade razoável (elevada kill probability), levando-se em consideração erros e ruídos oriundos das leituras de sensores e erros na execução de atuadores e na propulsão do míssil. Foi utilizado como base para o desenvolvimento da simulação o trabalho de graduação [1] desta maneira foram implementadas algumas das sugestões sugeridas pelo trabalho citado. A simulação levou em consideração a inserção de erros do tipo run-to-run e in-run nos seguintes subsistemas, aerodinâmica, propulsão, navegação, autodiretor, atuador e detecção da posição e velocidade do alvo. Além disso foi estudado os casos em que há limitações de leitura de dados como posição e velocidade de tracking do alvo pelo autodiretor e da distância de saturação do autodiretor, tais limitações modelam o que ocorre num voo real.			
12. GRAU DE SIGILO: (X) OSTENSIVO () RESERVADO () SECRETO			