ESTRATÉGIAS DE ACOMPANHAMENTO DE MÚLTIPLOS ALVOS: UMA COMPARAÇÃO ENTRE AS ABORDAGENS DE ASSOCIAÇÃO DE DADOS MHT E REDES NEURAIS

Manuel Ramón Vargas Avila

Dissertação de Mestrado apresentada ao Programa de Pós-graduação em Engenharia Elétrica, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Mestre em Engenharia Elétrica.

Orientadores: Luiz Pereira Calôba
Beatriz de Souza Leite Pires de Lima

Rio de Janeiro
Abril de 2015

ESTRATÉGIAS DE ACOMPANHAMENTO DE MÚLTIPLOS ALVOS: UMA COMPARAÇÃO ENTRE AS ABORDAGENS DE ASSOCIAÇÃO DE DADOS MHT E REDES NEURAIS

Manuel Ramón Vargas Avila

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA (COPPE) DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA ELÉTRICA.

Examinada por:

Prof. Luiz Pereira Calôba, Dr.Ing.

Prof^{a}. Beatriz de Souza Leite Pires de Lima, D.Sc.

Prof. Alexandre Gonçalves Evsukoff, Dr.

Prof. Rubens Lopes de Oliveira, D.Sc.

RIO DE JANEIRO, RJ – BRASIL
ABRIL DE 2015

Vargas Avila, Manuel Ramón

Estratégias de acompanhamento de múltiplos alvos: uma comparação entre as abordagens de associação de dados MHT e Redes Neurais/Manuel Ramón Vargas Avila. – Rio de Janeiro: UFRJ/COPPE, 2015.

XV, 162 p.: il.; 29, 7cm.

Orientadores: Luiz Pereira Calôba

Beatriz de Souza Leite Pires de Lima

Dissertação (mestrado) – UFRJ/COPPE/Programa de Engenharia Elétrica, 2015.

Referências Bibliográficas: p. 146 – 150.

1. Acompanhamento de múltiplos alvos. 2. Redes neurais. 3. Filtragem IMM. 4. Filtro de Kalman. 5. Multi Hypothesis Tracking. I. Calôba, Luiz Pereira *et al.*. II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia Elétrica. III. Título.

Este trabalho é dedicado a Deus por me permitir estar aqui, aos meus pais e irmãos pela educação e apoio, e à minha namorada Natalia pela paciência e apoio incondicional.

Agradecimentos

Aos meus orientadores Prof. Luiz Pereira Calôba e Profª. Beatriz de Souza Leite Pires de Lima, por sua paciência e orientação na elaboração deste trabalho.
Ao Instituto de Pesquisa da Marinha (IPqM), pela oportunidade de realizar este trabalho.
Aos colegas, Rubens Oliveira e Cachimo Assane, pela sua amizade e explicações sobre conceitos utilizados na elaboração deste trabalho.
Aos meus Professores do mestrado, pelo conhecimento compartilhado.
À CAPES pelo financiamento do mestrado.
À fundação COPPETEC pelo financiamento deste projeto.

Resumo da Dissertação apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

ESTRATÉGIAS DE ACOMPANHAMENTO DE MÚLTIPLOS ALVOS: UMA COMPARAÇÃO ENTRE AS ABORDAGENS DE ASSOCIAÇÃO DE DADOS MHT E REDES NEURAIS

Manuel Ramón Vargas Avila

Abril/2015

Orientadores: Luiz Pereira Calôba
Beatriz de Souza Leite Pires de Lima

Programa: Engenharia Elétrica

Esta dissertação apresenta uma metodologia de acompanhamento de múltiplos alvos a partir de informações provenientes de um sensor radar em ambientes marítimos utilizando Redes Neurais artificiais como método de associação de dados. O desempenho desta metodologia é comparado com o da abordagem clássica baseada no algoritmo de associação de dados *Multi Hypothesys Tracking* (MHT). Com propósito de realizar esta comparação e validar a metodologia proposta foi empregado um gerador artificial de dados e criada uma ferramenta de simulação que permite avaliar diferentes métricas de desempenho.

De modo geral, a metodologia proposta permitiu a redução do tempo de execução do algoritmo comparando com a abordagem MHT, adicionalmente, quando em alguns cenários são introduzidas perdas de detecção, a metodologia clássica apresenta dificuldades em manter o acompanhamento de alguns alvos, o que pode ser considerado um dos seus pontos fracos. Em relação ao número de falsos acompanhamentos criados, a metodologia clássica apresenta um número maior em relação à abordagem proposta, provavelmente devido à sua complexidade na estratégia de busca (combinatória) e o processo de inicialização. Isto demonstra a viabilidade de utilizar a metodologia proposta nesta dissertação.

Abstract of Dissertation presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

MULTIPLE TARGET TRACKING STRATEGIES: A COMPARISON BETWEEN THE DATA ASSOCIATION APPROACHES MHT AND NEURAL NETWORKS

Manuel Ramón Vargas Avila

April/2015

Advisors: Luiz Pereira Calôba
Beatriz de Souza Leite Pires de Lima

Department: Electrical Engineering

This work presents a multi-target tracking methodology based on radar sensor information in marine environments using artificial neural networks as data association method. The performance of this methodology is compared with the classical approach based on the data association algorithm Multi Hypothesys tracking (MHT). With the purpose of performing this comparison and validating the proposed methodology was used an artificial data generator and created a simulation tool for assessing different performance metrics.

In general, the proposed methodology allowed a reduction of the execution time of the algorithm by comparing with the MHT approach, in addition, in some scenarios where leak detection are introduced, the classical methodology presents difficulties in maintaining tracking of some targets, which may be considered one of its weaknesses. Regarding the number of false tracks created, the classical methodology presents a greater number in relation to the proposed methodology, probably because of the complexity in the search strategy (combinatorial) and the initialization process. This demonstrates the feasibility of using the methodology proposed in this dissertation.

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

Os algoritmos de acompanhamento de alvos são cada vez mais utilizados em sistemas de navegação e vigilância civis e militares, permitindo a identificação e rastreamento de ameaças em tempo real. Entretanto, acompanhar um alvo que trafega num ambiente com alta densidade de *clutter* ou falsos alarmes, caracterizado pela incerteza em relação à origem das observações é um processo extremamente complexo que exige algoritmos de associação de dados sofisticados capazes de lidar com um grande volume de dados e, conseqüentemente, o elevado custo computacional inerente. Atualmente, diversas técnicas alternativas eficientes têm sido desenvolvidas para resolver este problema, especialmente em aplicações militares como o controle tático e de armamentos, onde é necessário detectar ameaças e executar uma ação no menor tempo e com o menor erro possível.

No contexto deste trabalho, entenda-se por contato, medida ou observação qualquer ponto fornecido pelo sensor cuja fonte pode ser um alvo ou *clutter*. Define-se por alvo qualquer objeto de interesse, enquanto que *clutter* é um contato gerado devido às condições do mar, objetos como boias ou erros internos do sensor entre outros fatores. *Tracks* ou acompanhamentos referem-se ao conjunto de parâmetros ou medidas que podem representar o movimento de um alvo. Falsos acompanhamentos referem-se a acompanhamentos compostos em sua maioria por medidas geradas por clutter. *Scan* é cada varredura da antena do radar. E *tracking* ou rastreamento é o processo de estimação dos parâmetros da cinemática do alvo de interesse.

O interesse principal deste trabalho é desenvolver um algoritmo que resolva o problema de acompanhamento de múltiplos alvos (*Multiple Target Tracking*, MTT), em uma plataforma estática monosensor a partir de informação extraída do sensor Radar em ambientes marítimos com alta densidade de *clutter* e comparar o seu desempenho com um algoritmo de abordagem clássica deste problema. Importa destacar que neste trabalho, são tratados exclusivamente alvos de superfície, por uma questão de limitação de escopo, embora as simulações aqui realizadas incluam

alvos com velocidades superiores àquelas que geralmente caracterizam este tipo de alvos.

O processo de associação de dados é a etapa primordial de um sistema de acompanhamento de múltiplos alvos. Dele depende a correta atualização do *track* e o número de falsos acompanhamentos criados. A presença de *clutter* dificulta a associação dos dados na medida em que se torna praticamente impossível decidir, com certeza, se um contato se refere ao alvo de interesse ou simplesmente é *clutter*. Associar estas observações espúrias (*clutter*) ao alvo de interesse pode resultar na perda do acompanhamento. Além disso, a alta densidade de *clutters* pode produzir maior quantidade de falsos acompanhamentos, comprometendo o desempenho computacional dos algoritmos. Outro problema que, provavelmente, pode acontecer nesta etapa é a perda de detecção, ou seja, em uma ou mais varreduras da antena do radar não se encontra uma medida para associar ao acompanhamento, o que também pode levar à perda do acompanhamento.

Um dos métodos básicos de associação é a chamada Associação pelo vizinho mais próximo, o qual consiste em associar ao alvo o contato mais próximo à previsão da posição futura do alvo, obtida geralmente através do algoritmo de filtro de Kalman. Esta abordagem apresenta baixa eficiência quando é utilizada em ambientes com alta densidade de *clutter* ou quando existem alvos pouco espaçados [1]. Alternativamente, várias metodologias sofisticadas têm sido empregadas para abordar o problema de associação: Associação probabilística não Bayesiana – cuja decisão é baseada em testes de hipóteses ou função de verossimilhança; Associação probabilística Bayesiana – cujas probabilidades (condicionais) posteriores são calculadas para várias associações possíveis de acordo com o teorema de Bayes, partindo de probabilidades priores, e posteriormente usadas durante o processo de estimação.

Segundo Bar-shalom e Li [4] o processo de associação é feito em três níveis nomeadamente associação "medida - a - medida" ou inicialização de acompanhamento (*Track formation*); associação "medida - ao - track" atualização ou manutenção do acompanhamento (*Track maintenance*); e associação "track - ao - track" ou fusão de acompanhamentos, utilizada em situações com múltiplos sensores.

O algoritmo clássico de associação mais empregado no MTT é o *Multi Hypothesis Tracking* (MHT) [5], o qual, como a própria designação indica, se baseia na criação de hipóteses para obter o caminho mais provável do alvo. Mas, num cenário com alta densidade de *clutters* pode ser criado um grande número de hipóteses, aumentando o custo computacional deste algoritmo, dificultando sua implementação em aplicações que precisam de resultados em tempo real. Portanto, neste trabalho é proposta uma abordagem baseada em redes neurais como alternativa ao algoritmo MHT visando contornar as dificuldades associadas ao problema de otimização

combinatória anteriormente mencionadas. As redes neurais são amplamente conhecidas pela recursividade dos seus modelos, seu baixo custo computacional após o seu treinamento, capacidade de se adaptar às mudanças no ambiente, robustez na presença de dados ruidosos e potencial para sua computação massiva paralela [6, 7]. Além disso, é um aproximador universal que pode mapear qualquer função linear ou não linear, segundo a estrutura selecionada. Por estas razões a rede neural é uma ótima ferramenta para problemas de classificação e aproximação de funções, recomendada para aplicações em tempo real.

Algumas aplicações que abordam o problema de associação utilizando redes neurais são apresentadas em [8, 9, 10], onde basicamente o processo de associação é abordado como um problema de otimização combinatória, criando uma função custo que é minimizada por um modelo neural Hopfield. Esta técnica apresentou resultados satisfatórios em diversas situações de acompanhamentos. Outra forma de abordar este problema usando redes neurais é apresentada em [11], onde é usada uma rede neural *feed-forward* de duas camadas para estimar a probabilidade a posteriori de uma medida pertencer a uma determinada classe. A saída desta rede obedece duas restrições: cada saída deve estar no intervalo (0, 1) e o somatório de todas as saídas deve ser igual a 1. A desvantagem desta abordagem é que precisa-se conhecer a priori o número de tracks para construir a rede, já que cada saída corresponde a um *track* existente. Enquanto que, a solução proposta nesta dissertação para o problema de associação consiste no uso de uma rede neural com treinamento "Levenberg-Marquart backpropagation", utilizando como entradas informações de cinemática contidas nos acompanhamentos, discriminando se uma medida pode ser associada ou não a um acompanhamento existente, sendo que este ultimo pode conter informações de um alvo de interesse ou de clutter correlacionado, entendesse por clutter correlacionado contatos que sem ser gerados por algum alvo formam uma trajetória coerente, ou seja a trajetória respeita limites de cinemática estabelecidos. A medida selecionada será aquela que apresente maior coerência com a cinemática do acompanhamento. Esta abordagem ao contrário do MHT não depende do pressuposto de gaussianidade sobre as medições.

Um outro destaque com relação à diferença entre o algoritmo de associação MHT e o algoritmo de rede neural proposto é que o primeiro requer o uso de um algoritmo complementar que tenha a função de estimar a posição futura do alvo existente a fim de delimitar o conjunto de novas medidas que podem ser associadas a este mesmo alvo. Entretanto, uma vez associada a nova medida ao alvo existente, e tendo em conta às incertezas em relação à origem desta medida, ambas as abordagens precisam de uma etapa de filtragem como forma de estimar a trajetória real do alvo. Uma ferramenta popularmente utilizada para lidar com as etapas de previsão e filtragem é o algoritmo de filtro de Kalman.

O filtro de Kalman parte dos pressupostos da linearidade das equações de evolução de estado do sistema e de medição, e da normalidade tanto dos ruídos associados a estas equações quanto do vetor inicial de estado. Sob estas condições, o filtro de Kalman permite determinar o estimador linear do estado não viciado e ótimo, uma vez que a variabilidade dos erros (ou erro quadrático médio) é minimizada [12]. Na prática, a maioria dos modelos de estado e medição são não lineares pelo que o uso do filtro de Kalman padrão se torna inadequado. Nestas situações é utilizado o filtro de Kalman estendido, conhecido como um algoritmo subótimo de estimação de estado para sistemas não lineares. Esta abordagem utiliza técnicas como a expansão de Taylor para obter uma aproximação linear do modelo.

Devido à possibilidade de um alvo experimentar diferentes padrões de comportamento ao longo do tempo, a implementação de um único filtro de Kalman se torna ineficiente para adaptar-se as variações do movimento. Uma abordagem alternativa eficaz é o estimador Múltiplos Modelos Integrantes (IMM) [4, 13, 14]. O IMM é um algoritmo proposto por Blom e Bar-shalom em 1988 [15], sua principal vantagem reside no fato de permitir a utilização de mais de um modelo de KF em paralelo para descrever o movimento do alvo.

1.1 Objetivos

O objetivo principal desta dissertação é desenvolver uma metodologia de acompanhamento de múltiplos alvos a partir de informações provenientes de um único sensor radar em ambientes marítimos, visando melhorar o desempenho obtido por uma metodologia clássica utilizada para resolver este problema. No nosso caso é selecionada a metodologia clássica baseada no algoritmo de associação de dados MHT.

Uma das motivações para o desenvolvimento desta nova metodologia é a redução do tempo de execução do algoritmo, já que por ser a metodologia clássica estudada baseada em testes de hipóteses (problema de otimização combinatória) o seu tempo de execução cresce consideravelmente ao aumentar o número de acompanhamentos ou o número de contatos a serem analisados (incluindo os contatos gerados por clutter), dificultando sua implementação em aplicações que requerem resultados em tempo real. Com o propósito de atingir este objetivo são apresentadas as redes neurais artificiais como uma alternativa à metodologia MHT.

Neste trabalho pretende-se também estudar diferentes algoritmos de treinamento, visando a encontrar aquele que apresente a melhor solução para este problema. Os algoritmos de treinamento que serão testados são o algoritmo *backpropagation* com gradiente descendente e o *backpropagation* com otimização Levenberg-Marquart.

Com o propósito de realizar a comparação dos diferentes algoritmos de treinamento e das duas metodologias de acompanhamento, foram criados cenários com alvos que descrevem diversas trajetórias para cobrir grande parte dos tipos de manobras que acontecem na realidade, incluindo alvos que navegam próximos um dos outros, caso frequente em áreas como portos. Também foram estabelecidas algumas métricas para avaliar os desempenhos dos algoritmos estudados.

Finalmente, pretende-se realizar uma analise do processo de filtragem, o qual fornece a estimação da trajetória real do alvo que esta sendo acompanhado.

1.2 Organização do trabalho

Este trabalho esta estruturado da seguinte forma, no capítulo 2 será apresentada uma descrição do método clássico, detalhando brevemente cada uma das etapas que o compõem. No capítulo 3 será apresentada a teoria referente às redes neurais, descrevendo seus aspectos estruturais, processo de treinamento e algumas considerações praticas. A teoria referente à etapa de filtragem é apresentada no capítulo 4, nele é exibida a dedução matemática de cada uma das equações do filtro de Kalman padrão e estendido, além de isso, é detalhado cada uma das etapas do algoritmo da filtragem IMM. No capítulo 5 será apresentada a metodologia proposta. Neste capítulo é exibida a base de dados utilizada, o modelo neural e de filtragem utilizado e as métricas estabelecidas para a comparação das metodologias. No capítulo 6 serão exibidos e analisados os resultados obtidos do treinamento da rede neural e da execução da metodologia clássica e proposta. Finalmente, as conclusões e trabalhos futuros serão apresentados no capítulo 7.

Capítulo 2

Sistema MTT convencional

Nesta seção apresentaremos uma abordagem convencional que trata o problema de acompanhamento de múltiplos alvos, a qual será comparada com a abordagem proposta neste trabalho.

O problema de acompanhamento de múltiplos alvos ou MTT, pode ser decomposto em três processos fundamentais:

- **Inicialização**: devido às incertezas na medição, nesta etapa, faz-se o processo de associação "medida-a-media", criando os possíveis *tracks* candidatos para sua posterior confirmação ou eliminação.

- **Manutenção**: processo responsável em associar as medidas recebidas aos *tracks* confirmados.

- **Filtragem**: processo usado para executar duas funções. A primeira é de predizer a posição futura do alvo de interesse (filtragem de previsão), útil nos processos de associação, e a segunda consiste em aproximar a trajetória real dos alvos de interesse a partir das medições selecionadas nos passos anteriores (filtragem de alisamento).

Na figura 2.1 é mostrado o esquema básico do processo de MTT. É importante lembrar que estamos tratando de medidas fornecidas em coordenas polares por um sistema Radar de superfície e que são obtidas a partir de ambientes marítimos com alto nível de clutter.

Antes de descrever o algoritmo MTT é necessário definir alguns pressupostos práticos:

- Podem existir zero ou mais alvos de interesse na área de vigilância. A posição e o número destes alvos é desconhecida a priori.

- Cada alvo produz aleatoriamente uma ou nenhuma medida por scan. Isto é governado pela probabilidade de detecção (P_D) a qual pode ser diferente para cada alvo.

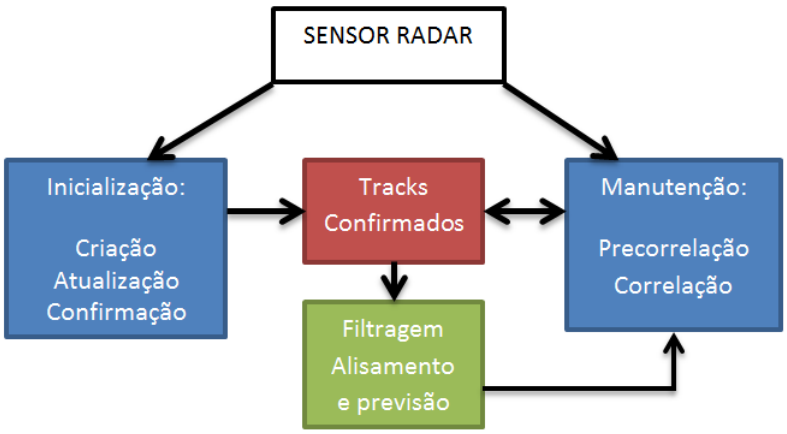


Figura 2.1: Esquema do processo de MTT

- O número e localização das medidas *clutter* é aleatório e segue uma função de distribuição de densidade Gaussiana.
- Cada medida é originada de uma única fonte. Assim, uma medida só pode ser utilizada para atualizar um único *track*, ou iniciar um novo *track*, ou então será classificada como falso alarme (*clutter*).

2.1 Inicialização de *tracks*

O objetivo do processo de inicialização é criar *tracks* no menor tempo possível, com um número mínimo de falsos *tracks*, estes falsos *tracks* são gerados pela presença de clutter no ambiente. Diversos enfoques têm sido desenvolvidos para abordar este problema, mas não existe uma solução geral, uma vez que o desempenho deste algoritmo varia conforme a situação (ambiente) na qual é aplicado. A escolha inapropriada do processo de inicialização pode resultar na saturação dos recursos computacionais, devido à criação de alto número de falsos *tracks*, ou num baixo desempenho, pelo fato de os verdadeiros *traks* não poderem ser inicializados ou simplesmente pode não se obter uma inicialização oportuna.

Uma abordagem comumente utilizada nesta etapa é a "two-stage cascaded logic" [4], na qual são utilizadas janelas de correlação para determinar as possíveis associações de medidas. Se existe uma determinada seqüencia de detecções nestas janelas que satisfazem um critério estabelecido, o *track* é aceito como válido. Esta é a abordagem considerada no desenvolvimento deste algoritmo. As etapas que compõem esta abordagem serão detalhadas a seguir, tomando como referencia o processo descrito em [4].

2.1.1 Criação do track preliminar

Qualquer medida não associada a um *track* existente é considerada como um "iniciador", usado para produzir uma "tentativa de *track*" (Tentative *track*).

Na varredura, após a detecção de um iniciador, uma janela de associação circular delimitada pelo alcance máximo e mínimo possíveis para o tipo de alvo que se quer identificar (aéreo ou de superfície) e pela intensidade de ruído da medida é construída entorno dele, i.e,

$$D.min = (V_{min} * t_r) - (3 * \sigma_d)$$
$$D.max = (V_{max} * t_r) + (3 * \sigma_d),$$

onde $D.min$ e $D.max$ são o alcance mínimo e máximo respectivamente, V_{min} e V_{max} representam as velocidades mínima e máxima definidas para o tipo de alvo de interesse, t_r é o tempo de rotação de antena do Radar e σ_d é o desvio padrão do ruído de medição na distância.

Se houver um alvo que deu origem ao iniciador, na varredura seguinte, uma medida (se detectada) aparecerá dentro da janela criada com probabilidade próxima de unidade, tornando a tentativa de *track* em "*track* preliminar". Caso não haja alguma medida que possa ser associada, a tentativa de *track* é descartada. Medidas fora da janela não são consideradas no processo de associação, mas resultam na criação de novos iniciadores. É importante destacar que existe a possibilidade que o *track* preliminar seja composto unicamente de clutter correlacionado. Neste caso o *track* é considerado como falso *track* preliminar que poderá ser eliminado ou não nos passo seguintes.

A definição do tamanho da janela é uma questão extremamente crítica, principalmente em ambientes com alto nível de ruído, devido à possibilidade de se criar um alto número de falsas hipóteses de *tracks* preliminares, pelo que é recomendável se tenha uma idéia da cinemática do alvo a ser acompanhado. É possível estabelecer algumas regras que permitam limitar o número de hipóteses como por exemplo, criar no máximo três hipóteses por cada iniciador.

Um exemplo típico deste procedimento é apresentado na figura 2.2, onde o ponto vermelho representa o iniciador, a área de cor verde representa a janela de associação, os pontos azuis representam às medidas obtidas na varredura seguinte e as linhas pontilhadas fazem referência as diferentes hipóteses criadas.

No caso do MTT aqui implementado foi definido $V_{min} = 50\ m/seg$, $V_{max} = 500\ m/seg$, $t_r = 2seg$ e $\sigma_d = 30\ m$.

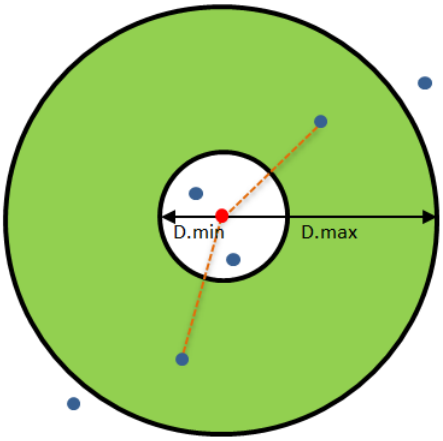


Figura 2.2: Janela de precorrelação

2.1.2 Atualização dos *tracks* preliminares

A atualização dos *tracks* é o processo chave para identificar as observações que podem ser associadas ao *track* preliminar. Esta etapa consta de 3 partes:

a) **Estimação da posição futura**:
A partir das duas medidas do *track* preliminar, que representam um possível alvo, um filtro de Kalman pode ser inicializado e utilizado para prever a posição deste alvo na varredura seguinte. Geralmente, nesta etapa é implementado um filtro de Kalman padrão e assumi-se que a cinemática do alvo é caracterizada pelo modelo de movimento retilíneo uniforme (MRU).

b) **Etapa de precorrelação**:
Com centro na posição prevista pelo filtro é criada uma janela de precorrelação baseada nas condições de cinemática do alvo contido no *track* preliminar, na magnitude do erro na previsão e na intensidade de ruído da medida, delimitando assim o número de possíveis associações do *track* preliminar com as medidas. A matriz de covariâncias dos erros de previsão obtida no filtro de Kalman fornece as incertezas, na estimação do estado, que são necessárias na criação desta janela [1].

c) **Associação das medidas (contatos)**:
A seleção da medida que será associada ao *track* preliminar é feita por um processo simples conhecido como associação pelo vizinho mais próximo, o qual consiste em associar a medida que tenha a menor distância com respeito à previsão feita pelo filtro de Kalman. O critério de distância considerado neste caso é a distancia euclidiana definida como

$$D = \sqrt{[x(k+1) - \hat{x}(k+1|k)]^2 + ([y(k+1) - \hat{y}(k+1|k)]^2)}, \qquad (2.1)$$

onde (x, y) representam a posição observada pelo radar, $(\hat{x}, \hat{y})$ é a posição prevista pelo filtro de Kalman.

2.1.3 Confirmação ou eliminação do *track*

A partir da terceira varredura, a lógica de M detecções em N scans é utilizada para decidir se o *track* preliminar é confirmado ou eliminado. Esta lógica é nomeada critério M/N. Se no final das $N + 2$ varreduras o critério lógico for satisfeito, o *track* preliminar é agora chamado por "*track* confirmado", de outra maneira o *track* é eliminado.

Esta lógica é executada de forma iterativa da seguinte maneira [4, 16]:

1) Cálculo do número de detecções até a varredura k

$$\lambda_D = \sum_{j=3}^{k} \delta_j \tag{2.2}$$

onde j é a j-ésima varredura, k é a varredura atual, δ é o vetor que contem a seqüencia de detecções de um *track* preliminar.

2) Cálculo do número de detecções perdidas até a varredura k

$$\lambda_P = \sum_{j=3}^{k} (1 - \delta_j) \tag{2.3}$$

3) Critério de confirmação ou eliminação.

Se $\lambda_D = M$, o *track* é confirmado.
Se $\lambda_P = N - M + 1$, o *track* é eliminado.
Uma escolha razoável dos parâmetros M e N, segundo Gini e Rangaswamy [17], pode ser encontrada respeitando a restrição

$$A_{vr} P_{fa} \leq \frac{M}{N} \leq P_d \tag{2.4}$$

Onde A_{vr} é a área média da região de validação, P_{fa} e P_d são respectivamente as probabilidades de falso alarme e de detecção do alvo. Neste algoritmo, os valores dos parâmetros M e N serão definidos de forma determinística. Assim, segundo a densidade de clutter na área de vigilância, estes parâmetros podem assumir valores inteiros entre 3 e 9. Na tabela abaixo é apresentado um exemplo da evolução do processo de confirmação ou eliminação de *tracks* para valores de $M = 2$ e $N = 3$.

Tabela 2.1: Exemplo da evolução do processo de eliminação e confirmação para $M = 2$ e $N = 3$ (adaptada de [4])

k	δ	λ_D	λ_P	Classificação
0	–	–	–	–
1	1	–	–	alvo tentativa
2	10	–	–	Eliminado
	11	–	–	Preliminar
3	110	0	1	Preliminar
	111	1	0	Preliminar
4	1100	0	2	Eliminado
	1101	1	1	Preliminar
	1110	1	1	Preliminar
	1111	2	0	Confirmado
5	11010	1	2	Eliminado
	11011	2	1	Confirmado
	11100	1	2	Eliminado
	11101	2	1	Confirmado

2.2 Manutenção

Uma vez confirmados os tracks preliminares a etapa seguinte é a atualização do acompanhamento a cada novo scan. Este procedimento é importante pelo fato de que uma decisão equivocada de associação, pode resultar na atualização do track com *clutter*, criando um falso acompanhamento ou, no pior dos casos, produzindo a eliminação do *track* verdadeiro. Geralmente, nesta etapa é empregada uma integração entre a filtragem e previsão feitas pelo algoritmo IMM com o processo de associação de dados feito pelo algoritmo MHT, esta integração é conhecida como IMM/MHT. Para melhor compreensão, esta etapa será dividida em três grandes partes descritas a seguir:

a) **Alisamento**, **estimação da posição futura e criação da janela de precorrelação**:
Com a informação contida no track confirmado é possível ter uma idéia mais completa da cinemática do alvo que esta sendo acompanhado (é possível conhecer a posição, velocidade e aceleração), então, uma filtragem mais sofisticada como o algoritmo IMM pode ser utilizada para fazer o alisamento do acompanhamento e para prever a posição do alvo no instante seguinte. Entorno desta posição é criada a janela de precorrelação de igual forma como descrito na seção 2.1.2. Estas janelas são necessárias para a eliminação de associações de medidas ao *track* pouco prováveis, diminuindo assim o número de hipóteses criadas no processo de associação.

b) **Algoritmo de associação**:

Segundo Blackman [1], o Multiple Hypothesis tracking (MHT) é geralmente uma abordagem aceita e amplamente utilizada para solucionar o problema de associação no MTT. O princípio chave desta abordagem é que as decisões difíceis de associação de dados, como no caso mostrado na figura 2.3 onde diferentes *tracks* apresentam medidas em comum nas janelas de precorrelação, são adiadas até que mais dados sejam recebidos. Este princípio faz com que a hipótese seja selecionada com menor nível de incerteza possível.

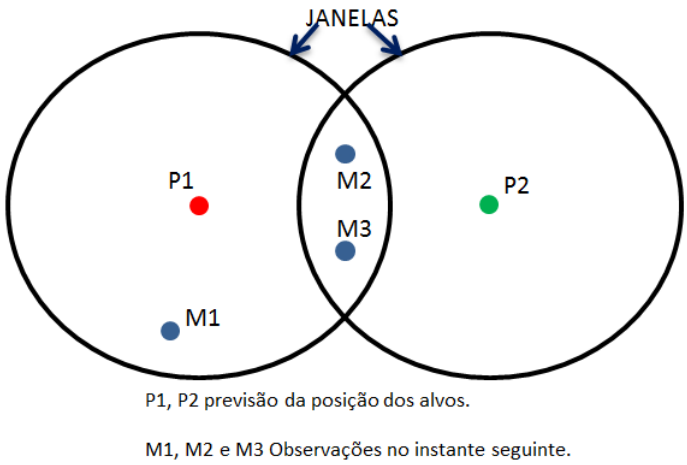


Figura 2.3: Exemplo de conflito de associação típico (adaptada de [1])

Devido à formulação matemática complexa desta abordagem e por não ser o enfoque desta dissertação, apresentaremos resumidamente o seu funcionamento evitando detalhar as expressões matemáticas associadas.

Reid [18] foi o primeiro a desenvolver um enfoque do algoritmo completo da abordagem MHT, conhecido como MHT orientado a hipóteses (HOMHT). Esta abordagem funciona da seguinte forma: ao receber novos dados, cada uma das hipóteses existentes é expandida num conjunto de novas hipóteses, considerando todas as associações "medida-ao-*track*"possíveis para os *tracks* contidos nas hipóteses originais. Só são aceitas hipóteses que respeitem a restrição de compatibilidade. *Tracks* são ditos compatíveis se não apresentam observações em comum.

Uma vez criados os *tracks*, é feito um processo de avaliação probabilística que inclui aspectos como a probabilidade a priori da presença do alvo, a densidade de falsos alarmes, a seqüencia de detecção, entre outros. Com isto, utiliza-se o logaritmo da razão da verossimilhança como critério de avaliação de cada um dos *tracks*, cujo procedimento matemático é detalhado em [19]. A avaliação (pontuação) da hipótese é feita somando as avaliações atribuídas a cada *track* pertencente a esta hipótese. Dada esta avaliação a probabilidade da hipótese

pode ser calculada. Finalmente, o valor de probabilidade correspondente a um *track* é obtido somando-se as probabilidades de todas as hipóteses que possuem este *track*. Uma visão geral da lógica do HOMHT é apresentada na figura 2.4

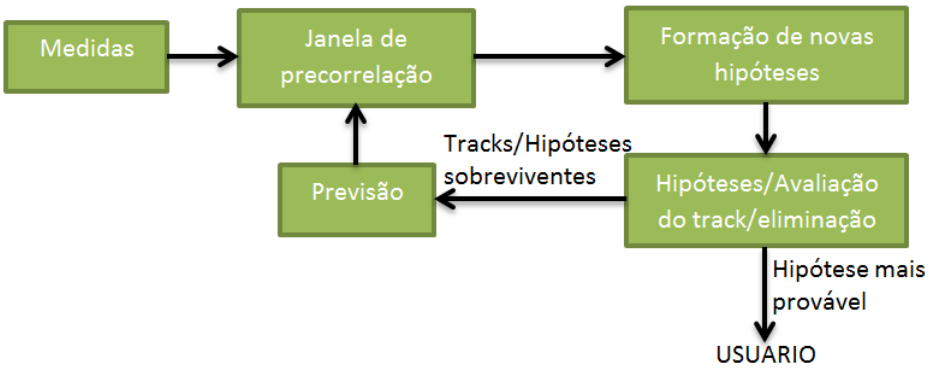


Figura 2.4: Esquema da lógica HOMHT (adaptado de [2])

O processo de formação de hipóteses é considerado um método de enumeração exaustivo uma vez que realiza todas as possíveis combinações medida-ao-*track*. Portanto, dependendo da densidade de *clutter* no ambiente estudado, o número de hipóteses definidas pode crescer exponencialmente, aumentando o custo computacional e, conseqüentemente, dificultando a implementação em tempo real. Por exemplo, na situação da figura 2.3 extraída de [1], mostra-se que com apenas 2 *tracks* e 3 medidas candidatas a ser associadas, foram obtidas 10 possíveis hipóteses. Devido a este problema de crescimento do número das hipóteses associado à proposta de Reid [18], algumas abordagens alternativas têm sido desenvolvidas. Para cenários com alta densidade de *clutter*, uma abordagem amplamente utilizada é o algoritmo MHT orientado ao track (Track-oriented MHT, TOMHT), a qual viabiliza a operação em tempo real [20].

O TOMHT foi introduzido por Bar-Shalom em 1990 [21] e, ao contrário da abordagem orientada a hipótese, não mantém o conjunto de hipóteses entre varreduras, mas sim o conjunto de *tracks* que podem ser incompatíveis. Define-se por *tracks* incompatíveis aqueles que tem medidas em comum. Além disso, um argumento forte para o uso desta abordagem é que, tipicamente, o número de *tracks* criados é menor do que o número de hipóteses formadas pelo HOMHT.

O esquema da lógica do TOMHT é apresentado na figura 2.5, cujas etapas são descritas conforme se segue [2]:

A etapa inicial do algoritmo consiste na criação da janela de precorrelação, cujo objetivo, como referido anteriormente é selecionar as medidas que podem ser utilizadas para atualizar um *track* existente ou em certos casos, inicializar um novo *track*. A etapa seguinte consiste na manutenção dos *tracks*, onde o conjunto de *tracks* existente é expandido utilizando todas as possíveis associações dos *tracks* com um novo conjunto de medidas. Paralelamente a esta etapa é calculada também a probabilidade de cada *track* conforme apresentado em [19]. Na etapa de redução, *tracks* com probabilidades menores do que um *threshold* definido poderão ser eliminados, ademais, os *tracks* similares podem ser fundidos de acordo com algum critério de similaridade. Depois da redução, pode ser feita uma poda do conjunto de tracks.Para isso devemos voltar N varreduras no passado e decidir quais associações *tracks*-medidas são corretas. Finalmente é formada a hipótese mais provável que contém um conjunto de *tracks* compatíveis. Deve-se lembrar que no processo de manutenção podem ser gerados *tracks* incompatíveis, pelo que os *tracks* são organizadas em ordem decrescente de probabilidade. O *track* mais provável é adicionado à hipótese, o seguinte *track* mais provável também pode ser adicionado se for compatível com os demais *tracks* contidos na hipótese. Este processo é feito até que não existam mais *tracks* compatíveis.

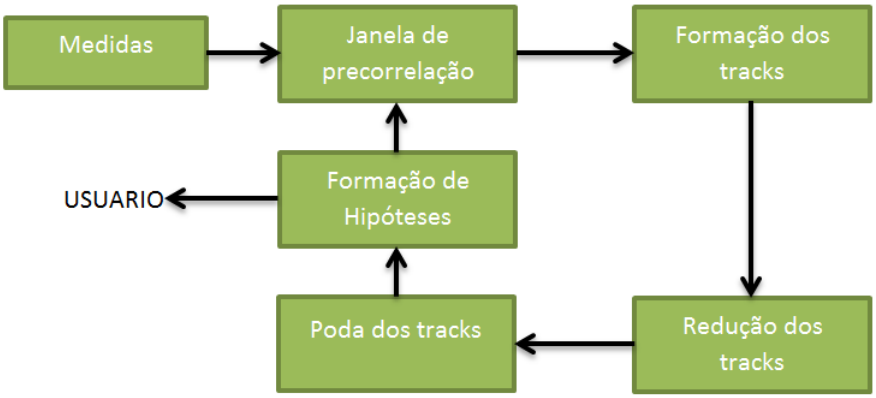


Figura 2.5: Esquema da lógica TOMHT, (adaptado de [2]

Considerando as vantagens da abordagem TOMHT, mencionadas anteriormente, a mesma foi selecionada para tratar o problema de associação de dados no algoritmo que serve de base de comparação para o algoritmo de associação proposto nesta dissertação.

c) **Critério de Terminação**:
De forma análoga ao critério M/N podem-se criar heurísticas para considerar a terminação de um tracks. Comumente N perdas consecutivas de

detecção levam à terminação do acompanhamento [1]. Neste algoritmo, serão consideradas um máximo de 3 perdas consecutivas para terminar o acompanhamento.

Capítulo 3

Redes Neurais

As Redes Neurais Artificiais ou RNAs foram inspiradas originalmente na capacidade e habilidade do cérebro humano para executar em paralelo tarefas não lineares complexas de forma rápida e eficiente, as quais num computador convencional poderiam levar dias para serem executadas. Isto se deve à capacidade do cérebro de organizar seus componentes estruturais, chamados de neurônios, para realizar tarefas determinadas, como o reconhecimento de padrões, percepção, entre outras.

O neurônio é a unidade fundamental do sistema nervoso e particularmente do cérebro. Cada neurônio é uma unidade processadora que recebe e mistura sinais que são transmitidos para outros neurônios interconectados. As conexões entre neurônios são chamadas de sinapses, as quais se fortalecem ou enfraquecem durante o processo de aprendizagem. Se a mistura de entradas é suficientemente forte a saída do neurônio é ativada. É com base neste modelo biológico que as redes neurais artificiais são desenvolvidas.

Uma RNA pode ser definida como um processador paralelo, composto por unidades de processamento simples, cujo funcionamento é determinado pelos pesos das sinapses que conectam os neurônios e pelas funções de ativação selecionadas para cada um deles. Algumas propriedades das RNAS destacam-se:

- São consideradas métodos auto-adaptativos, que adquirem conhecimento mediante a experiência sem que existam suposições a priori sobre os modelos estudados. Em outras palavras, seu comportamento se ajusta em função do contexto.

- A sua estrutura paralela tornam as redes capazes de resolver problemas complexos de grande escala, que seriam difíceis de tratar com métodos tradicionais. Além disso, a estrutura pode conter tanto neurônios lineares como não lineares, e apresentar múltiplas camadas, isto é especialmente útil em problemas de classificação complexos que lidam com classes linearmente não separáveis.

- Tem a capacidade de generalizar, devido ao fato de a rede neural produzir saídas adequadas para entradas que não tenham sido treinas, tendo habilidade para lidar com dados ruidosos, incompletos ou imprecisos.

- A sua execução pode ser feita em duas fases independentes: treinamento e teste. Na fase de treinamento são ajustados os pesos das sinapse, com o objetivo de minimizar o erro de saída da rede. Em seguida, na fase de teste são apresentadas novas entradas à rede, calculando sua saída sem alterar os pesos. É importante destacar que a fase de treinamento representa a maior parte do custo computacional total da rede.

- Segundo [22, 23] qualquer função contínua pode ser aproximada por uma rede neural com uma única camada intermediária contendo um número finito de neurônios, o que faz das RNAs ótimas ferramentas para o mapeamento de funções de separação e de funções de estimação.

As propriedades anunciadas acima tornam as RNAs viáveis para resolver o problema de associação em tempo real, que é objeto de estudo desta dissertação.

3.1 Aspectos estruturais

O primeiro modelo matemático de um neurônio foi desenvolvido em 1943 por McCulloch e Pitts. Os autores assumem a saída do modelo de neurônio binária e mostram que, uma rede com um número suficiente de neurônios, conectados por sinapses ajustadas apropriadamente e operando de forma síncrona, em princípio, realizaria a computação de qualquer função [24]. Este modelo é a base para a maioria das arquiteturas neurais que são utilizados desde então.

Em 1958, Rosenblatt desenvolveu o modelo perceptron [25], hoje considerado o modelo básico de neurônio artificial. Nesse trabalho os neurônios eram organizados em camadas, uma de entrada e outra de saída, e os pesos das sinapses eram ajustáveis. Mas, Minsky e Papert em 1969 [26], mostraram as limitações deste modelo, afirmando que uma RNA com esta estrutura não conseguiria separar classes que não fossem linearmente separáveis. Para contornar este problema, foram propostas as redes Perceptron multi-camadas (MLP), que diferem do modelo de Rosenblatt pelo fato de possuírem uma ou mais camadas intermediárias de processamento. O número de camadas dependerá do problema que esteja sendo estudado, porém, segundo [22, 23], apenas uma camada intermediária com um número finito de neurônios é suficiente para aproximar qualquer função contínua. Atualmente este é o modelo de RNAs mais utilizado. Uma representação da rede MLP totalmente conectada é apresentada na figura 3.1, onde N é o número de

entradas, n_1 e n_2 são respectivamente o número de neurônios da primeira e segunda camada e R é o número de saídas.

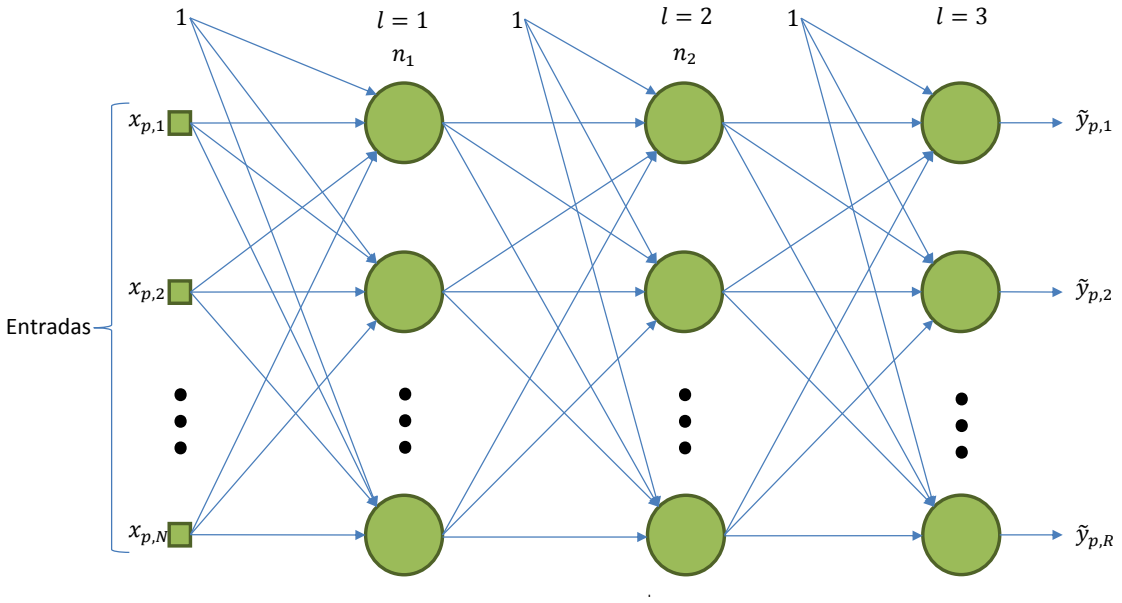


Figura 3.1: Estrutura RNA MLP

De acordo com o sentindo de propagação dos sinais dentro da sua estrutura, a rede MLP pode ser classificada em rede alimentada para frente (*feedforward*) ou rede recorrente. Nas redes *feedforward*, as conexões entre os neurônios não formam ciclos, i.e, o sentido de propagação é único. No caso das redes recorrentes, existe pelo menos um laço de realimentação (a saída da rede é realimentada na sua entrada), que geralmente, serve para introduzir uma componente de memória na rede neural, fornecendo-lhe a habilidade de tratar problemas variantes no tempo.

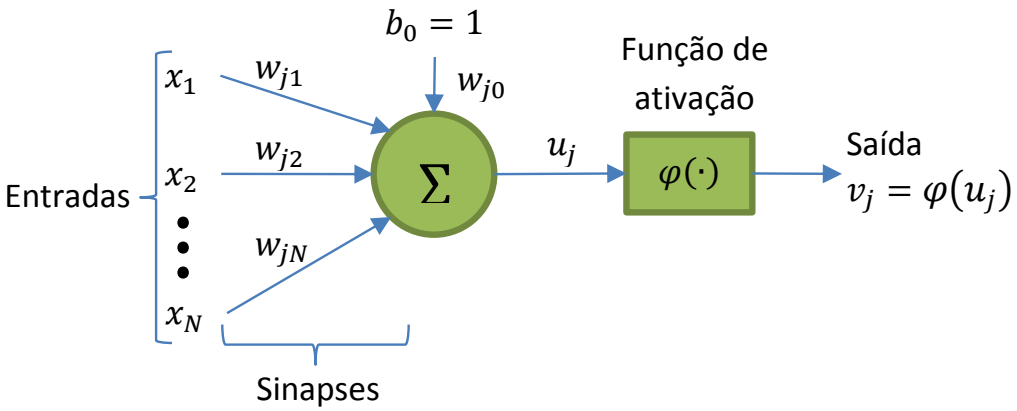


Figura 3.2: Modelo de neurônio (adaptada de [3])

A figura 3.2 representa o modelo de um neurônio utilizado na rede MLP acima descrita. Basicamente, este modelo opera em três etapas. Na primeira, que chamaremos de ponderação, o conjunto de sinais de entradas é transmitido ao neurônio por um conjunto de sinapses, as quais possuem um peso próprio, w_{ji}, que multiplica cada uma das entradas, ponderando assim cada uma delas de acordo com o grau de conexão da sinapse. Este peso é ajustável e seu valor pode ser positivo ou negativo. Quanto à notação utilizada, o índice j referencia o número do neurônio em questão e i o ponto de origem da sinapse. Na etapa seguinte é feita uma combinação linear das entradas ponderadas, para assim, determinar um único sinal denominado potencial de ativação do neurônio, u_j. Assim, matematicamente teremos que

$$u_j = \sum_{i=1}^{N} x_i * w_{ji} + b_0 * w_{j0}, \tag{3.1}$$

onde N é o número de entradas do neurônio j e o termo "$b_0 * w_{j0}$"é chamado de limiar de ativação ou viés (*bias*) do neurônio. Finalmente, este potencial de ativação é processado por uma função de ativação, $\varphi(u_j)$, para produzir o sinal de saída do neurônio, v_j. Esta função também é conhecida como função restritiva, já que restringe o intervalo permissível de amplitude do sinal de saída a um valor finito. Esta função pode ser linear, degrau ou não linear, sendo que esta última representa, com maior precisão, o comportamento dos neurônios biológicos. Algumas funções de ativação tipicamente usadas são expressas na tabela abaixo [3].

Tabela 3.1: funções de ativação

Função linear	$y_j = u_j$
Função Degrau	$y_j = \begin{cases} 1 & \text{se} \quad u_j \geq 0 \\ -1 & \text{se} \quad u_j < 0 \end{cases}$
Função logística	$y_j = \frac{1}{1+e^{-a*u_j}}$
Função Tangente Hiperbólica	$y_j = \tanh u_j$

A escolha do tipo de função depende do problema a ser tratado. Assim, no caso de problemas de classificação, onde as saídas são discretas, usualmente são utilizados neurônios tipo tangente hiperbólica, tanto na camada intermediária como na de saída. Isto faz com que a saída da rede esteja no intervalo $[1, -1]$, onde valores acima de zero indica que a entrada pertence a uma determinada classe.

3.2 Processo de treinamento

Entende-se por treinamento, no contexto de redes neurais, o processo de ajuste dos pesos das sinapses de cada um dos neurônios de forma que a rede se adapte aos diferentes estímulos de entrada. Segundo a forma como o processo de treinamento é implementado, podem-se distinguir dois tipos [6, 27]:

- **Supervisionado**: também chamado treinamento assistido, é apresentado à rede um vetor composto por cada par de entrada e saída desejada, de modo que a rede possa calcular o seu erro de treinamento que é utilizado para ajustar cada um dos pesos das sinapses. O algoritmo *Backpropagation* pertence a este tipo de treinamento.

- **Não supervisionado**: somente é fornecido o vetor de entradas à rede. Neste caso, os pesos são ajustados de forma que padrões de entrada similares produzam o mesmo padrão de saída. Pode-se citar como exemplo o treinamento competitivo, ver [3] para detalhes deste treinamento.

3.2.1 Algoritmo *Backpropagation* e Levenberg-Marquart

Em 1986, Rumelhart et al. [28], desenvolveram um algoritmo para que uma rede MLP aprendesse a relação existente entre um conjunto de padrões de entrada e suas saídas correspondentes. Isto é feito recalculando os pesos de cada uma das sinapses de forma a minimizar uma função de erro. Este algoritmo ficou conhecido como *Backpropagation*. A inovação deste algoritmo foi à atribuição do erro obtido na saída da rede às camadas ocultas. Uma forma simples de ajustar os pesos é utilizar o método de gradiente descendente [29]. Assim, os pesos são ajustados como

$$w_{ji}(k+1) = w_{ji}(k) + \Delta w_{ji} \tag{3.2}$$

$$\Delta w_{ji} = -\alpha * \frac{\partial f_0(w)}{\partial w_{ji}}, \tag{3.3}$$

onde Δw_{ji} são conhecidos como ajustes dos pesos, α é a taxa de aprendizagem e $f_0(w)$ é a função de erro a ser minimizada. Existem diferentes funções de erro que podem ser utilizadas para este fim, dentre elas destaca-se o erro quadrático médio, utilizado no algoritmo *backpropagation* padrão [28]. Este algoritmo é um dos mais utilizados na literatura por sua simplicidade de implementação. No entanto, devido a sua convergência lenta, dependência dos parâmetros iniciais e a possibilidade de cair em mínimos locais, variações deste algoritmo têm sido desenvolvidas. Segundo Hagan e Menhaj [29], estas modificações podem ser diferenciadas em duas categorias: a primeira envolve técnicas *ad hoc* e a segunda envolve métodos numéricos de

otimização. Um dos métodos mais populares desta última categoria é o algoritmo Levenberg-Marquart [30], que será descrito nesta seção.

O algoritmo Levenberg-Marquart é uma interpolação entre o método do gradiente descendente e o algoritmo de Gauss-Newton, herdando assim a velocidade de convergência do algoritmo de Gauss-Newton e a estabilidade do método do gradiente.

Assim, assumindo uma função $f_0(w)$ a ser minimizada com respeito ao vetor de pesos w, aplicando-se o método de Newton teremos que

$$\Delta w = -[\nabla^2 f_0(w)]^{-1} \nabla f_0(w), \tag{3.4}$$

onde $\nabla^2 f_0(w)$ é a matriz hessiana, $\nabla f_0(w)$ é o gradiente e $f_0(w)$ é definida como a soma do erro quadrático dado por

$$f_0(w) = \frac{1}{2} \sum_{p=1}^{P} \sum_{r=1}^{R} (\epsilon_{p,r}(w))^2, \tag{3.5}$$

onde P é o total de pares de treinamento, R o número de saídas da rede e $\epsilon_(p, r)$ é o erro de treinamento da saída r ao aplicar o par p [29, 31, 32]. A partir desta equação podemos encontrar que

$$\begin{aligned} \nabla f_0(w) &= \sum_{p=1}^{P} \sum_{r=1}^{R} \epsilon_{p,r}(w) \frac{\partial \epsilon_{p,r}(w)}{\partial w} \\ &= J^T(w)\varepsilon(w), \qquad (3.6) \\ \nabla^2 f_0(w) &= J^T(w)J(w) + S(w), \qquad (3.7) \end{aligned}$$

onde ε é o vetor de erro de forma

$$\varepsilon = \begin{bmatrix} \epsilon_{1,1} \\ \vdots \\ \epsilon_{1,R} \\ \vdots \\ \epsilon_{P,1} \\ \vdots \\ \epsilon_{P,R} \end{bmatrix}, \tag{3.8}$$

e $J(w)$ é a matriz jacobiana com componentes $\frac{\partial \epsilon_{p,r}(w)}{\partial w_{j,i}}$. Os pares de índices (j, i) podem ser mapeados num único índice k, tomando valores entre $1, ..., K$, sendo K

o número total de pesos na rede. Desta forma teremos

$$J(w) = \begin{bmatrix} \frac{\partial \epsilon_{1,1}(w)}{\partial w_1} & \frac{\partial \epsilon_{1,1}(w)}{\partial w_2} & \cdots & \frac{\partial \epsilon_{1,1}(w)}{\partial w_K} \\ \frac{\partial \epsilon_{1,2}(w)}{\partial w_1} & \frac{\partial \epsilon_{1,2}(w)}{\partial w_2} & \cdots & \frac{\partial \epsilon_{1,2}(w)}{\partial w_K} \\ \vdots & \vdots & \cdots & \vdots \\ \frac{\partial \epsilon_{1,R}(w)}{\partial w_1} & \frac{\partial \epsilon_{1,R}(w)}{\partial w_2} & \cdots & \frac{\partial \epsilon_{1,R}(w)}{\partial w_K} \\ \vdots & \vdots & \cdots & \vdots \\ \frac{\partial \epsilon_{P,1}(w)}{\partial w_1} & \frac{\partial \epsilon_{P,1}(w)}{\partial w_2} & \cdots & \frac{\partial \epsilon_{P,1}(w)}{\partial w_K} \\ \vdots & \vdots & \cdots & \vdots \\ \frac{\partial \epsilon_{P,R}(w)}{\partial w_1} & \frac{\partial \epsilon_{P,R}(w)}{\partial w_2} & \cdots & \frac{\partial \epsilon_{P,R}(w)}{\partial w_K} \end{bmatrix}. \tag{3.9}$$

Substituindo 3.6 e 3.7 em 3.4 e adicionando a modificação de Levenberg-Marquardt [30, 31, 32], teremos a regra de aprendizado:

$$\Delta w = -[J^T(w)J(w) + \mu I]^{-1} J^T(w)\varepsilon(w), \tag{3.10}$$

onde I é a matriz identidade e μ é um valor sempre positivo, designado por coeficiente de combinação. Observe-se que quando μ é muito pequeno ou próximo de zero, o algoritmo de gauss-newton é utilizado, e caso μ seja grande utilízasse o método do gradiente descendente.

Pode ser inferido, a partir de 3.10, que o passo chave do algoritmo Levenberg-Marquardt é o cálculo da matriz jacobiana, este pode ser feito realizando algumas modificações no algoritmo *backpropagation* tradicional. A principal diferença entre estes dois algoritmos é que o processo de retropropagação no algoritmo Levenberg-Marquardt deve ser repetido para cada saída separadamente, com o propósito de obter cada um dos elementos da matriz jacobina.

O processo de execução deste algoritmo para o treinamento de RNAs, consiste basicamente em duas etapas [31, 33, 34]: na primeira, um padrão é apresentado à camada de entrada da rede, propagando-o para frente, camada por camada, até se obter a saída e seu erro. Na segunda etapa, os erros são retropropagados desde a camada de saída até a camada de entrada com o objetivo de recalcular os pesos de maneira a minimizar o critério de erro.

Assim, considerando uma rede MLP de L camadas ($l = 1, ..., L$, sendo L a camada de saída), com N entradas, R saídas e n_l neurônios por camada, treinada com P pares de entrada-saída definidos por $(\overrightarrow{X}_1, \overrightarrow{Y}_1), (\overrightarrow{X}_2, \overrightarrow{Y}_2), ..., (\overrightarrow{X}_P, \overrightarrow{Y}_P)$, onde $\overrightarrow{X}_p = x_{p,1}, x_{p,2}, ..., x_{pN}$ e $\overrightarrow{Y}_p = y_{p,1}, y_{p,2}, ..., y_{p,R}$, conectada como mostrado na figura 3.1, e seguindo o modelo de neurônio da figura 3.2, o algoritmo levenberg-Marquardt é executado da seguinte forma [31]:

Propagação para frente:

- Cálculo da saída do neurônio j da camada l produzida pelo par de treinamento p:

$$v^l{}_{p,j} = f_j^l(u^l{}_{p,j}) \tag{3.11}$$

$$u^l{}_{p,j} = \sum_{i=1}^{n_{l-1}} w_{j,i}^l * v^{l-1}{}_{p,i} + w_{j,0}^l, \tag{3.12}$$

onde n_{l-1} é o número de neurônios da camada anterior, $v^{l-1}{}_{p,i}$ é a saída do neurônio i da camada anterior ao ser aplicado o par p, f_j^l e $w_{j,0}^l$ são respectivamente a função de ativação e o peso do viés do neurônio j da camada l. Se o neurônio j pertence à primeira camada intermediária, ou seja, $l = 1$, então $v^0{}_{p,i} = x_{p,i}$, onde $x_{p,i}$ é o $i - ésimo$ elemento do vetor de entrada $\overrightarrow{X}_p$ e $n_0 = N$. No caso de o neurônio j pertencer à camada de saída, ou seja, $l = L$, então $v^L{}_{p,j} = \tilde{y}_{p,r}$.

- Cálculo do vetor de erros:

$$\epsilon_{p,r} = y_{p,r} - \tilde{y}_{p,r}, \tag{3.13}$$

onde $y_{p,r}$ é o $r - ésimo$ elemento do vetor de saídas desejadas $\overrightarrow{Y}_p$.

Retropropagação:

- Cálculo dos elementos da matriz jacobiana

$$\begin{aligned} \frac{\partial \epsilon_{p,r}(w)}{\partial w_{j,i}^l} &= \frac{\partial \epsilon_{p,r}(w)}{\partial u_{p,j}^l} \frac{\partial u_{p,j}^l}{\partial w_{j,i}^l} \\ &= -\delta_{pr,j}^l v^{l-1}{}_{p,i} \end{aligned} \tag{3.14}$$

Conforme referido anteriormente, o processo de retropropagação é feito separadamente para cada saída r. Assim, define-se $\delta_{pr,j}^l$ como o erro retropropagado, calculado para cada neurônio j e cada saída r produzida por o par p. Se o neurônio j pertence à camada de saída, então

$$\begin{cases} \delta_{pr,j}^L = \dot{f}_j^L(u^L{}_{p,j}) & \text{se} \quad r = j \\ \\ \delta_{pr,j}^L = 0 & \text{se} \quad r \neq j \end{cases} \tag{3.15}$$

Para outras camadas

$$\delta_{pr,j}^l = \dot{f}_j^l(u^l{}_{p,j}) \sum_{c=1}^{n_{l+1}} \delta_{pr,c}^{l+1} w_{c,j}^{l+1}, \tag{3.16}$$

onde n_{l+1} é o número de neurônios da camada seguinte e $\dot{f}_j^l$ é a derivada da função de ativação do neurônio j respeito à ${u^l}_{p,j}$.

No caso do viés teremos

$$\frac{\partial \epsilon_{p,r}(w)}{\partial w_{j,0}^l} = -\delta_{pr,j}^l {v^{l-1}}_{p,0}, \tag{3.17}$$

onde ${v^{l-1}}_{p,0} = 1$.

Lembre-se que os elemento $w_{j,i}^l$ podem ser mapeados num só índice. Desta forma definiremos

$$\begin{aligned} w &= [w_{1,0}^1, w_{1,1}^1, ..., w_{n_1,n_0}^1, w_{1,0}^2, ..., w_{n_2,n_1}^2, ..., w_{n_L,n_{L-1}}^L] \\ &= [w_1, w_2, ..., w_K], \end{aligned} \tag{3.18}$$

onde K é o número total de neurônios na rede.

Este processo é repetido para cada uma das saídas e para cada par de treinamento. Uma vez terminado o processo, é formada a matriz $J(w)$ conforme apresentado em 3.9

- Os cálculos dos ajustes dos pesos são feito através da equação 3.10, onde μ é um parâmetro dinâmico, o qual é ajustado de acordo com a variação do erro. Se no instante atual há um incremento da função de erro $f_0(w)$, $\mu = \mu * \beta$. No caso em que $f_0(w)$ diminue, $\mu = \frac{\mu}{\beta}$. Uma escolha apropriada destes parâmetros segundo [35] é $\mu = 0.01$ e $\beta = 10$.

3.3 Considerações práticas

Ao implementar a rede neural devem-se tomar decisões quanto à estrutura e o processo de treinamento afim de melhorar o desempenho da rede. Algumas dessas decisões são destacadas a seguir:

- Estabelecer o número de camadas intermediárias, assim como o número de neurônios destas camadas. Conforme referido anteriormente, basta apenas uma camada intermediaria para aproximar qualquer tipo de função contínua. Importa destacar que não existe uma heurística estabelecida que permita definir o número de neurônios necessários, sendo que, geralmente, este é definido empiricamente.

- Ao iniciar o treinamento, os valores dos pesos iniciais podem ser definidos por valores aleatórios uniformemente distribuídos num intervalo definido.

- Um problema bastante freqüente é a perda de generalização, resultado de um treinamento excessivo (*overtraining*). Este problema acontece pelo fato de que a rede pode acabar memorizando os dados de treinamento, mas não as características da função a ser modelada. Um método amplamente utilizado na literatura para contornar este problema é dividir os dados em três conjuntos: treinamento, teste e validação. A distribuição dos dados é feita aleatoriamente, sendo que o tamanho do conjunto de treinamento deve ser maior, geralmente são consideradas proporções de $60\%, 20\% e 20\%$ do total dos dados para cada um dos conjuntos respectivamente. O conjunto de treinamento é o único utilizado para ajustar os pesos, enquanto que os outros são aplicados à rede (a cada iteração) com o propósito de calcular o erro. Com base neste erro, um critério de parada prematura do treinamento é estabelecido, i.e, o treinamento pode se interromper no ponto onde o erro global é menor. Isso garante que a rede atinja o menor erro possível sem que haja perda de generalização.

- É recomendável fazer uma seleção adequada das variáveis de entrada que possuem informações relevantes para caracterizar o problema, uma vez que o excesso de variáveis aumenta a dimensionalidade do espaço de busca, tornando computacionalmente ineficiente o uso da rede neural. Outro processo importante a ser feito é o pré-processamento dos dados o qual, de modo geral, envolve três etapas (Haykin, 2001): remoção da média, descorrelação das variáveis e normalização. A análise da correlação é útil para identificar variáveis com informação redundante e aquelas com informação relevante para caracterizar o problema.

- É recomendável realizar um processo de pos-treinamento no qual são identificadas anomalias como dados intrusos, regiões de baixa população, entre outras que podem comprometer o desempenho da rede. Além disso, pode ser estudada a possibilidade de poda de neurônios.

- Também é recomendável, no caso de problemas de classificação com múltiplas classes, utilizar codificação maximamente esparsa, i.e, cada saída da rede representa uma classe.

Capítulo 4

Filtro de Kalman

O filtro de Kalman (KF) parte da modelagem em espaço de estados de sistemas dinâmicos lineares, e dos pressupostos de normalidade tanto dos ruídos associados às equações do modelo (equação de processo e de medição) quanto do vetor inicial de estado. Sob estas condições o KF fornece uma solução recursiva para o problema de filtragem ótima linear, permitindo determinar o estimador do estado não viciado e ótimo, uma vez que a variabilidade dos erros (ou erro quadrático médio) é minimizada [12]. A solução é considerada recursiva porque cada estimativa atualizada do estado é calculada a partir da estimativa anterior e os novos dados de entrada. Portanto, apenas a estimativa anterior requer armazenamento, eliminando assim a necessidade de armazenar todo o passado dos dados observados [36]. O filtro de kalman permite encontrar estimativas do estado passado, presente e até mesmo futuro, inclusive quando a natureza real do sistema modelado é desconhecida [37].

Na prática, o estado ou o ambiente de interesse não evolui de forma linear e, além disso, as observações fornecidas pelos sistemas de sensores não se relacionam linearmente com as variáveis de estado, fazendo com que os modelos de processo e medição sejam não lineares, tornando o uso do filtro de Kalman padrão inadequado [38]. Nestas situações é utilizado o filtro de Kalman estendido (EKF), conhecido como um algoritmo subótimo de estimação de estado para sistemas não lineares. Esta abordagem utiliza técnicas como a expansão de Taylor para obter uma aproximação linear do modelo.

Uma outra situação freqüente em casos de acompanhamento de alvo é a possibilidade de que este experimente diferentes padrões de comportamento ao longo do tempo. Por exemplo, um alvo que trafega em um movimento retilíneo uniforme num determinado instante tem a possibilidade de efetuar uma manobra no instante seguinte. Neste caso, para se atingir um desempenho satisfatório é necessário algoritmos capazes de se ajustar a essas variações [13] pelo que a implementação de um único filtro de Kalman se torna ineficiente. O algoritmo de Múltiplos Modelos Interagentes (IMM) é considerado uma das abordagens que apresentam desempenhos

significativamente melhores em comparação com o KF [4, 13, 14] e demanda um custo computacional menor em relação a filtragem Bayesiana [39]. Sua principal vantagem reside no fato de permitir a utilização de mais de um modelo de KF em paralelo para descrever o movimento do alvo.

Neste capítulo são detalhados os algoritmos de filtro de Kalman padrão, de filtro de Kalman estendido e da abordagem IMM.

4.1 Filtro de Kalman padrão

Basicamente, a tarefa do filtro de Kalman consiste em encontrar uma estimativa do estado no instante t_{k+1}, $\hat{X}(t_{k+1}|t_{k+1})$, que minimize um critério de erro pré-estabelecido, no nosso caso o erro quadrático médio (MSE). Portanto, definindo o erro como a diferença entre o valor real do estado, $X(t_{k+1})$, e sua estimativa a posteriori, $\hat{X}(t_{k+1}|t_{k+1})$,

$$\varepsilon(t_{k+1}|t_{k+1}) = X(t_{k+1}) - \hat{X}(t_{k+1}|t_{k+1}), \tag{4.1}$$

o objetivo é minimizar a covariância dos erros desta estimativa, ou seja, minimizar

$$P(t_{k+1}|t_{k+1}) = E[\varepsilon(t_{k+1}|t_{k+1}).\varepsilon(t_{k+1}|t_{k+1})^T]. \tag{4.2}$$

Conforme referido anteriormente, o KF padrão parte da modelagem na forma das equações de espaço de estados de um sistema dinâmico linear e discreto no tempo, perturbado por ruído branco com distribuição de probabilidade Gaussiana. Esta modelagem consiste nas equações do processo e da medida, dadas respectivamente por:

$$X(t_{k+1}) = F(t_k)X(t_k) + G(t_k)u(t_k) + W(t_k) \tag{4.3}$$

$$Z(t_{k+1}) = H(t_{k+1})X(t_{k+1}) + e(t_{k+1}), \tag{4.4}$$

onde os componentes das equações acima são descritos como se segue:

- $X(t_k)$ é o vetor (n-dimensional) de estados;
- $F(t_k)$ é a matriz $(n \times n)$ de transição de estados;
- $W(t_k)$ é o vetor de ruído associado ao processo cujos componentes são assumidos variáveis aleatórias Gaussianas, independentes e identicamente distribuídas ($i.i.d$) com média zero e covariância $Q(t_k)$;
- $u(t_k)$ representa o grau de controle que observador tem sobre a dinâmica do alvo. Por exemplo, se a posição do sensor for utilizada como a origem do

sistema coordenada a partir da qual se determina a posição do alvo, então o movimento do sensor tem influência sobre a posição deste alvo. A matriz $G(t_k)$ controla os efeitos do $u(t_k)$. Conforme referido anteriormente, neste trabalho considera-se um observador estático, por isso, o termo $G(t_k)u(t_k)$ não é incorporado na equação do processo;

- $Z(t_{k+1})$ é o vetor (m-dimensional) de medida do alvo;
- $H(t_{k+1})$ é a matriz $(m \times n)$ de medidas (relaciona as variáveis do estado do alvo às medidas); e
- $e(t_{k+1})$ é o vetor de ruído associado a medida cujos componentes são assumidos variáveis aleatórias Gaussiana, *i.i.d.* com média zero e covariância $R(t_{k+1})$.

Por simplicidade, de agora em diante, a notação matemática de instante de tempo t_k será substituída pelo seu índice k. Por exemplo, o termo $X(t_{k+1})$ será substituído por $X(k+1)$.

Com base nestas equações o algoritmo do KF é desenvolvido. A execução deste algoritmo é feita em duas etapas recursivas: Previsão e atualização do estado [4].

a) **Previsão**

Sabemos que a média amostral é um estimador não viciado da média populacional [40], por conseguinte, podemos encontrar uma estimativa a priori do vetor de estados no instante $k+1$ baseada em todas as observações disponíveis até o instante k, $D_k = \{Z(0), Z(1), ..., Z(k)\}$. Assim, dispondo destas observações e aplicando o valor esperado da equação 4.3, teremos

$$E[X(k+1|D_k)] = \hat{X}(k+1|D_k) = F(k) * E[X(k|D_k)] + E[W(k)]. \quad (4.5)$$

Para simplicidade de anotação, substituiremos o termo $X(\cdot|D_k)$ por $X(\cdot|k)$. Considerando $E[W(k)] = 0$, a equação 4.5 é reescrita como

$$\hat{X}(k+1|k) = F(k) * E[X(k|k)], \quad (4.6)$$

onde $\hat{X}(k+1|k)$ é a estimativa a priori do vetor de estados no instante $k+1$ (ou estado previsto) e o termo $E[X(k|k)] = \hat{X}(k|k)$ é a estimativa a posteriori no instante k.

Seguindo o raciocínio anterior, aplicando o valor esperado da equação 4.4, teremos a estimativa a priori da medida no instante $k+1$ (substituindo o termo $Z(\cdot|D_k)$ por $Z(\cdot|k)$) dada por

$$\hat{Z}(k+1|k) = H(k+1) * \hat{X}(k+1|k). \quad (4.7)$$

Observe que esta equação representa uma projeção das variáveis do vetor de estados nas variáveis do vetor de medida.

De forma análoga à equação 4.2, a matriz de covariância dos erros da estimativa a priori do estado no instante $k+1$ é dada por

$$P(k+1|k) = E[\varepsilon(k+1|k).\varepsilon(k+1|k)^T] \tag{4.8}$$

onde o erro $\varepsilon(k+1|k)$ é definido como

$$\varepsilon(k+1|k) = X(k+1) - \hat{X}(k+1|k). \tag{4.9}$$

Substituindo 4.3 e 4.6 em 4.9 obtemos

$$\varepsilon(k+1|k) = F(k) * \Big(X(k) - \hat{X}(k|k)\Big) + W(k). \tag{4.10}$$

Por sua vez, substituindo 4.10 em 4.8 e lembrando que $E[W(k)] = 0$, teremos

$$\begin{aligned} P(k+1|k) = {} & F(k) * E[(X(k) - \hat{X}(k|k)).(X(k) - \hat{X}(k|k))^T] * F(k)^T \\ & + E[W(k).W(k)^T], \end{aligned} \tag{4.11}$$

onde o termo $E[W(k).W(k)^T]$ representa a matriz de covariância associada ao ruído do processo, $Q(k)$, e o termo $E[(X(k) - \hat{X}(k|k))(x(k) - \hat{X}(k|k))^T]$ é a covariância dos erros da estimativa a posterior do estado no instante k, $P(k|k)$. Desta forma,

$$P(k+1|k) = F(k) * P(k|k) * F(k)^T + Q(k). \tag{4.12}$$

b) **Atualização**

A estimativa a posteriori (atualizada) do estado no instante $k+1$, $\hat{X}(k+1|k+1)$, é obtida através de uma combinação linear da estimativa a priori do estado, $\hat{X}(k+1|k)$, e da diferença ponderada entre a última medida recebida, $Z(k+1)$, e sua estimativa a priori, $\hat{Z}(k+1|k)$, ou seja,

$$\hat{X}(k+1|k+1) = \hat{X}(k+1|k) + K(k+1) * \nu(k+1), \tag{4.13}$$

onde $\nu(k+1)$ é o vetor de inovação ou erro da estimativa da medida dado por

$$\nu(k+1) = Z(k+1) - \hat{Z}(k+1|k), \tag{4.14}$$

e $K(k+1)$ representa o ganho do filtro de Kalman. O valor de $K(k+1)$

é calculado de forma a encontrar uma estimativa a posteriori do estado que minimize o MSE, definido em 4.2. Desta forma, a primeira coisa a ser feita é expressar 4.2 em termos do ganho. Para isso, substituiremos 4.4 e 4.7 em 4.13 e o resultado em 4.1. Assim teremos

$$\varepsilon(k+1|k+1) = X(k+1) - \hat{X}(k+1|k) - K(k+1) * \Big(H(k+1) * X(k+1) + e(k+1) - H(k+1) * \hat{X}(k+1|k)\Big), \tag{4.15}$$

operando e reagrupando 4.15 encontramos que

$$\varepsilon(k+1|k+1) = \Big(I - K(k+1) * H(k+1)\Big) * \varepsilon(k+1|k) - K(k+1) * e(k+1), \tag{4.16}$$

onde $\varepsilon(k+1|k)$ é definido em 4.9. Substituindo 4.16 em 4.2, obtemos,

$$P(k+1|k+1) = E\Big[\Big((I - K(k+1) * H(k+1)) * \varepsilon(k+1|k) - K(k+1) * e(k+1)\Big)\Big((I - K(k+1) * H(k+1)) * \varepsilon(k+1|k) - K(k+1) * e(k+1)\Big)^T\Big], \tag{4.17}$$

operando e reagrupando temos que

$$P(k+1|k+1) = K(k+1) * R(k+1) * K(k+1)^T + \Big(I - K(k+1) * H(k+1)\Big) * P(k+1|k) * \Big(I - K(k+1) * H(k+1)\Big)^T, \tag{4.18}$$

onde o termo $R(k+1) = E[e(k+1) * e(k+1)^T]$ é a covariância do ruído associado à medida e $P(k+1|k)$ é definido em 4.8.

A equação 4.18 é uma função que, quando minimizada, fornecerá o valor de $K(k+1)$ que leva a estimativa ótima. Assim, com o propósito de minimizar 4.18, calculamos o traço desta equação,

$$\begin{aligned} Tr(P1) = {} & Tr(P2) - Tr(K(k+1) * H(k+1) * P2) \\ & - Tr(P2 * K(k+1)^T * H(k+1)^T) \\ & + Tr(K(k+1) * H(k+1) * P2 * K(k+1)^T * H(k+1)^T) \\ & + Tr(K(k+1) * R(k+1) * K(k+1)^T), \end{aligned} \tag{4.19}$$

onde $P1 = P(k+1|k+1)$ e $P2 = P(k+1|k)$, isto é feito por simplicidade de notação. Seguidamente derivamos 4.19 com respeito a $K(k+1)$ e igualamos a zero, ou seja,

$$\begin{aligned}\frac{dTr(P1)}{dK(k+1)} &= -(P2 * H(k+1))^T - (P2 * H(k+1)^T) \\ &+ K(k+1) * H(k+1) * P2 * H(k+1)^T \\ &+ K(k+1) * H(k+1)^T * P2^T * H(k+1) \\ &+ K(k+1) * R(k+1) + K(k+1) * R(k+1)^T = 0. \end{aligned} \tag{4.20}$$

Como $P2$ e $R(k+1)$ são matrizes simétricas, a equação anterior pode ser simplificada da seguinte forma:

$$\begin{aligned}0 = &-2 * P2 * H(k+1)^T + 2 * K(k+1) * H(k+1) * P2 * H(k+1)^T \\ &+ 2 * K(k+1) * R(k+1). \end{aligned} \tag{4.21}$$

Finalmente, isolando $K(k+1)$ e lembrando que $P2 = P(k+1|k)$ teremos

$$K(k+1) = P(k+1|k) * H(k+1)^T * S(k+1)^{-1}, \tag{4.22}$$

onde o termo $S(k+1)$ é a matriz de covariância do erro da estimativa da medida, definida por

$$S(k+1) = H(k+1) * P(k+1|k) * H(k+1)^T + R(k+1). \tag{4.23}$$

Dada 4.22, a equação 4.18 pode ser reduzida da seguinte forma:

$$P(k+1|k+1) = (I - K(k+1) * H(k+1)) * P(k+1|k). \tag{4.24}$$

Resumidamente, uma seqüencia do processo do KF é dada pelas equações 4.6, 4.12, 4.22, 4.13 e 4.24. É importante destacar que para a execução do KF assume-se que o vetor de estados inicial, $X(0)$, é uma variável aleatória Gaussiana, com parâmetros conhecidos, dados respectivamente pela média, $\hat{X}(0|0)$, e matriz de covariância, $P(0|0)$ [41]. O ciclo completo do procedimento do KF é apresentado na figura 4.1.

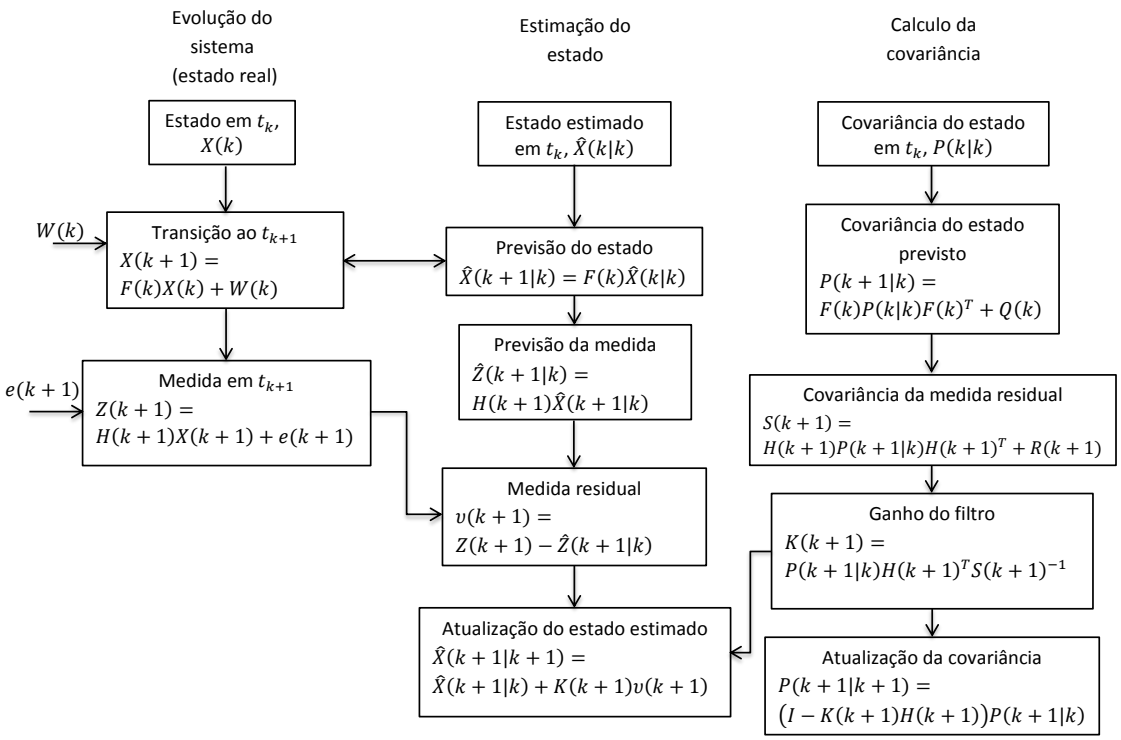


Figura 4.1: fluxograma KF (adaptado de [4])

4.2 Filtro de Kalman Estendido

Conforme destacado anteriormente, na prática, os sistemas dinâmicos não são lineares. Neste caso, a aplicação de KF pode ser estendida através de um procedimento de linearização. O filtro resultante é conhecido como EKF, cuja derivação depende da linearização do modelo de espaço de estado supondo que o desvio da linearidade é de primeira ordem. Sendo assim, as equações 4.3 e 4.4 são reescritas como

$$X(k+1) = f(k, X(k)) + W(k) \tag{4.25}$$

$$Z(k+1) = fh(k+1, X(k+1)) + e(k+1), \tag{4.26}$$

onde $f(k, X(k))$ é a função vetorial não linear de transição de estados e $fh(k+1, X(k+1))$ é a função vetorial não linear que relaciona as variáveis de estado às medidas. Assim, estas funções são definidas como

$$f(k, X(k)) = \begin{bmatrix} f_1(k, X(k)) \\ f_2(k, X(k)) \\ \vdots \\ f_n(k, X(k)) \end{bmatrix} \tag{4.27}$$

$$fh(k+1, X(k+1)) = \begin{bmatrix} fh_1(k+1, X(k+1)) \\ fh_2(k+1, X(k+1)) \\ \vdots \\ fh_m(k+1, X(k+1)) \end{bmatrix} \tag{4.28}$$

onde n é o número de elementos do vetor de estados e m é o número de elementos do vetor de medidas. É importante ressaltar que basta que uma destas funções seja não linear para aplicar a metodologia do EKF. Para tratar o problema não linear, este filtro utiliza a expansão em series de Taylor de primeira ordem no entorno da estimativa atual como uma forma de aproximar linearmente estas funções, ou seja,

$$f(k, X(k)) \approx f(k, \hat{X}(k|k)) + J[f(k, \hat{X}(k|k))] * (X(k) - \hat{X}(k|k)) \tag{4.29}$$

$$fh(k+1, X(k+1)) \approx fh(k+1, \hat{X}(k+1|k)) + J[fh(k+1, \hat{X}(k+1|k))] * (X(k+1) - \hat{X}(k+1|k)), \tag{4.30}$$

onde $J[\cdot]$ é a matriz Jacobiana da função. Por uma questão de simplicidade, faremos $J[f(k, \hat{X}(k|k))] = F(k)$ e $J[fh(k+1, \hat{X}(k+1|k))] = H(k+1)$. Os componentes destas matrizes são calculados respectivamente por:

$$f_{dq} = \frac{\partial(f_d)}{\partial X_q} \tag{4.31}$$

$$h_{pq} = \frac{\partial(fh_p)}{\partial X_q} \tag{4.32}$$

onde $d = 1, 2, ..., n$, $p = 1, 2, ..., m$ e $q = 1, 2, ..., n$. Assim, as matrizes são estabelecidas como:

$$F = \begin{bmatrix} f_{11} & \cdots & f_{1n} \\ f_{21} & \cdots & f_{2n} \\ \vdots & \cdots & \vdots \\ f_{n1} & \cdots & f_{nn} \end{bmatrix} \tag{4.33}$$

$$H = \begin{bmatrix} h_{11} & \cdots & h_{1n} \\ h_{21} & \cdots & h_{2n} \\ \vdots & \cdots & \vdots \\ h_{m1} & \cdots & h_{mn} \end{bmatrix}. \tag{4.34}$$

Substituindo as equações 4.29 e 4.30 em 4.25 e 4.26, encontramos as equações lineares de processo e de medida, dadas respectivamente por:

$$X(k+1) = f(k, \hat{X}(k|k)) + F(k) * X(k) - F(k) * \hat{X}(k|k) + W(k) \tag{4.35}$$

$$Z(k+1) = fh(k+1, \hat{X}(k+1|k)) + H(k+1) * X(k+1) - H(k+1) * \hat{X}(k+1|k) + e(k+1) \tag{4.36}$$

A partir destas equações, aplicando o mesmo procedimento feito para a dedução das equações do KF padrão, obtemos o conjunto de equações que formam a solução recursiva do EKF.

a) **Previsão do estado**

Estimativa a priori do estado:

$$\hat{X}(k+1|k) = f(k, \hat{X}(k|k)). \tag{4.37}$$

Estimativa a priori da medida:

$$\hat{Z}(k+1|k) = fh(k+1, \hat{X}(k+1|k)). \tag{4.38}$$

Matriz de covariância dos erros da estimativa a priori do estado:

$$P(k+1|k) = F(k) * P(k|k) * F(k)^T + Q(k). \tag{4.39}$$

Ganho do filtro:

$$K(k+1) = P(k+1|k) * H(k+1)^T * \Big(H(k+1) * P(k+1|k) * H(k+1)^T + R(k+1) \Big)^{-1}. \tag{4.40}$$

b) **Atualização da estimativa:**

Estimativa a posteriori do estado:

$$\hat{X}(k+1|k+1) = \hat{X}(k+1|k) + K(k+1) * (\nu(k+1)), \tag{4.41}$$

onde $\nu = Z(k+1) - \hat{Z}(k+1|k)$ é o vetor de inovação ou medida residual. Matriz de covariância dos erros da estimativa atualizada:

$$P(k+1|k+1) = (I - K(k+1) * H(k+1)) * P(k+1|k). \tag{4.42}$$

Tal como no KF padrão, para iniciar a execução do algoritmo, é necessário definir a estimativa do estado inicial $\hat{X}(0|0)$ e sua matriz de covariâncias $P(0|0)$, considerando

$X(0)$ uma variável aleatória Gaussiana.

O ciclo completo do procedimento do EKF é apresentado na figura 4.2.

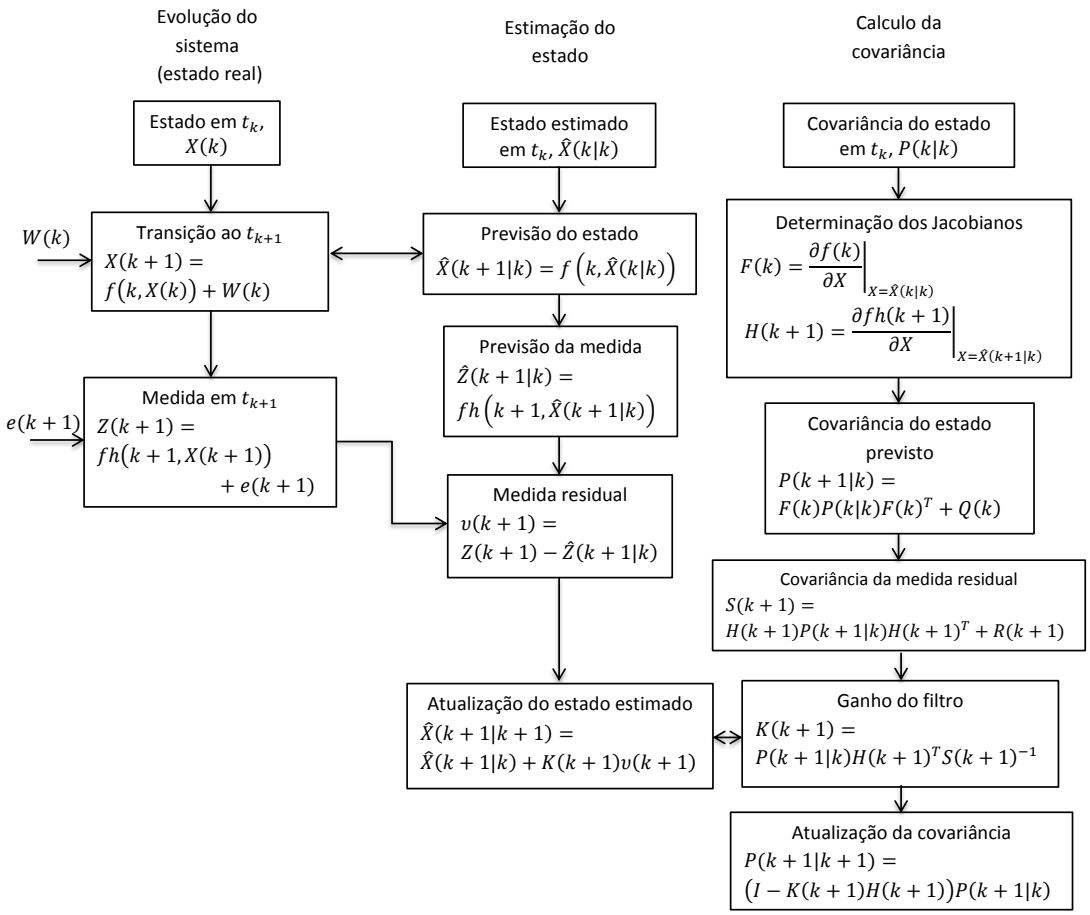


Figura 4.2: fluxograma EKF (adaptado de [4])

4.3 IMM

O IMM é um algoritmo proposto por Blom e Bar-shalom em 1988 [15], cuja principal característica é a habilidade de estimar o estado de um sistema dinâmico e sua respectiva matriz de covariância através da soma ponderada das estimativas de N modelos de KF executados em paralelo [12, 42].

A implementação do algoritmo IMM segue as etapas de operação de uma cadeia de Markov, na qual são calculados os fatores de ponderação utilizados para a combinação das estimativas de cada um dos modelos de filtro.

Antes de descrever os procedimentos do algoritmo IMM, definimos os seguintes termos [43]:

- $\mu_i(k|k)$: probabilidade de que o movimento do alvo seja caracterizado pelo estado do modelo i no tempo k;

- $\mu_{i|j}(k|k)$: probabilidade condicional de que o alvo transite do estado do modelo i para o estado do modelo j no tempo k, denominada por probabilidade de mistura;
- $\mu_j(k+1|k)$: probabilidade prevista de que o alvo esteja no modelo j no tempo $k+1$;
- $\hat{X}_i$: estado estimado pelo modelo de filtro i;
- P_i: matriz de covariância do modelo de filtro i;
- M_j: Modelo de filtro j.

Segundo DANG *et al.* [44], um ciclo prático do algoritmo IMM consiste nos seguintes passos:

1. **Definição da matriz de transição**:

   Inicialmente, são definidas as probabilidades de transição p_{ij}, i.e, probabilidade de que o movimento do alvo seja caracterizado pelo estado do modelo j no instante t dado que no instante $t-1$ era caracterizado pelo estado do modelo i. Estas probabilidades, consideradas estáticas e conhecidas a priori, podem ser expressas em forma de uma matriz P, conhecida como matriz de transição (4.43).

$$P = \begin{bmatrix} p_{11} & \cdots & p_{1j} \\ \vdots & \ddots & \vdots \\ p_{i1} & \cdots & p_{ij} \end{bmatrix} \tag{4.43}$$

   Uma forma de estabelecer estas probabilidades segundo BAR-SHALOM e LI [4] e BLACKMAN e POPOLI [43], é computar

$$p_{ii} = 1 - \frac{T}{\tau_i} \tag{4.44}$$

   onde τ_i é o tempo esperado de permanência no modelo i e T é o tempo de rotação de antena. Os elementos fora da diagonal, p_{ij} $(i \neq j)$,são selecionados segundo critério do designer respeitando a seguinte restrição

$$\sum_{j \neq i} p_{ij} = 1 - p_{ii}. \tag{4.45}$$

   É importante destacar que a escolha das probabilidades de transição não tem influência significativa no desempenho do algoritmo IMM. No entanto, ela fornece um grau de equilíbrio (*trade-off*) entre o nível máximo dos erros de

estimação no início da manobra e a redução máxima dos erros de estimação durante o movimento uniforme [45].

2. **Cálculo da probabilidade de mistura**:

   Neste passo é calculada a probabilidade de mistura, $\mu_{i|j}(k|k)$ da seguinte maneira

$$\mu_{i|j}(k|k) = \frac{p_{ij}\mu_i(k|k)}{\mu_j(k+1|k)}, \tag{4.46}$$

   onde $i, j = 1, ..., r$ e

$$\mu_j(k+1|k) = \sum_{i=1}^{r} p_{ij}\mu_i(k|k). \tag{4.47}$$

   Considerando que, neste passo é necessária a definição de um valor inicial para à probabilidade do modelo, $\mu_i(0|0)$, para cada um dos r modelos.

3. **Cálculo da mistura inicial do estado e da covariância** :

   Iniciando com o estado estimado previo, $\hat{X}_i(k|k)$, obtido como saída desde cada um dos r diferentes modelos de KF no tempo k, e correspondentes matrizes de covariância $P_i(k|k)$, a condição inicial de mistura para o modelo de filtro M_j no tempo k é calculada como:

$$\hat{X}_j^0(k|k) = \sum_{i=1}^{r} \hat{X}_i(k|k)\mu_{i|j}(k|k), \tag{4.48}$$

$$P_j^0(k|k) = \sum_{i=1}^{r} \mu_{ij}(k|k)\Big\{P_i(k|k) + [\hat{X}_i(k|k) - \hat{X}_j^0(k|k)] \cdot [\hat{X}_i(k|k) - \hat{X}_j^0(k|k)]^T\Big\} \quad . \tag{4.49}$$

   onde $\hat{X}_j^0(k|k)$ e $P_j^0(k|k)$ são respectivamente as misturas iniciais da estimativa do estado e da matriz de covariância.

4. **Atualização da probabilidade do modelo, $\mu_j(k+1|k+1)$:**

   A probabilidade de que o movimento do alvo seja caracterizado pelo modelo de filtro M_j é calculada através da função de verossimilhança da inovação, i.e,

$$\begin{aligned}\Lambda_j(k+1) &= N[\nu_j(k+1), 0, S_j(k+1)] \\ &= (2\pi)^{-\frac{m}{2}} \cdot |S_j(k+1)|^{-\frac{1}{2}} \cdot \exp\left[-\frac{1}{2} \cdot d_j^2\right],\end{aligned} \tag{4.50}$$

   onde

$$d_j^2 = \nu_j(k+1)^T * S_j(k+1)^{-1}\nu_j(k+1), \tag{4.51}$$

   $\nu_j(k+1)$ é o vetor de inovação da medida m dimensional, dado por 4.14, com

distribuição Gaussiana, de média 0 e matriz de covariância $S_j(k+1)$, dada por 4.23. Utilizando a regra de Bayes, a atualização das probabilidades do modelo é dada por

$$\mu_j(k+1|k+1) = \frac{p_j(k+1)}{\sum_{j=1}^{r} p_j(k+1)} \tag{4.52}$$

onde r é o numero de modelos e

$$p_j(k+1) = \Lambda_j(k+1) * \mu_j(k+1|k) \tag{4.53}$$

5. **Calculo da saída da filtragem IMM:**

   Finalmente, a estimativa do estado do alvo e sua respectiva matriz de covariância são obtidas respectivamente por

$$\hat{X}(k+1|k+1) = \sum_{j=1}^{r} \hat{X}_j(k+1|k+1)\mu_j(k+1|k+1), \tag{4.54}$$

$$P(k+1|k+1) = \sum_{j=1}^{r} \mu_j(k+1|k+1)\Big\{P_j(k+1|k+1) + [\hat{X}_j(k+1|k+1) - \hat{X}(k+1|k+1)] \cdot [\hat{X}_j(k+1|k+1) - \hat{X}(k+1|k+1)]^T\Big\}. \tag{4.55}$$

   Isto é feito unicamente com o propósito de obter a saída do algoritmo IMM.

O diagrama do algoritmo IMM é apresentado na figura 4.3.

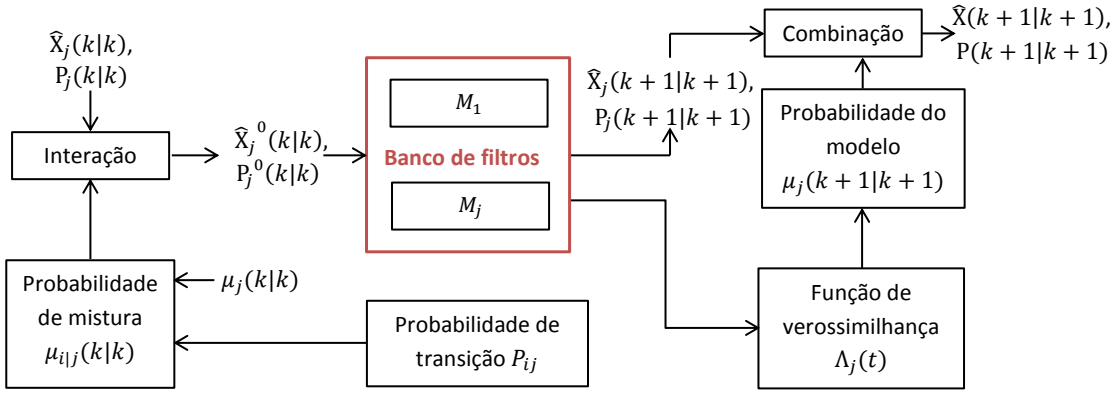


Figura 4.3: Diagrama do algoritmo IMM (adaptado de [4])

Capítulo 5

Metodologia

Tendo em vista os objetivos desta dissertação, é desenvolvido um método para abordar o problema de acompanhamento de múltiplos alvos (MTT) a partir de informações ruidosas, provenientes de um único sensor Radar. O método consiste basicamente em três etapas: na primeira, os acompanhamentos preliminares são criados, em seguida, ao se receber novas medidas é feita a atualização dos acompanhamentos mediante uma rede neural e por último um processo de filtragem é aplicado com o objetivo de encontrar a aproximação da trajetória real do alvo que é de interesse para o usuário do sistema. O esquema funcional do método desenvolvido neste trabalho é apresentado na figura 5.1

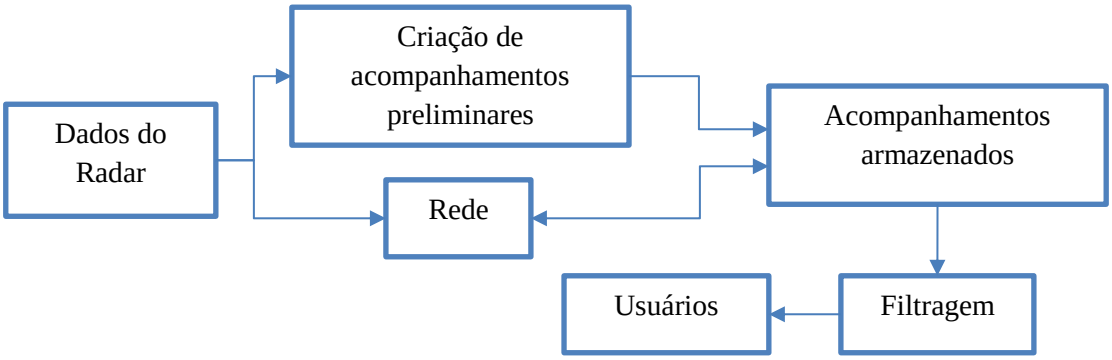


Figura 5.1: Método Proposto

Para o desenvolvimento deste trabalho foi utilizado o *software* Matlab, por ser um ambiente que possui funções desenvolvidas especificamente para o processamento de dados, implementação de redes neurais, além de permitir operações matriciais com grande volumes de dados, entre outras características.

Como produto desta dissertação, foram desenvolvidos dois programas com funções distintas, sendo que um tem a tarefa de treinar a rede neural que será utilizada para atualizar os acompanhamentos e o outro serve para testar o

funcionamento do método proposto. No primeiro programa é gerado um arquivo que contém a estrutura e os parâmetros da rede treinada e no segundo são inicializados e atualizados cada um dos acompanhamentos contidos no cenário, além de se realizar a comparação entre o seu desempenho e o do método convencional descrito no capítulo 2.

Neste capítulo são descritos os dados utilizados para o treinamento da rede neural e a validação do método proposto. Além disso, são descritas as etapas que compõem o método proposto e as métricas utilizadas para compara-lo com o método descrito no capítulo 2.

5.1 Medidas do sistema Radar

Nesta dissertação é utilizado um sistema Radar de busca de superfície com modulação por pulsos, conhecido por Radar de pulsos. Este tipo de Radar é um dos mais utilizados na navegação marítima. A sua antena gira entorno da sua posição de forma a "varrer" 360°, e o tempo que toma em dar uma volta é conhecido como tempo de rotação de antena.

O sistema Radar é baseado no princípio de propagação das ondas eletromagnéticas, descrito nas equações de Maxwell (1981). Basicamente o funcionamento do Radar de pulsos consiste na emissão de pulsos de ondas eletromagnéticas os quais ao interagir com algum objeto são refletidos de volta ao sensor. Tomando a medida do tempo em que o pulso volta ao sensor e pelo fato de, ignorando condições externas, a velocidade de propagação destas ondas ser uma constante conhecida (próxima da velocidade da luz), pode se inferir a distância do objeto ao sensor. Enquanto que o azimuth é determinado pela orientação da antena no instante de recepção do pulso refletido pelo objeto. Desta forma é obtida a localização do objeto em coordenadas polares a cada instante k de tempo discreto. O tempo entre as medições é uma variável que depende da velocidade, direção do navio e do tempo de rotação de antena [46]. Estas medidas, assumidas não tendenciosas, são representadas em forma de vetor

$$Z(k) = \begin{bmatrix} d(k) \\ \theta(k) \end{bmatrix}, \tag{5.1}$$

onde $d(k)$ é a distância (em metros) do alvo ao observador e $\theta(k)$ é o azimuth (em graus) medido desde o norte geográfico ao centro do alvo, conforme ilustrado na figura 5.2. É sabido que estas medidas são perturbadas por ruídos, na distância, $e_d(k)$, e no azimuth, $e_\theta(k)$, os quais são assumidos independentes e estacionários com distribuição Gaussiana de media zero e variâncias constantes σ_d^2 e σ_θ^2 respectivamente

[47]. Assim, matematicamente $Z(k)$ pode ser expressa como

$$Z(k) = \begin{bmatrix} d(k) \\ \theta(k) \end{bmatrix} = \begin{bmatrix} d_r(k) \\ \theta_r(k) \end{bmatrix} + e(k), \tag{5.2}$$

onde $d_r(k)$ e $\theta_r(k)$ representam a localização do alvo perturbada pelo ruído de processo, e

$$e(k) = \begin{bmatrix} e_d(k) \\ e_\theta(k) \end{bmatrix}. \tag{5.3}$$

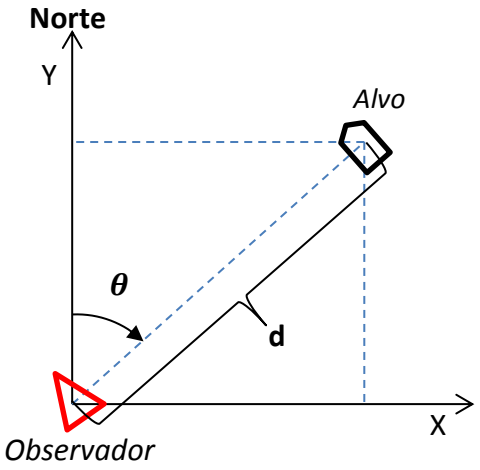


Figura 5.2: Definições da distância (d) e ângulo (θ)

Segundo MINGUENS [46] e KANG [47] , ignorando as condições externas, o ruído $e_d(k)$ se deve principalmente à largura de pulso. Entende-se por largura de pulso a duração de cada pulso transmitido, expressa em microsegundos ou em termos de distância multiplicando a velocidade de propagação das ondas eletromagnéticas (aproximadamente $3*10^8$ m/seg, velocidade da luz) pelo tempo de duração do pulso. Esta distância é conhecida como comprimento do pulso. A verdadeira posição do objeto se encontra em qualquer lugar entre mais ou menos metade do comprimento. A partir desta largura de pulso também são definidos os alcances máximos e mínimos do radar e a resolução em distância do mesmo. Esta resolução se refere à distância mínima de separação entre dois objetos com o mesmo azimuth, para que não sejam detectados pelo radar como um único contato. Enquanto que o ruído $e_\theta(k)$ se deve à largura do feixe no plano horizontal. Entende-se por largura de feixe a abertura angular entre os pontos de meia potência (-3dB). Esta largura depende da freqüência da energia transmitida pelo pulso e da forma e dimensões da antena. O objeto pode estar localizado na borda superior ou inferior ou em algum lugar dentro da largura do feixe. Além disso, a largura do feixe serve para definir a resolução em marcação do radar, ou seja, estabelecer o ângulo mínimo entre dois objetos situados na mesma

distância, para que não sejam detectados como um único contato. Para mais detalhes de funcionamento do sensor Radar consultar [47].

5.2 Geração de dados

Pelo fato de não se ter a disposição dados reais provenientes do sensor radar para realizar o treinamento da rede neural e validação do método proposto, foi necessário utilizar um gerador artificial de dados. Com este propósito, foram empregados dois programas nomeadamente gerador de trajetória de alvos e gerador de cenários. Estes programas foram fornecidos pelo Instituto de Pesquisa da Marinha (IPqM) apenas para fins de realização desta dissertação.

O gerador de trajetória permite criar trajetórias coerentes de alvos com diferentes padrões de movimentos. A trajetória é criada em seções, onde cada seção representa um padrão de movimento. Para cada uma destas seções é possível alterar parâmetros como velocidade, aceleração (tangencial e normal) e sua duração. Na figura 5.3 é apresentado um exemplo de uma trajetória gerada através deste programa. Neste exemplo, a trajetória gerada é baseada em três seções: a primeira representa um movimento retilíneo uniforme, seguida por um movimento curvilíneo e finalizando com um movimento retilíneo uniformemente acelerado. É importante destacar que, inicialmente, as trajetórias aqui geradas não apresentam ruído de processo ou de medição. Para o nosso caso, foram geradas trajetórias com uma velocidade máxima de 200 m/seg, velocidade mínima de 70 m/seg, aceleração tangencial máxima de 5 m/seg^2 e taxa de guinada máxima de 5 grau/seg.

No programa gerador de cenários são acrescentados tanto os ruídos de processo como os de medição. O ruído de processo é visto como uma perturbação na aceleração e supõem-se que tem média zero e covariância constante. Ele é incrementado à trajetória em coordenadas cartesianas. Quanto ao ruído de medição, o programa permite ajustar o desvio da distribuição dos ruídos segundo as características do sensor que se quer modelar. Além disso, permite definir parâmetros como alcances máximo e mínimo do Radar, a quantidade de clutter por varredura, tempo entre varreduras, número de sensores, número de alvos no cenário, entre outros. Assim, nesta dissertação, foi estabelecido um desvio da distribuição dos ruídos de processo de 2 m/seg^2 para as componentes x e y, e um desvio de 30 m na distância e de 3 mrad no azimuth para o ruído de medição. O alcance máximo estabelecido é de 30 Km e o mínimo é de 5 Km. O tempo de rotação da antena é de 2 seg e a quantidade de clutters considerada é de 100 pontos por varredura, sendo que foi utilizado um único sensor com múltiplos alvos por cenário.

É importante destacar que quanto maior a distância entre a localização do alvo e o centro do observador maior é a perturbação na medição feita pelo Radar. A

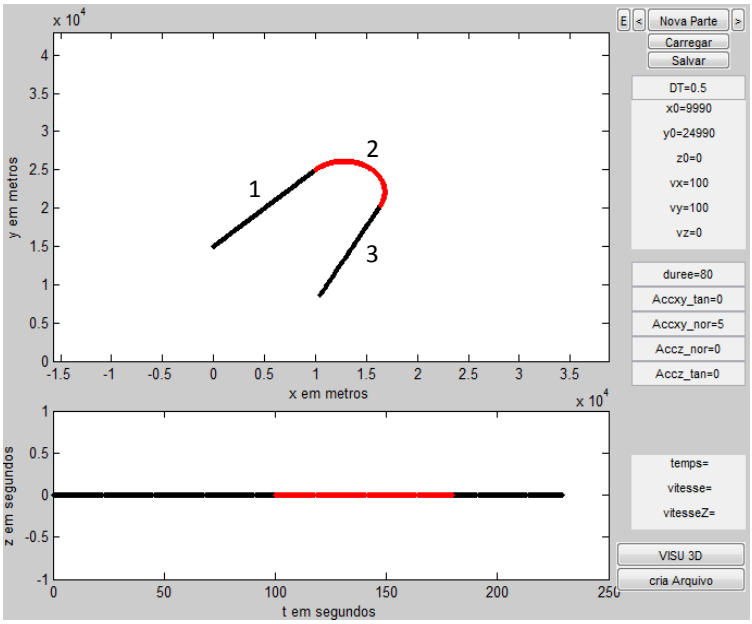


Figura 5.3: Exemplo gerador de trajetória

interface gráfica deste programa é apresentada na figura 5.4.

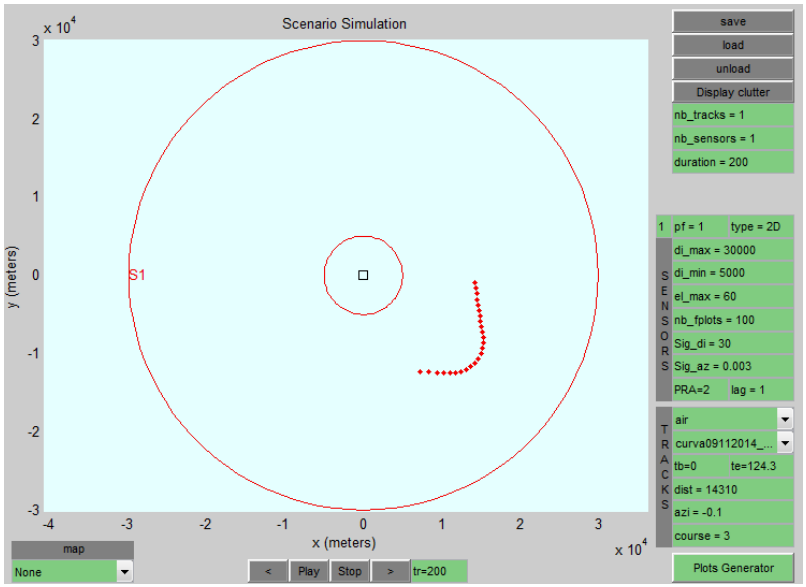
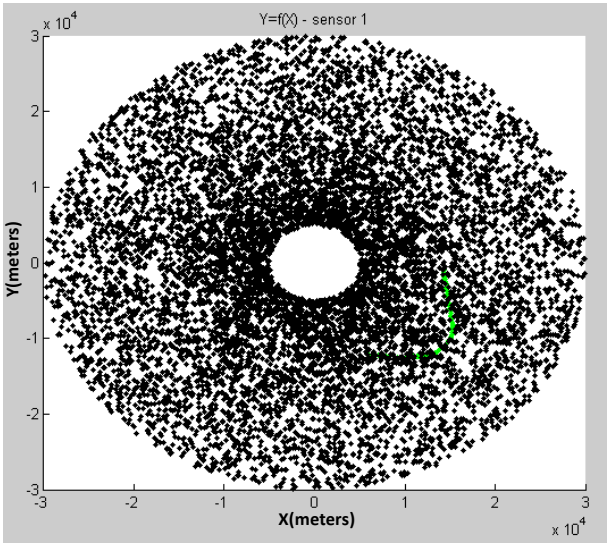


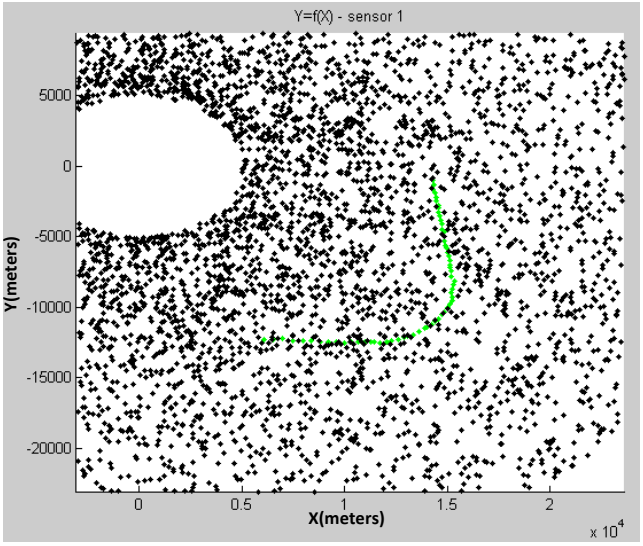
Figura 5.4: Interface do programa gerador de cenários

A cada momento em que o programa é executado, um cenário diferente com ruído aleatório é gerado. Exemplo de um cenário gerado com os parâmetros acima definidos é mostrado na figura 5.5. Nesta figura, os pontos de cor verde representam a trajetória do alvo e o ponto preto o clutter gerado. Como resultado da geração do cenário, é formada uma matriz que contem informações referentes à distância e azimuth de cada um dos contatos contidos no cenário e o tempo de detecção desses

contatos. Adicionalmente, é formado um vetor que contem informação sobre o tipo de contato, i.e, se o contato é clutter ou se é gerado pelo alvo.



(a) Cenário gerado



(b) Ampliação da imagem

Figura 5.5: Exemplo de geração de cenário

Uma situação que se deve levar em conta na geração dos dados é a possibilidade de acontecer perdas de detecção do alvo por algumas varreduras. Assim, para o nosso caso, opta-se por gerar trajetórias com um máximo de três perdas consecutivas. A escolha deste número de perdas foi intuitiva pelo fato de que quanto maior o número de perdas consecutivas maior será a incerteza na associação dos contatos.

5.2.1 Descrição dos cenários gerados para o treinamento da Rede neural

i. Cenário 1:

O cenário 1 utiliza um alvo que descreve um movimento retilíneo com velocidade constante de 99 m/seg durante os primeiros $50seg$, seguido de um movimento retilíneo com aceleração tangencial de 5 m/seg^2, permanecendo nesta aceleração durante 26 seg (figura 5.6). Assim, a execução do cenário tem uma duração de 76 seg, o que equivale a 38 varreduras de antena.

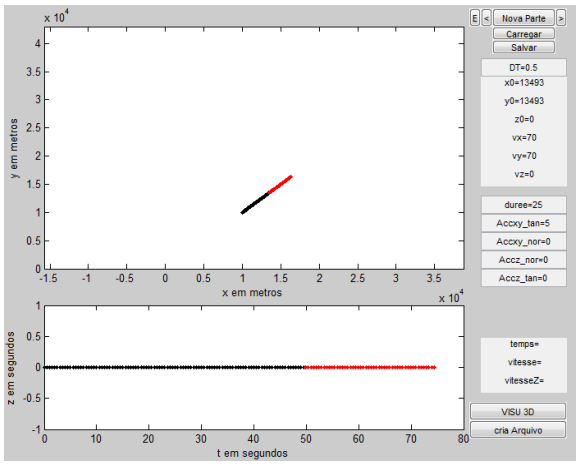


Figura 5.6: Trajetoria 1

A figura 5.7 mostra a trajetória inserida no programa gerador de cenários e na figura 5.8 é apresentado o cenário gerado. A figura 5.9 mostra a trajetória do alvo com adição do ruído de processo e de medição resultante deste cenário omitindo os contatos referentes ao clutter.

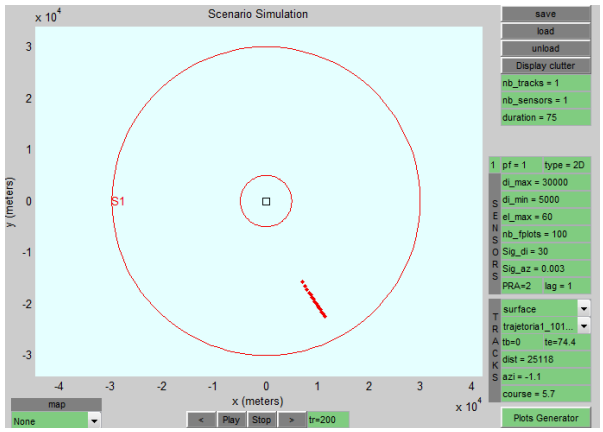


Figura 5.7: Trajetória 1 inserida no programa Gerador de cenários

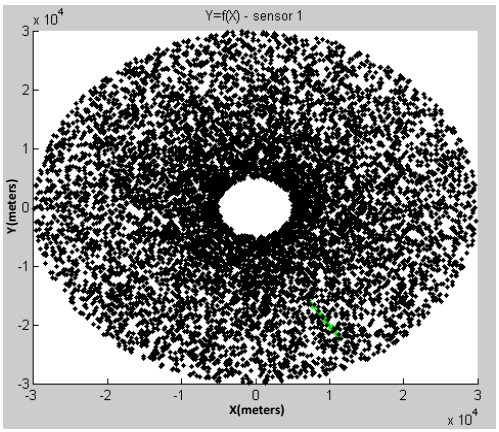


Figura 5.8: Cenário gerado 1

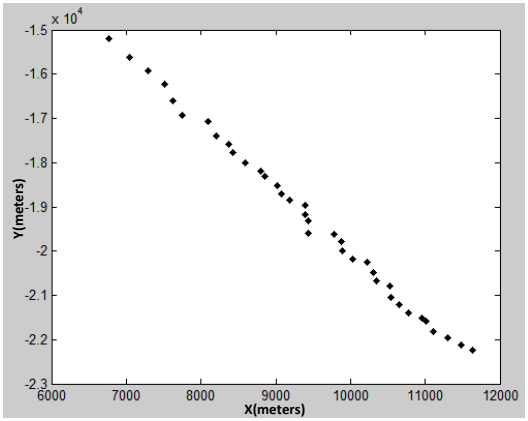


Figura 5.9: Trajetória resultante da geração do cenário 1

ii. Cenário 2:

O cenário 2 utiliza a mesma trajetória descrita no cenário 1 (figura 5.6), porém inserida num local diferente no programa gerador de cenários (figura 5.10). Além de isso, são introduzidas perdas de detecção nos intervalos de tempo apresentados na tabela 5.1. Nesta tabela também são apresentadas especificamente as varreduras onde foram introduzidas as perdas de detecção.

O cenário gerado é apresentado na figura 5.11. A trajetória do alvo com adição do ruído de processo e de medição resultante da geração do cenário omitindo o clutter é apresentada na figura 5.12.

Tabela 5.1: Intervalos de tempo e varreduras com perda de detecção no cenário2

Tempo (seg)	Varredura
$[12-18]$	7 a 8
$[28-36]$	15 a 17
$[46-52]$	24 a 25
$[58-62]$	30
$[70-78]$	36 a 38

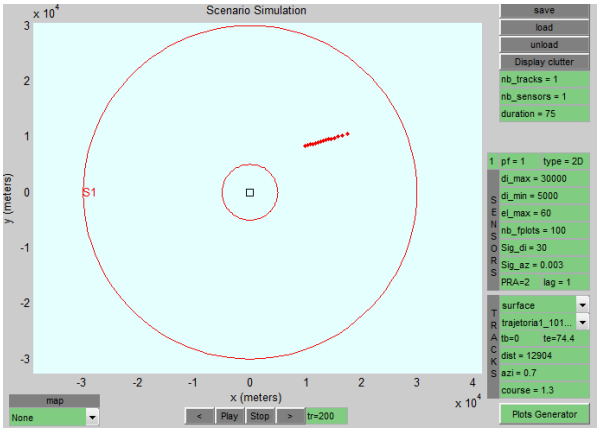


Figura 5.10: Trajetória 1 inserida num local diferente no programa gerador de cenários

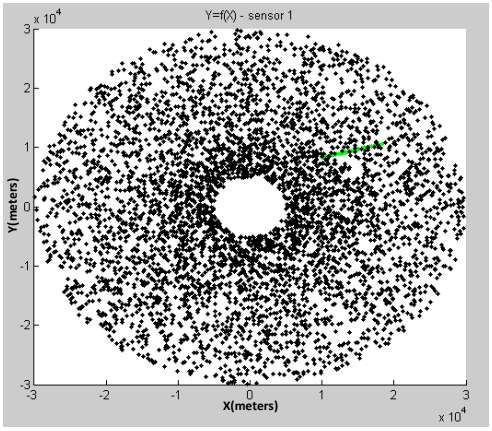


Figura 5.11: Cenário gerado 2

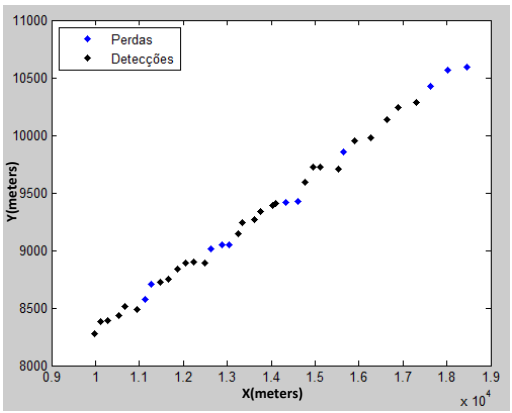


Figura 5.12: Trajetória resultante da geração do cenário 2

iii. Cenário 3:

O cenário 3 utiliza um alvo que descreve um movimento retilíneo com velocidade constante de 99 m/seg durante os primeiros 80 seg, seguido de um movimento curvilíneo com aceleração normal de 5 m/seg^2 (taxa de giro de 2.89 graus/seg) durante 100 / seg e finalmente o alvo descreve um movimento retilíneo com velocidade constante de 99 m/seg durante 80 seg (figura 5.13). Assim, o cenário tem uma duração de 260 seg ou seja 130 varreduras de antena.

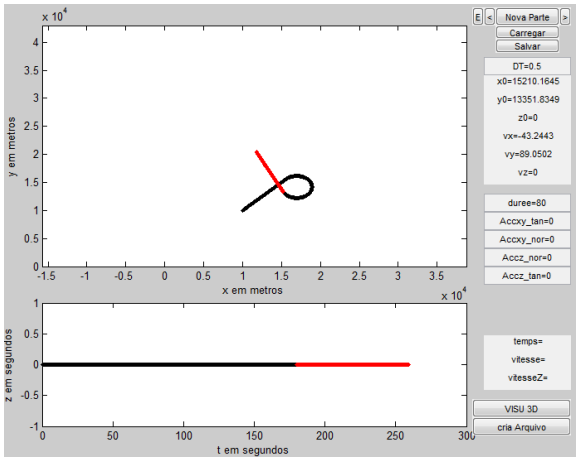


Figura 5.13: Trajetória 2

Os intervalos de tempo e as respectivas varreduras onde são introduzidas as perdas de detecção são apresentadas na tabela 5.2

A figura 5.14 mostra a trajetória inserida no programa gerador de cenários e o cenário gerado é apresentado na figura 5.15. A trajetória do alvo com adição do ruído de processo e de medição resultante deste cenário incluindo as perdas de detecção e ignorando o clutter é apresentada na figura 5.16.

Tabela 5.2: Intervalos de tempo e varreduras com perda de detecção no cenário 3

Tempo (seg)	Varredura
$[12-18]$	7 a 8
$[28-36]$	15 a 17
$[46-52]$	24 a 25
$[58-62]$	30
$[70-78]$	36 a 38
$[82-88]$	42 a 43
$[92-96]$	47
$[100-108]$	51 a 53
$[198-206]$	100 a 102
$[228-232]$	115

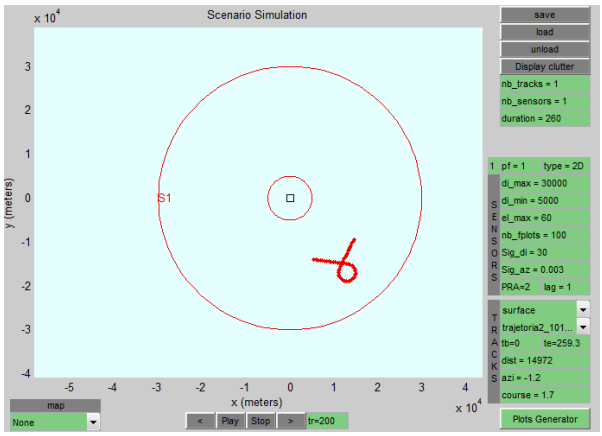


Figura 5.14: Trajetória 2 inserida no programa gerador de cenários

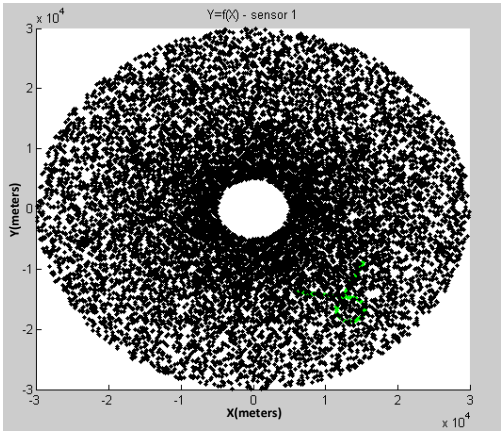


Figura 5.15: Cenário gerado 3

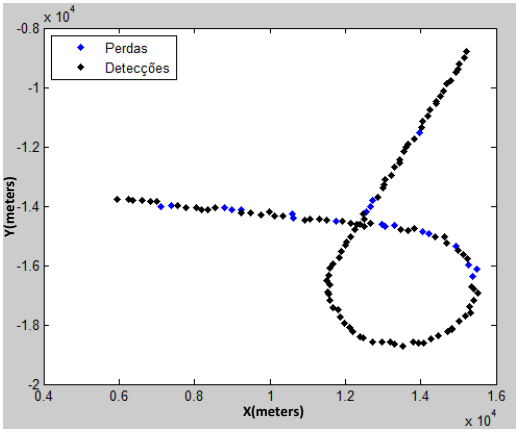


Figura 5.16: Trajetória resultante da geração do cenário 3

iv. Cenário 4:

O cenário 4 utiliza a trajetória descrita no cenário 3 (figura 5.13) considerando as mesmas perdas de detecção, porém inserindo-a num local diferente no programa gerador de cenário (figura 5.17).

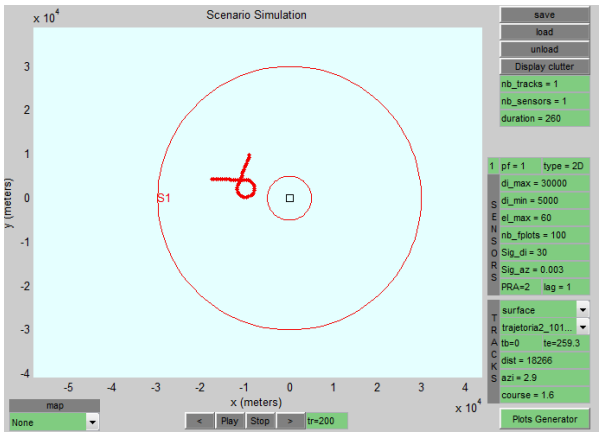


Figura 5.17: Trajetória 2 inserida num local diferente no programa gerador de cenários

O cenário gerado é apresentado na figura 5.18. A trajetória do alvo com adição do ruído de processo e de medição resultante deste cenário omitindo o clutter é apresentada na figura 5.19.

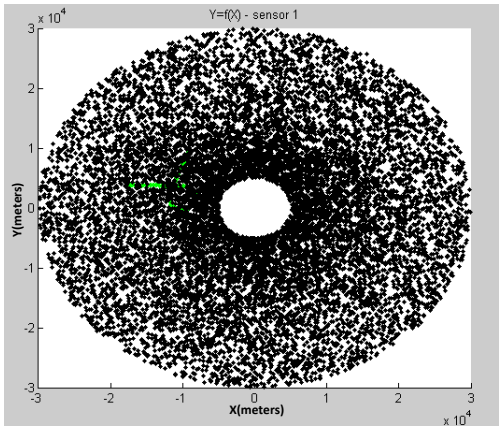


Figura 5.18: Cenário gerado 4

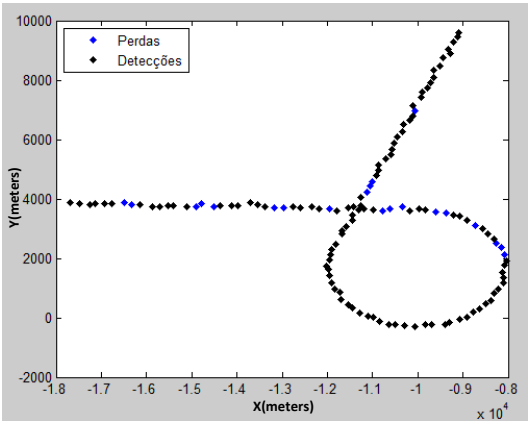


Figura 5.19: Trajetória resultante da geração do cenário 4

v. Cenário 5:

O cenário 5 utiliza um alvo que descreve um movimento curvilíneo com velocidade constante de 85 m/seg e aceleração normal de 3 m/seg^2 (taxa de giro de 2.026 grau/seg) durante 100 seg, ver figura 5.20. Assim, o cenário é composto por 50 varreduras de antena.

Os intervalos de tempo e as respectivas varreduras onde são introduzidas as perdas de detecção são apresentadas na tabela 5.3.

A figura 5.21 mostra a trajetória inserida no programa gerador de cenários e o cenário gerado é apresentado na figura 5.22. A trajetória do alvo com adição do ruido de processo e de medição resultante deste cenário omitindo o clutter é apresentada na figura 5.23.

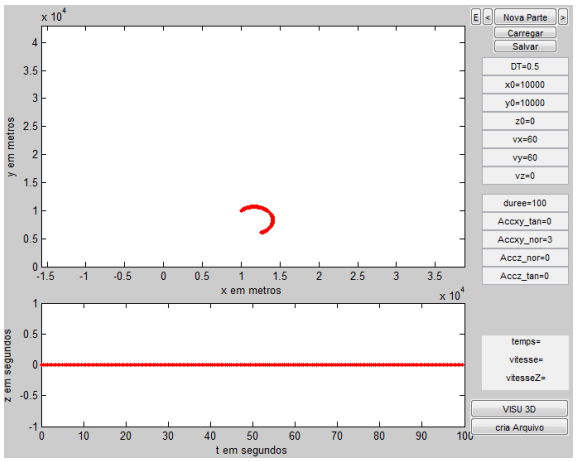


Figura 5.20: Trajetória 3

Tabela 5.3: Intervalos de tempo e varreduras com perda de detecção no cenário 5

Tempo (seg)	Varredura
$[12-18]$	7 a 8
$[28-36]$	15 a 17
$[46-52]$	24 a 25
$[58-62]$	30
$[70-78]$	36 a 38
$[82-88]$	42 a 43
$[92-96]$	47

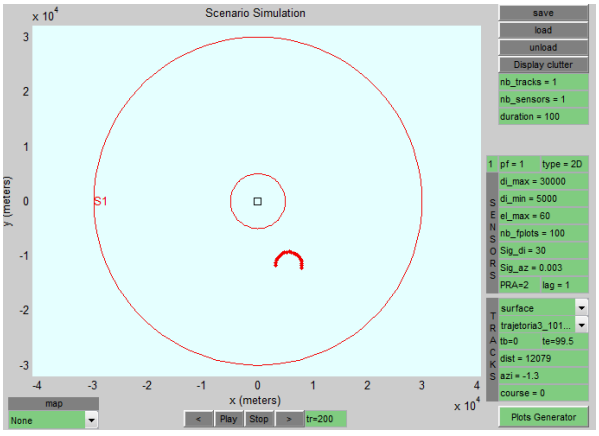


Figura 5.21: Trajetória 3 inserida no programa gerador de cenários

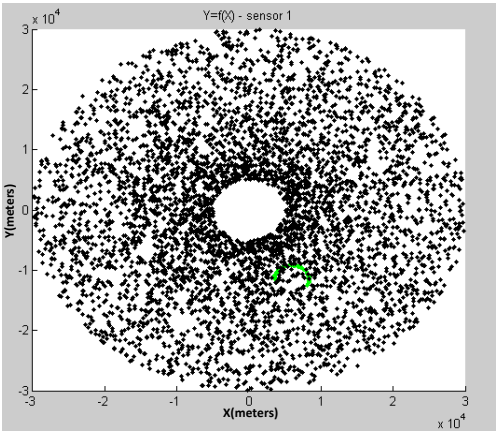


Figura 5.22: Cenário gerado 5

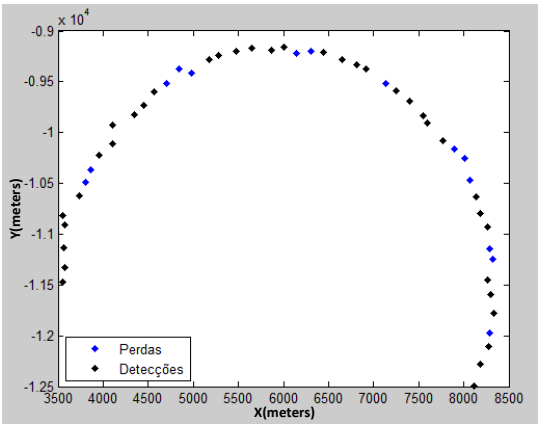


Figura 5.23: Trajetória resultante da geração do cenário 5

vi. Cenário 6:

O cenário 6 utiliza um alvo que descreve um movimento retilíneo com velocidade constante de 150 m/seg durante os primeiros 50 seg, seguido de um movimento curvilíneo com aceleração normal de 13 m/seg^2 (taxa de giro de 5 graus/seg) com duração de 36 seg, na seqüencia sua aceleração normal é alterada para -13 m/seg^2 (taxa de giro de -5 graus/seg) com duração de 36 seg e por fim esta aceleração é novamente alterada para 6 m/seg^2 (taxa de giro de 2.29 graus/seg) com duração de 40 seg, ver figura 5.24. Assim, o cenário tem uma duração de 162 seg, ou seja, 81 varreduras de antena.

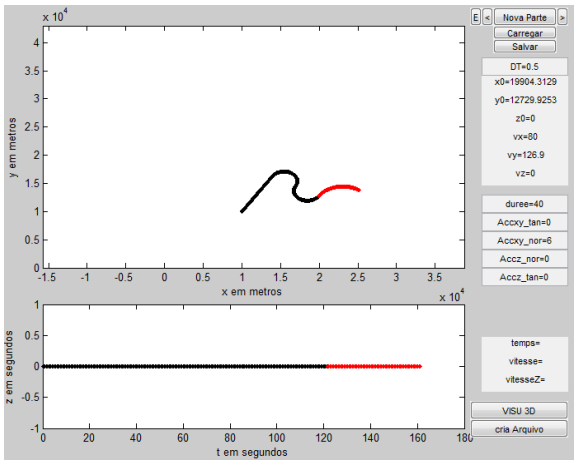


Figura 5.24: Trajetória 4

Os intervalos de tempo e as respectivas varreduras onde são introduzidas as perdas de detecção são apresentadas na tabela 5.4.

Tabela 5.4: Intervalos de tempo e varreduras com perda de detecção no cenário 6

Tempo (seg)	Varredura
$[12-18]$	7 a 8
$[28-36]$	15 a 17
$[46-52]$	24 a 25
$[58-62]$	30
$[70-78]$	36 a 38
$[82-88]$	42 a 43
$[92-96]$	47
$[100-108]$	51 a 53

A figura 5.25 mostra a trajetória inserida no programa gerador de cenários e o cenário gerado é apresentado na figura 5.26. A trajetória do alvo com adição do ruido de processo e de medição resultante deste cenário omitindo o clutter é apresentada na figura 5.27.

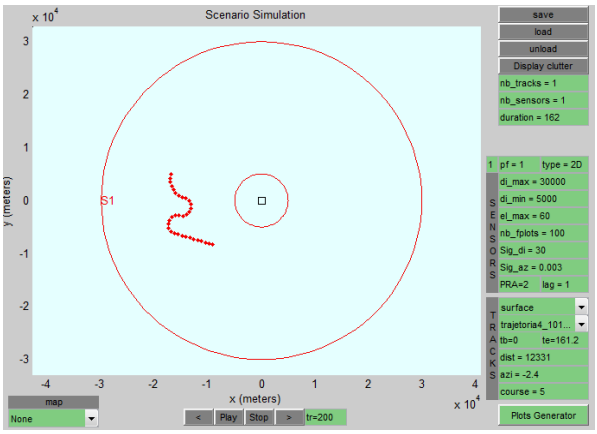


Figura 5.25: Trajetória 4 inserida no programa gerador de cenários

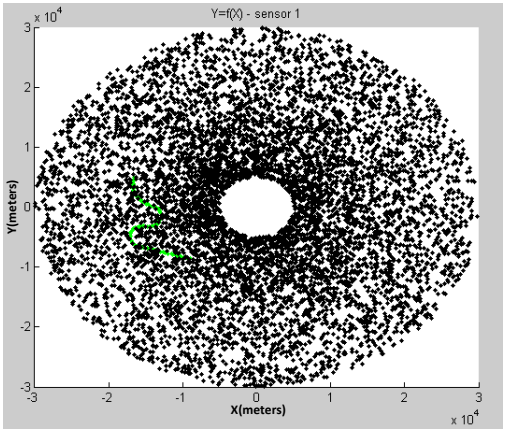


Figura 5.26: Cenário gerado 6

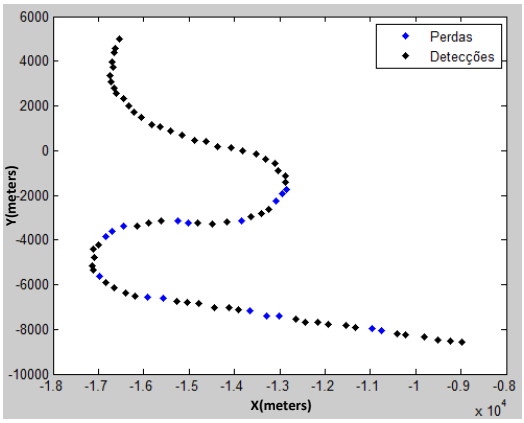


Figura 5.27: Trajetória resultante da geração do cenário 6

vii. Cenário 7:

O cenário 7 utiliza a trajetória descrita no cenário 6 (figura 5.24) considerando as mesmas perdas de detecção porém inserido-a num local diferente no programa gerador de cenário (ver figura 5.28).

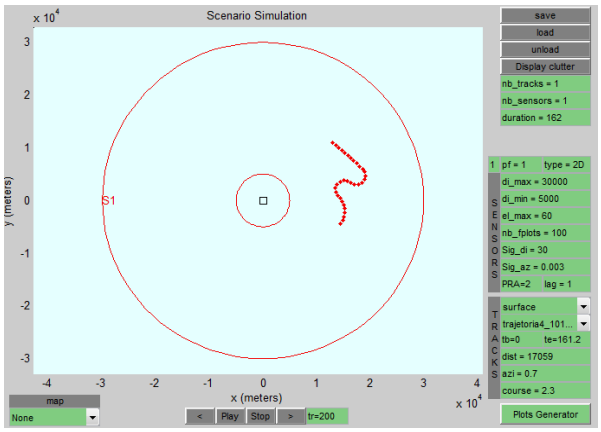


Figura 5.28: Trajetória 4 inserida num local diferente no programa gerador de cenários

O cenário gerado é apresentado na figura 5.29. A trajetória do alvo com adição do ruido de processo e de medição resultante deste cenário omitindo o clutter é apresentada na figura 5.30.

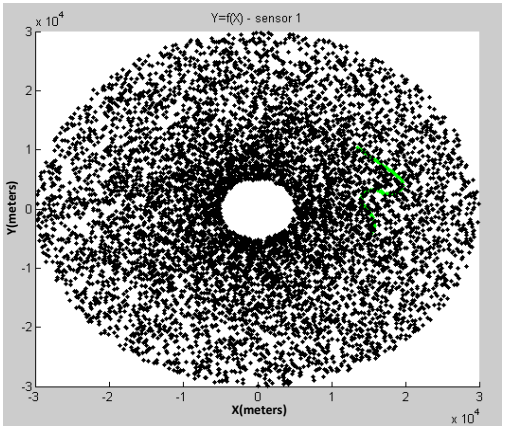


Figura 5.29: Cenário gerado 7

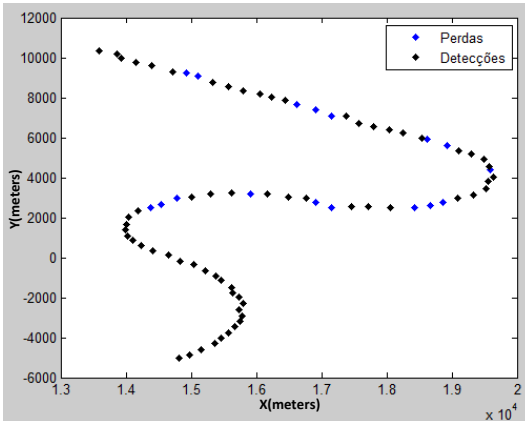


Figura 5.30: Trajetória resultante da geração do cenário 7

viii. Cenário 8:

O cenário 8 utiliza um alvo que descreve um movimento curvilíneo com velocidade inicial de 85 m/seg, aceleração tangencial de 3 m/seg^2 e aceleração normal de 5 m/s^2 com duração de 30 seg, ver figura 5.31. Assim, o cenário é composto por 15 varreduras de antena.

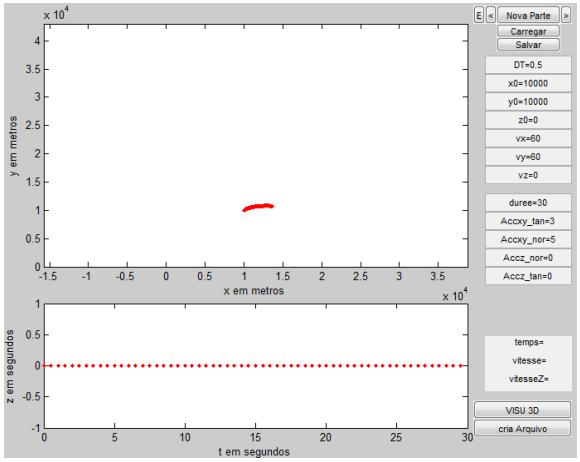


Figura 5.31: Trajetória 5

A figura 5.32 mostra a trajetória inserida no programa gerador de cenários e o cenário gerado é apresentado na figura 5.33. A trajetória do alvo com adição do ruido de processo e de medição resultante deste cenário omitindo o clutter é apresentada na figura 5.34.

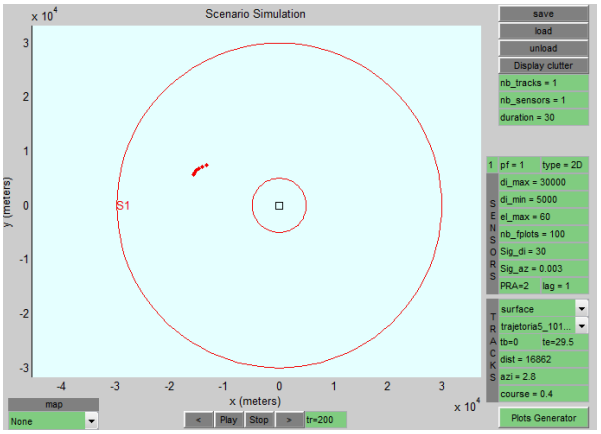


Figura 5.32: Trajetória 5 inserida no programa gerador de cenários

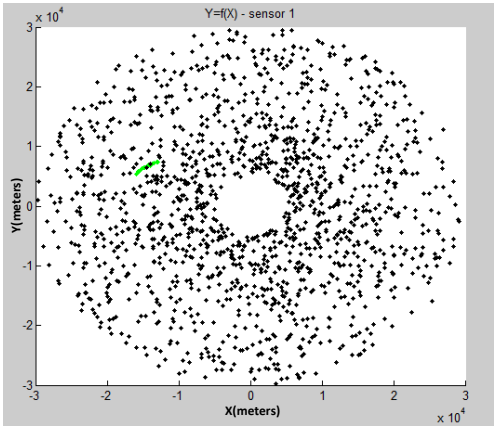


Figura 5.33: Cenário gerado 8

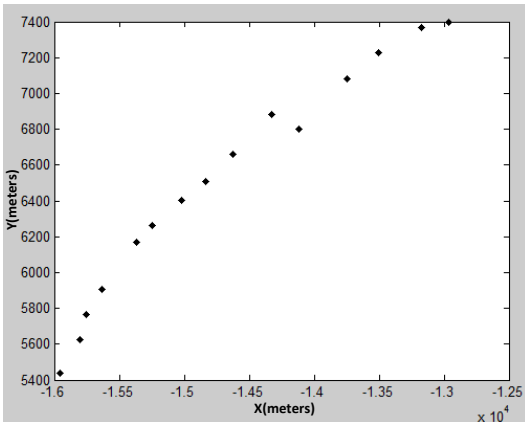


Figura 5.34: Trajetória resultante da geração do cenário 8

ix. Cenário 9:

O cenário 9 utiliza a trajetória descrita no cenário 8 (figura 5.31), mas é inserida num local diferente no programa gerador de cenários (figura 5.35). Além disso, são introduzidas perdas de detecção no intervalo de 12 a 18 segundos (varreduras de antena 7 e 8).

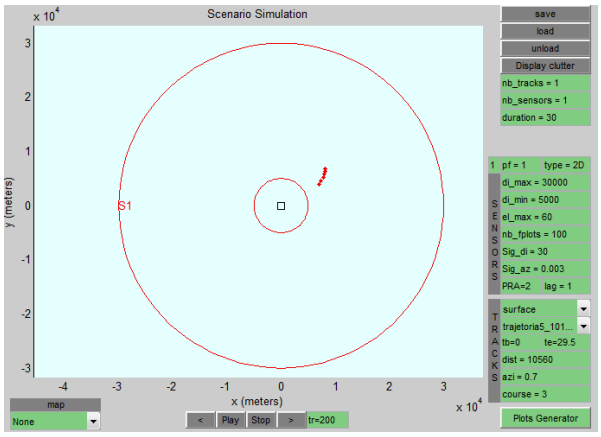


Figura 5.35: Trajetória 5 inserida num local diferente no programa gerador de cenários

O cenário gerado é apresentado na figura 5.36. A trajetória do alvo com adição do ruido de processo e de medição resultante deste cenário ignorando o clutter é apresentada na figura 5.37.

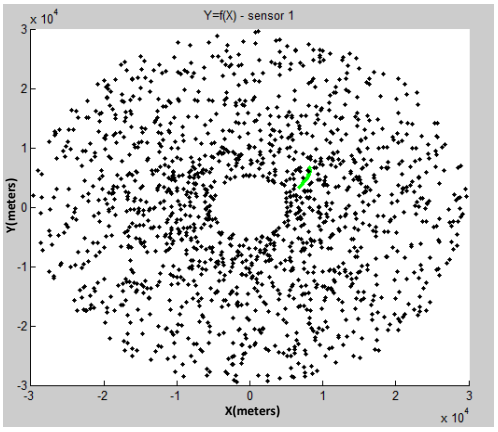


Figura 5.36: Cenário gerado 9

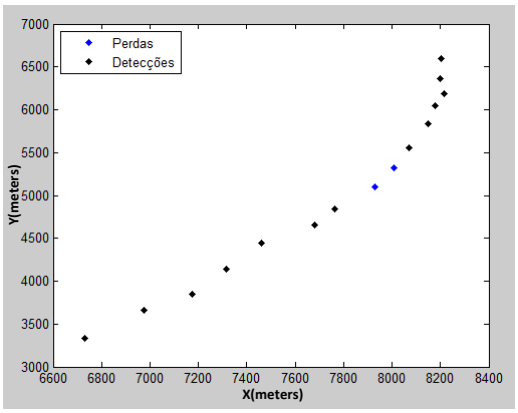


Figura 5.37: Trajetória resultante da geração do cenário 9

x. Cenário 10:

O cenário 10 utiliza um alvo que descreve um movimento retilíneo com velocidade inicial de 85 m/seg e aceleração tangencial de 4 m/seg^2 durante os primeiros 30 seg. Em seguida, a aceleração tangencial é alterada para -4 m/seg^2, permanecendo nesta aceleração durante os últimos $30seg$, ver figura 5.38. Assim, o cenário tem uma duração de 60 seg, ou seja, 30 varreduras de antena.

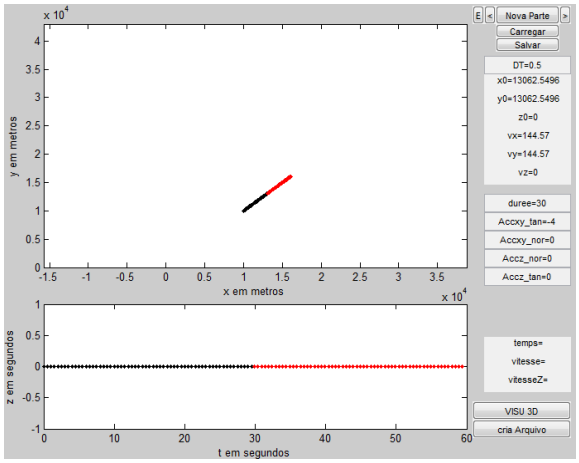


Figura 5.38: Trajetória 6

Os intervalos de tempo e as respectivas posições de varreduras onde são introduzidas as perdas de detecção são apresentadas na tabela 5.5.

Tabela 5.5: Intervalos de tempo e varreduras com perda de detecção no cenário 10

Tempo (seg)	Varredura
$[12-18]$	7 a 8
$[28-36]$	15 a 17
$[46-52]$	24 a 25

A figura 5.39 mostra a trajetória inserida no programa gerador de cenários e o cenário gerado é apresentado na figura 5.40. A trajetória do alvo com adição do ruido de processo e de medição resultante deste cenário ignorando o clutter é apresentada na figura 5.41.

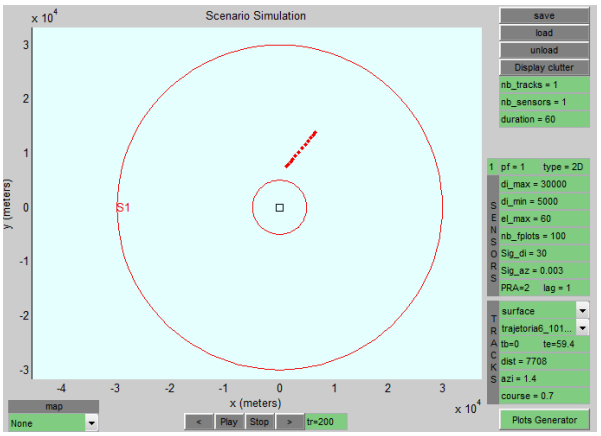


Figura 5.39: Trajetória 6 inserida no programa gerador de cenários

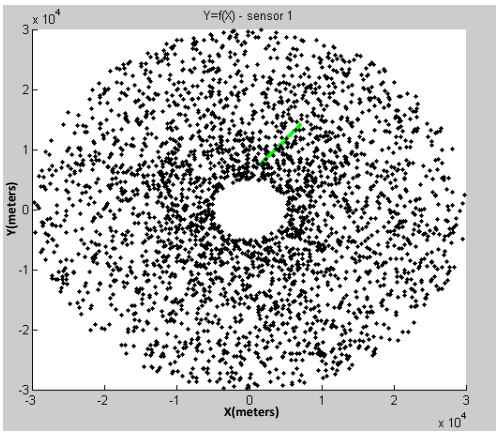


Figura 5.40: Cenário gerado 10

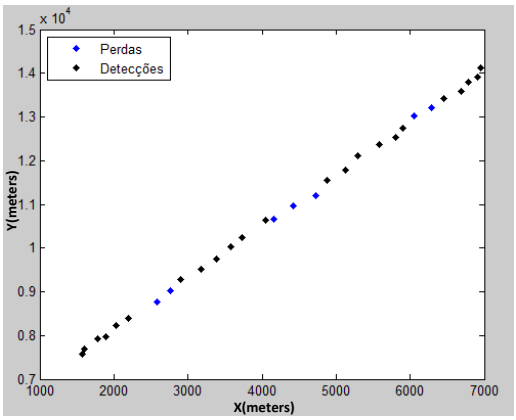


Figura 5.41: Trajetória resultante da geração do cenário 10

xi. Cenário 11:

O cenário 11 utiliza um alvo que descreve um movimento retilíneo com velocidade inicial de 85 m/seg e aceleração tangencial de 4 m/seg^2 com uma duração de 30 seg. Em seguida a aceleração tangencial é alterada para 0 m/seg^2 e adicionada uma aceleração normal de 10.33 m/seg^2 (taxa de giro de 2.89 graus/seg), descrevendo assim um movimento curvilíneo com uma duração de 50 seg. Finalmente esta aceleração normal é alterada para 12.88 m/seg^2 (taxa de giro de 3.6 graus/seg), permanecendo nesta aceleração por mais 40 seg, ver figura 5.42. Assim, este cenário tem uma duração de 120 seg, ou seja, 60 varreduras de antena.

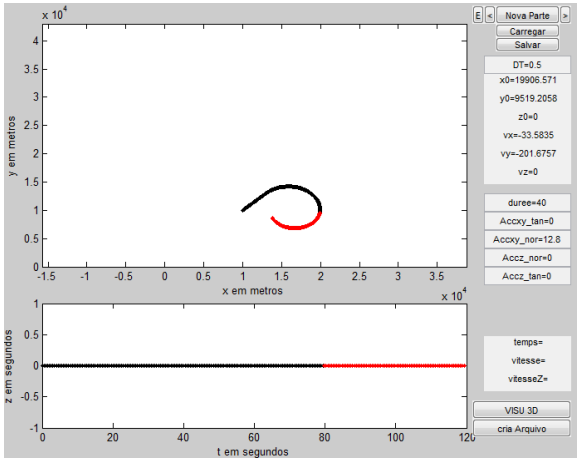


Figura 5.42: Trajetória 7

Os intervalos de tempo e as respectivas varreduras onde são introduzidas as perdas de detecção são apresentadas na tabela 5.6.

Tabela 5.6: Intervalos de tempo e varreduras com perda de detecção no cenário 11

Tempo (seg)	Varredura
$[12-18]$	7 a 8
$[28-36]$	15 a 17
$[46-52]$	24 a 25
$[58-62]$	30
$[70-78]$	36 a 38
$[82-88]$	42 a 43
$[92-96]$	47
$[100-108]$	51 a 53

A figura 5.43 mostra a trajetória inserida no programa gerador de cenários e o cenário gerado é apresentado na figura 5.44. A trajetória do alvo com adição do ruido de processo e de medição resultante deste cenário omitindo o clutter é apresentada na figura 5.45.

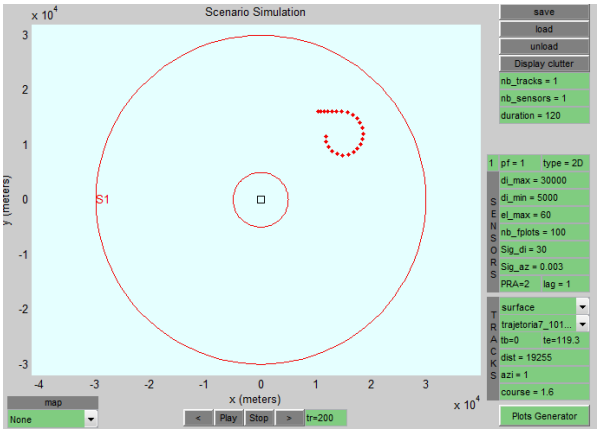


Figura 5.43: Trajetória 7 inserida no programa gerador de cenários

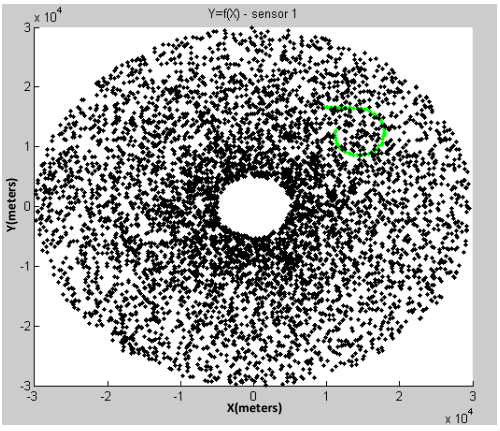


Figura 5.44: Cenário gerado 11

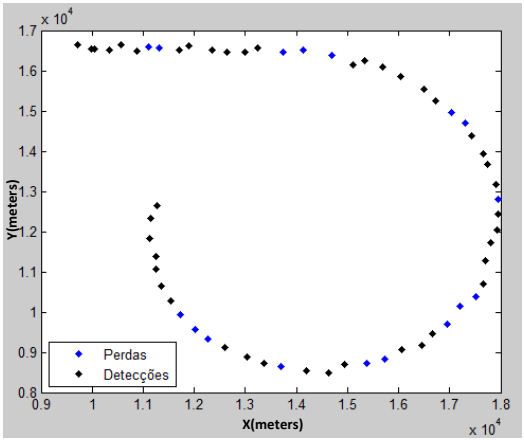


Figura 5.45: Trajetória resultante da geração do cenário 11

xii. Cenário 12:

O cenário 12 utiliza um alvo que descreve um movimento retilíneo com velocidade constante de 141.4 m/seg durante os primeiros 40 seg, em seguida é adicionada uma aceleração normal de -10 m/seg^2 (taxa de giro de -4.05 graus/seg), descrevendo assim um movimento curvilíneo com uma duração de 50 seg. Finalmente esta aceleração normal é alterada para -5 m/seg^2 (taxa de giro de -2.03 graus/seg), permanecendo nesta aceleração durante 50 seg, ver figura 5.46. Assim, o cenário tem uma duração de 140 seg, ou seja, 70 varreduras de antena.

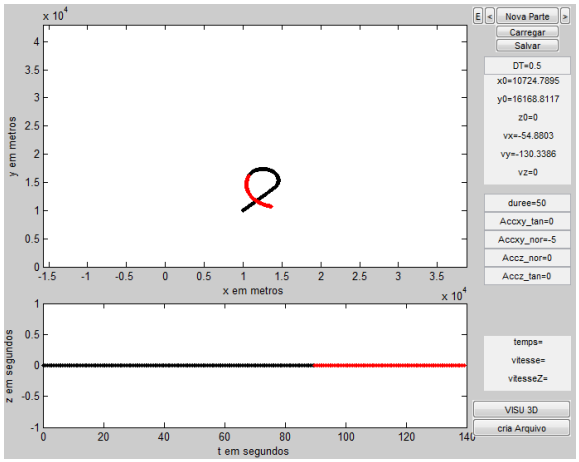


Figura 5.46: Trajetória 8

Os intervalos de tempo e as respectivas varreduras onde são introduzidas as perdas de detecção são apresentadas na tabela 5.7.

Tabela 5.7: Intervalos de tempo e varreduras com perda de detecção no cenário 12

Tempo (seg)	Varredura
$[12-18]$	7 a 8
$[28-36]$	15 a 17
$[46-52]$	24 a 25
$[58-62]$	30
$[70-78]$	36 a 38
$[82-88]$	42 a 43
$[92-96]$	47
$[100-108]$	51 a 53

A figura 5.47 mostra a trajetória inserida no programa gerador de cenários e o cenário gerado é apresentado na figura 5.48. A trajetória do alvo com adição do ruido de processo e de medição resultante deste cenário omitindo o clutter é apresentada na figura 5.49.

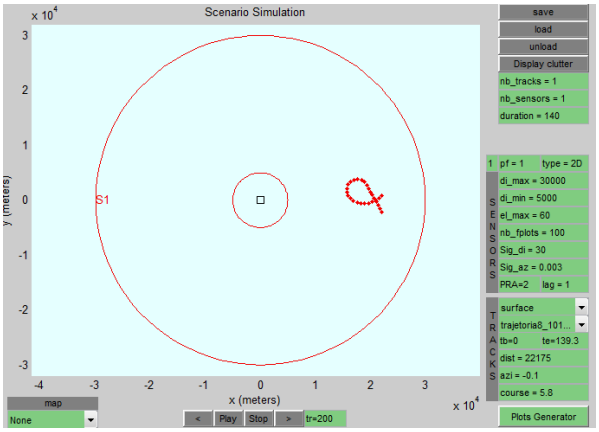


Figura 5.47: Trajetória 8 inserida no programa gerador de cenários

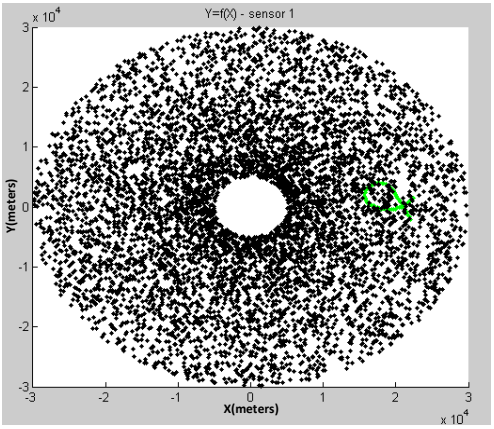


Figura 5.48: Cenário gerado 12

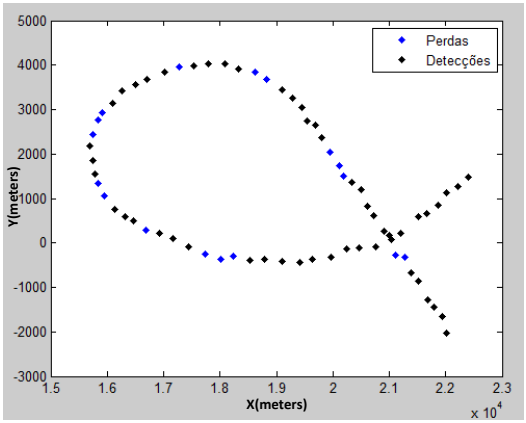


Figura 5.49: Trajetória resultante da geração do cenário 12

xiii. Cenário 13:

O cenário 13 utiliza a trajetória descrita no cenário 12 (figura 5.46) considerando as mesmas perdas de detecção, porém inserindo-a num local diferente no programa gerador de cenário (ver figura 5.50).

O cenário gerado é apresentado na figura 5.51. A trajetória do alvo com adição do ruido de processo e de medição resultante deste cenário omitindo o clutter é apresentada na figura 5.52.

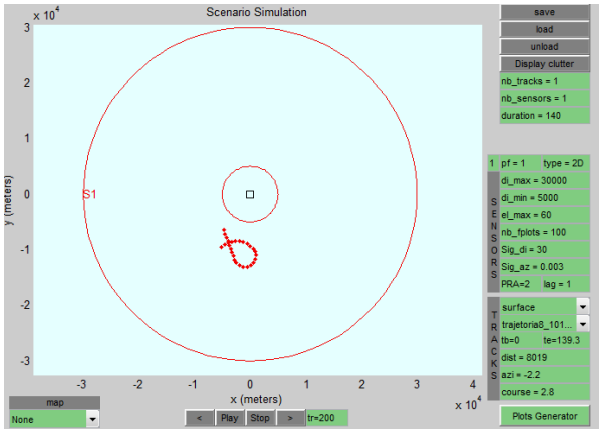


Figura 5.50: Trajetória 8 inserida num local diferente no programa gerador de cenários

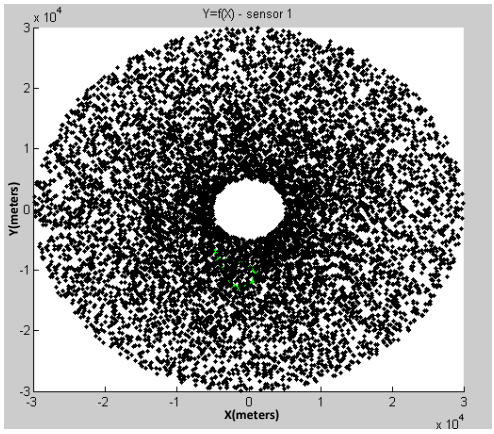


Figura 5.51: Cenário gerado 13

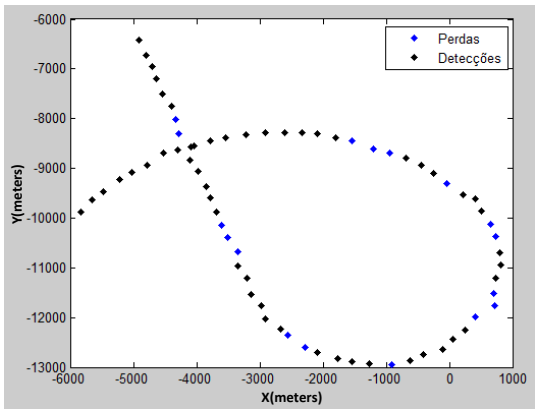


Figura 5.52: Trajetória resultante da geração do cenário 13

xiv. Cenário 14:

O cenário 14 utiliza um alvo que descreve um movimento curvilíneo com velocidade constante de 108.17 m/seg e uma aceleração normal de 3.48 m/seg^2 (taxa de giro de 1.8 graus/seg) durante 200 seg, ver figura 5.53. Assim, o cenário é composto por 100 varreduras de antena.

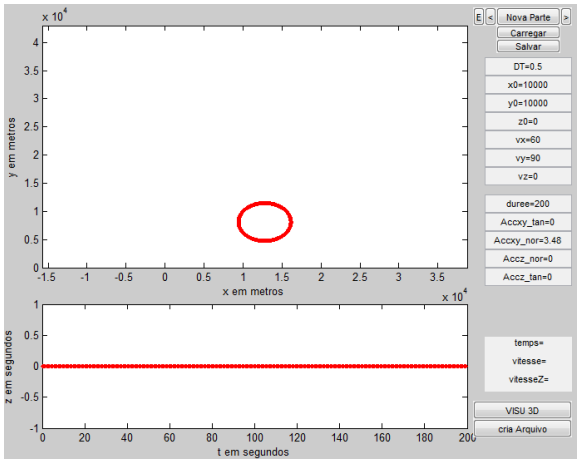


Figura 5.53: Trajetória 9

Os intervalos de tempo e as respectivas varreduras onde são introduzidas as perdas de detecção são apresentadas na tabela 5.8.

Tabela 5.8: Intervalos de tempo e varreduras com perda de detecção no cenário 14

Tempo (seg)	Varredura
$[12-18]$	7 a 8
$[28-36]$	15 a 17
$[46-52]$	24 a 25
$[58-62]$	30
$[70-78]$	36 a 38
$[82-88]$	42 a 43
$[92-96]$	47
$[100-108]$	51 a 53
$[124-132]$	63 a 65
$[140-146]$	71 a 72
$[168-172]$	85

A figura 5.54 mostra a trajetória inserida no programa gerador de cenários e o cenário gerado é apresentado na figura 5.55. A trajetória do alvo com adição do ruido de processo e de medição resultante deste cenário omitindo clutter é apresentada na figura 5.56.

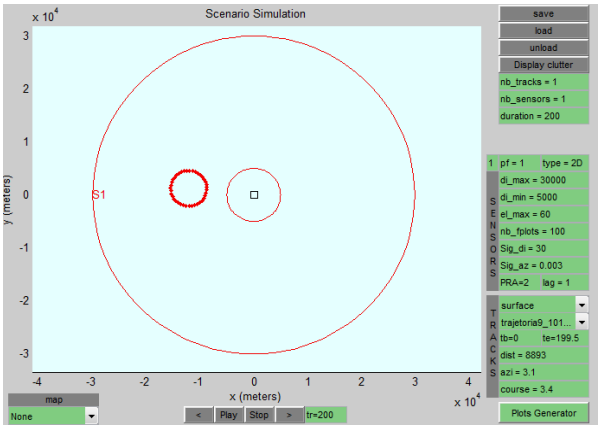


Figura 5.54: Trajetória 9 inserida no programa gerador de cenários

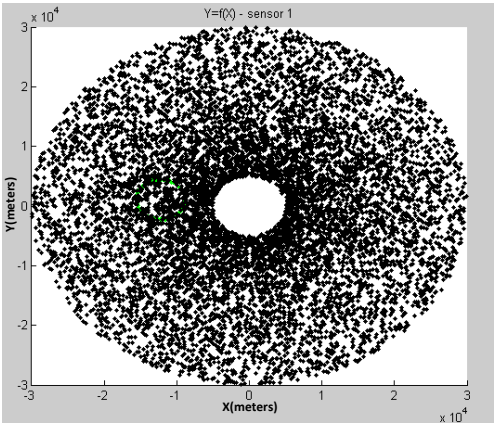


Figura 5.55: Cenário gerado 14

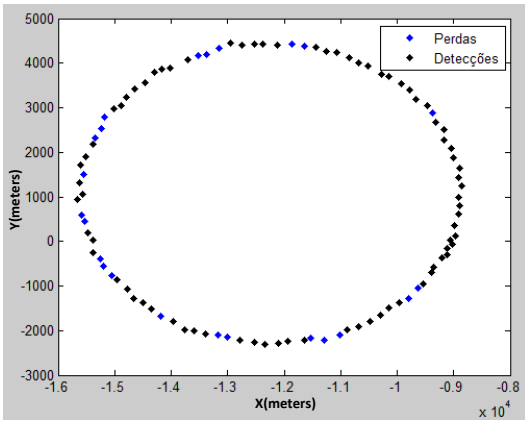


Figura 5.56: Trajetória resultante da geração do cenário 14

xv. Cenário 15:

O cenário 15 utiliza um alvo que descreve um movimento curvilíneo com velocidade constante de 127.28 m/seg e uma aceleração normal de -6.6 m/seg^2 (taxa de giro de -3 graus/seg) durante 120 seg, ver figura 5.57. Assim, o cenário é composto por 60 varreduras de antena.

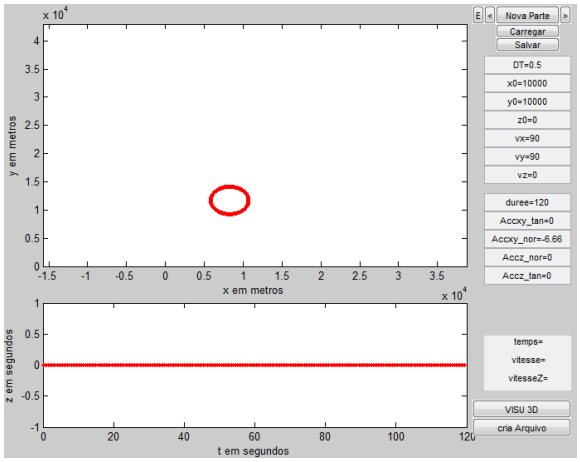


Figura 5.57: Trajetória 10

Os intervalos de tempo e as respectivas varreduras onde são introduzidas as perdas de detecção são apresentadas na tabela 5.9.

Tabela 5.9: Intervalos de tempo e varreduras com perda de detecção no cenário 15

Tempo (seg)	Varredura
$[12-18]$	7 a 8
$[28-36]$	15 a 17
$[46-52]$	24 a 25
$[58-62]$	30
$[70-78]$	36 a 38
$[82-88]$	42 a 43
$[92-96]$	47
$[100-108]$	51 a 53

A figura 5.58 mostra a trajetória inserida no programa gerador de cenários e o cenário gerado é apresentado na figura 5.59. A trajetória do alvo com adição do ruido de processo e de medição resultante deste cenário omitindo o clutter é apresentada na figura 5.60.

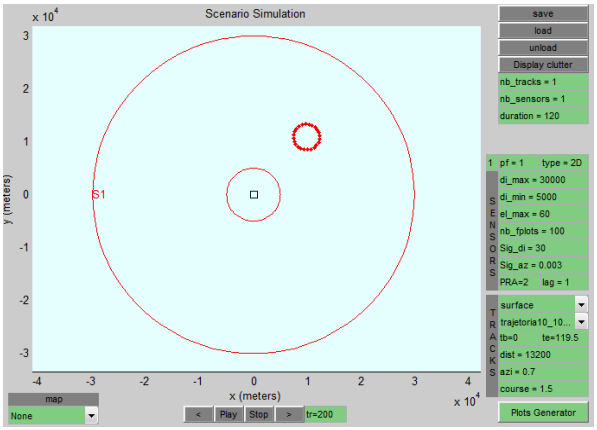


Figura 5.58: Trajetória 10 inserida no programa gerador de cenários

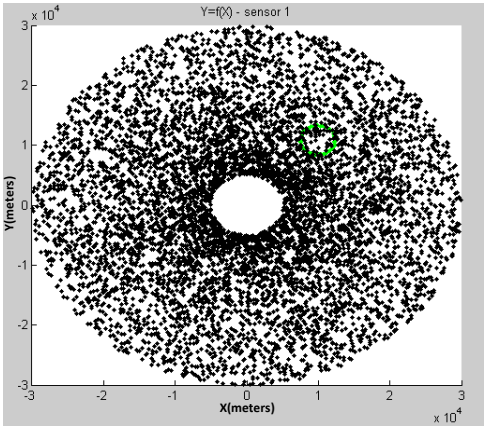


Figura 5.59: Cenário gerado 15

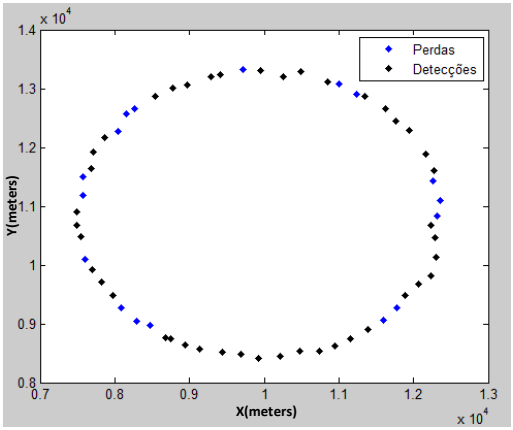


Figura 5.60: Trajetória resultante da geração do cenário 15

xvi. Cenário 16:

O cenário 16 utilizou um alvo que descreve um movimento retilíneo com velocidade constante de 113.14 m/seg durante os primeiros 100 seg, em seguida é adicionada uma aceleração normal de -5 m/seg^2 (taxa de giro de 2.53 graus/seg), permanecendo nesta aceleração por mais 50 seg, ver figura 5.61. Assim, o cenário é composto por 75 varreduras de antena.

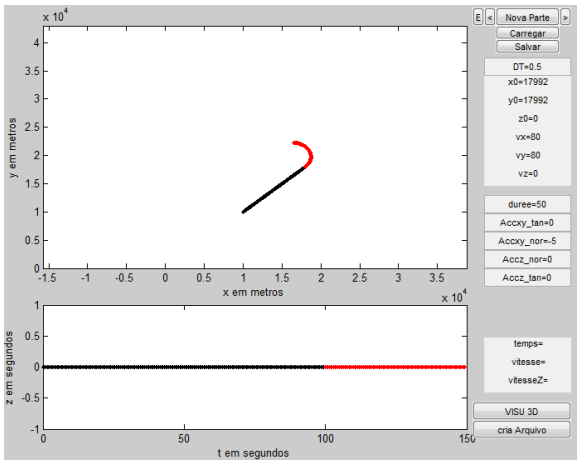


Figura 5.61: Trajetória 11

Os intervalos de tempo e as respectivas varreduras onde são introduzidas as perdas de detecção são apresentadas na tabela 5.10.

Tabela 5.10: Intervalos de tempo e varreduras com perda de detecção no cenário 16

Tempo (seg)	Varredura
$[12-18]$	7 a 8
$[28-36]$	15 a 17
$[46-52]$	24 a 25
$[58-62]$	30
$[70-78]$	36 a 38
$[82-88]$	42 a 43
$[92-96]$	47
$[100-108]$	51 a 53

A figura 5.62 mostra a trajetória inserida no programa gerador de cenários e o cenário gerado é apresentado na figura 5.63. A trajetória do alvo com adição do ruido de processo e de medição resultante deste cenário omitindo o clutter é apresentada na figura 5.64.

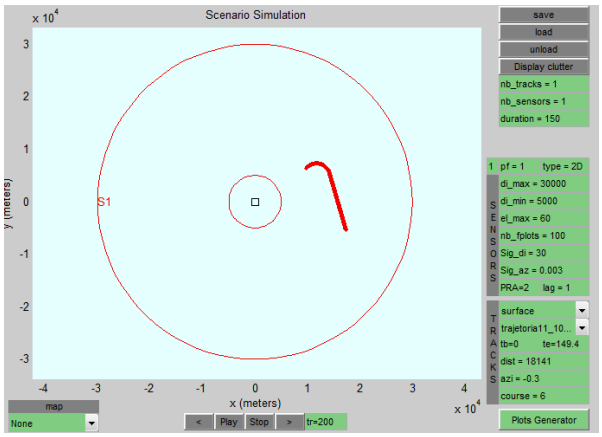


Figura 5.62: Trajetória 11 inserida no programa gerador de cenários

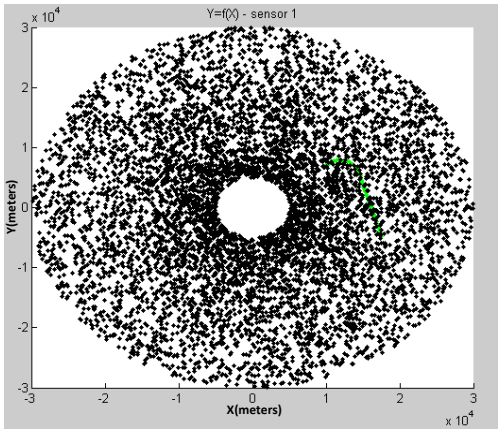


Figura 5.63: Cenário gerado 16

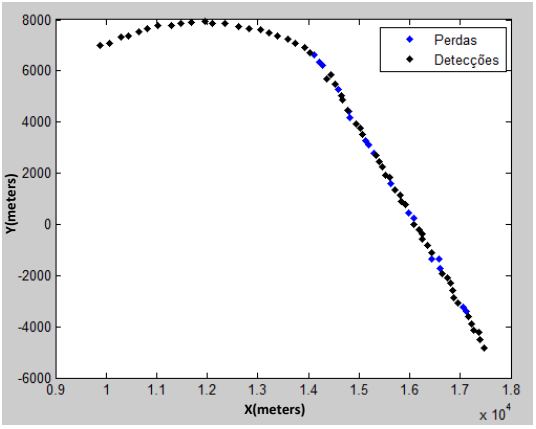


Figura 5.64: Trajetória resultante da geração do cenário 16

xvii. Cenário 17:

O cenário 17 utiliza um alvo que descreve um movimento curvilíneo com velocidade constante de 160 m/seg e aceleração normal de 11.2 m/seg^2 (taxa de giro de 4 graus/seg) durante 40 seg, em seguida, esta aceleração é alterada para -11.2 m/seg^2 (taxa de giro de -4 graus/seg) durante 40 seg e finalmente alterada novamente para 13 m/seg^2 (taxa de giro de 4.7 graus/seg), permanecendo durante $40seg$, ver figura 5.65. Assim, o cenário é composto por 60 varreduras de antena.

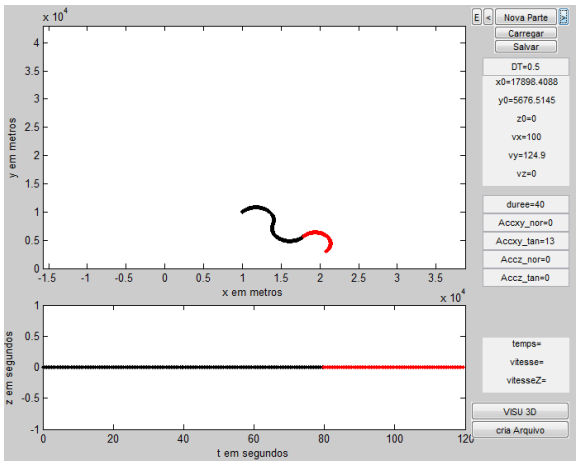


Figura 5.65: Trajetória 12

Os intervalos de tempo e as respectivas varreduras onde são introduzidas as perdas de detecção são apresentadas na tabela 5.11.

Tabela 5.11: Intervalos de tempo e varreduras com perda de detecção no cenário 17

Tempo (seg)	Varredura
$[12-18]$	7 a 8
$[28-36]$	15 a 17
$[46-52]$	24 a 25
$[58-62]$	30
$[70-78]$	36 a 38
$[82-88]$	42 a 43
$[92-96]$	47
$[100-108]$	51 a 53

A figura 5.66 mostra a trajetória inserida no programa gerador de cenários e o cenário gerado é apresentado na figura 5.67. A trajetória do alvo com adição do ruido de processo e de medição resultante deste cenário omitindo o clutter é apresentada na figura 5.68.

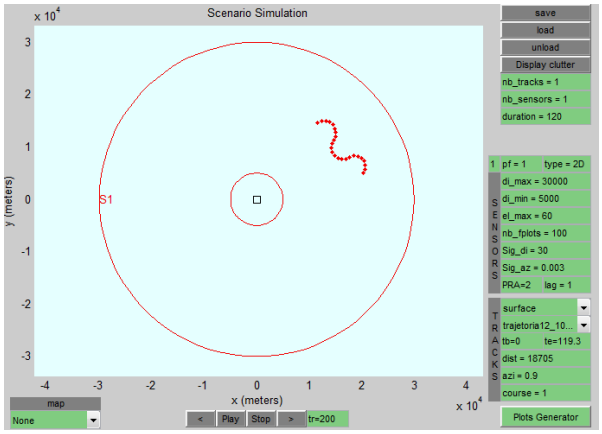


Figura 5.66: Trajetória 12 inserida no programa gerador de cenários

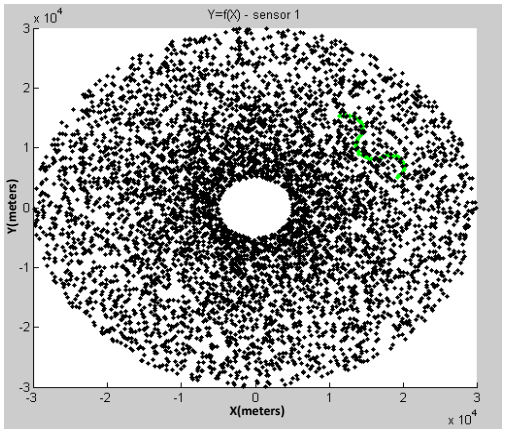


Figura 5.67: Cenário gerado 17

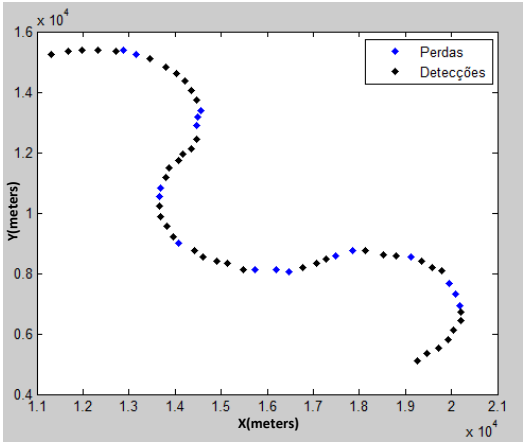


Figura 5.68: Trajetória resultante da geração do cenário 17

5.2.2 Descrição dos cenários gerados para a validação do método proposto

Para a criação dos cenários utilizados para a validação do método proposto são geradas as seguintes trajetórias.

Trajetória 1: representa um alvo que descreve um movimento retilíneo com velocidade constante de 70.7 m/seg com duração de 50 seg, seguidamente é adicionada uma aceleração normal de 4.94 m/seg^2 (taxa de giro de 4 graus/seg), permanecendo nesta aceleração por 50 seg. Transcorrido este tempo, a aceleração é alterada para -4.94 m/seg^2 (taxa de giro de -4 graus/seg). Finalmente, a aceleração normal é novamente alterada para 0 m/seg e, imediatamente, é adicionada uma aceleração tangencial de 3 m/seg, permanecendo assim por mais 20 seg, ver figura 5.69.

Trajetória 2: representa um alvo que descreve um movimento retilíneo uniforme com velocidade de 141.4 m/seg com uma duração de 150 seg, ver figura 5.70.

Trajetória 3: representa um alvo que descreve um movimento curvilíneo com velocidade constante de 196 m/seg e aceleração normal de 10.3 m/seg^2 (taxa de giro de 3 graus/seg) durante 60 seg, ver figura 5.71.

Trajetória 4: representa um alvo que descreve um movimento retilíneo uniforme com velocidade de 103 m/seg com uma duração de 150 seg, ver figura 5.72.

Trajetória 5: representa um alvo que descreve um movimento retilíneo com velocidade inicial de 85 m/seg e aceleração tangencial de 5 m/seg^2 durante os primeiros 20 seg. Em seguida, esta aceleração é alterada para 0 m/seg^2 e, imediatamente, é adicionada uma aceleração normal de 6.44 m/seg^2 (taxa de giro de 2 graus/seg), que permanece por um período de 45 seg. Finalmente, o alvo descreve um movimento retilíneo uniforme durante os últimos 50 seg, ver figura 5.73.

Trajetória 6: representa um alvo que descreve um movimento curvilíneo com velocidade constante de 70.7 m/seg e aceleração normal de 1.24 m/seg^2 (taxa de giro de 1 graus/seg) durante os primeiros 90 seg, em seguida, esta aceleração é alterada para -1.24 m/seg^2 (taxa de giro de -1 graus/seg), permanecendo por um período de 90 seg, ver figura 5.74.

Trajetória 7: representa um alvo que descreve um movimento retilíneo com velocidade inicial de 85 m/seg e aceleração tangencial de 5 m/seg^2 durante os primeiros 20 seg, em seguida, esta aceleração é alterada para 0 m/seg^2 e, imediatamente, é adicionada uma aceleração normal de -6.44 m/seg^2 (taxa de giro de -2 graus/seg), que permanece por um período de 45 seg. Finalmente, o alvo descreve um movimento retilíneo uniforme durante os últimos 50 seg, ver figura 5.75.

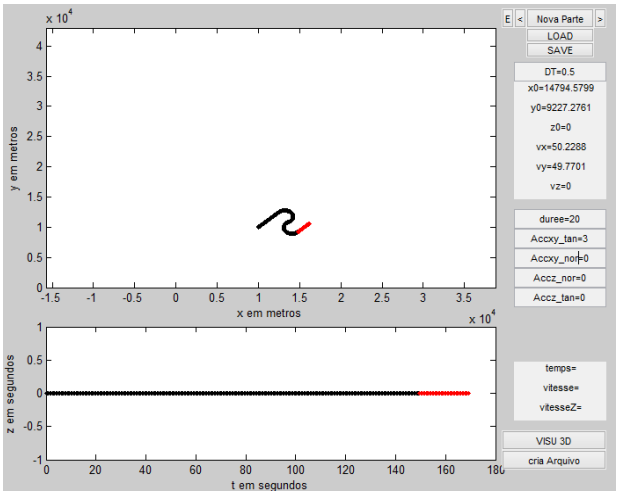


Figura 5.69: Trajetória 1

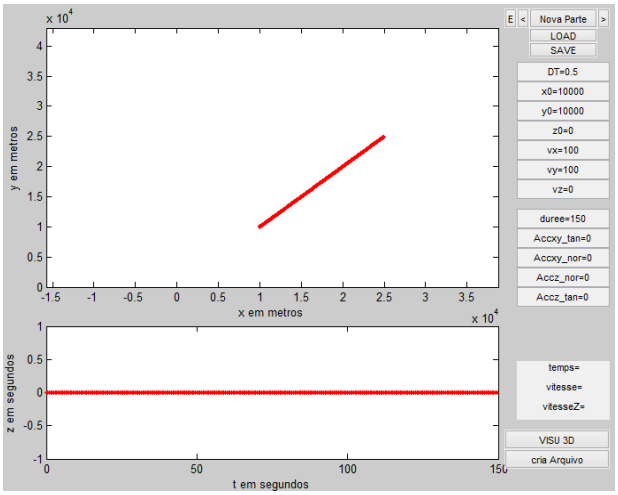


Figura 5.70: Trajetória 2

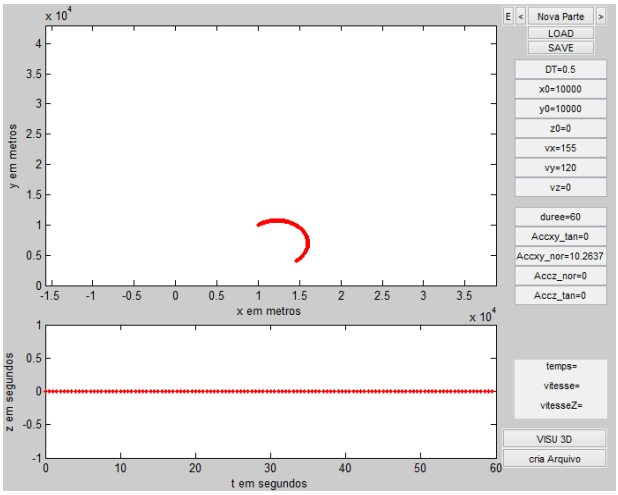


Figura 5.71: Trajetória 3

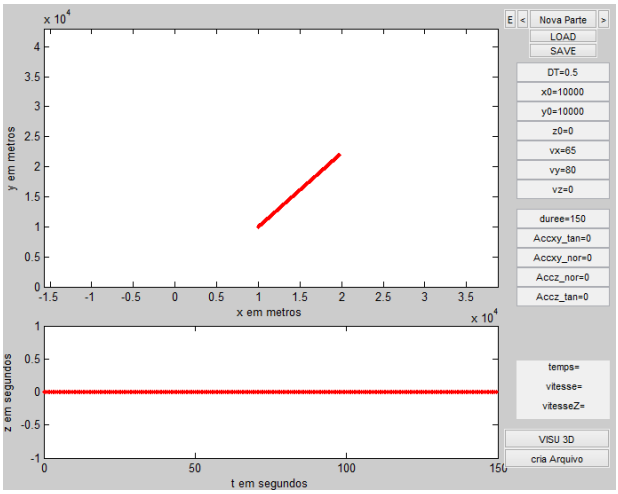


Figura 5.72: Trajetória 4

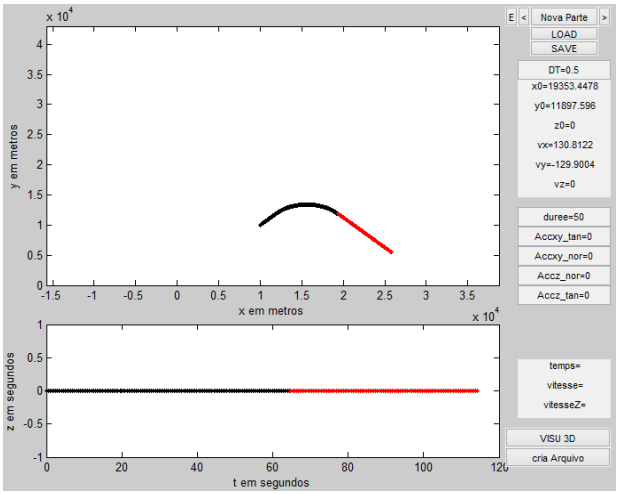


Figura 5.73: Trajetória 5

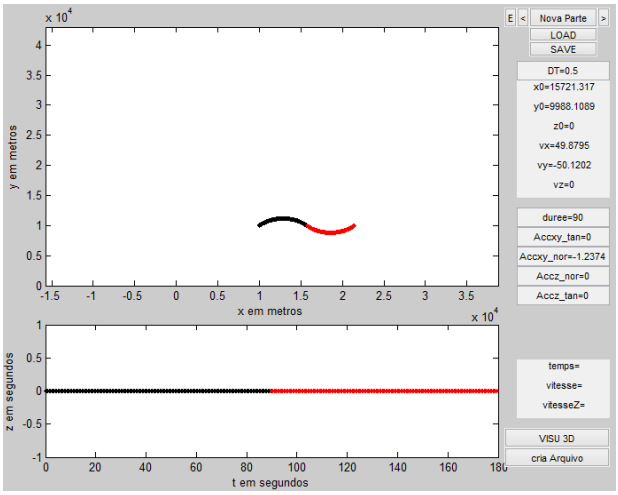


Figura 5.74: Trajetória 6

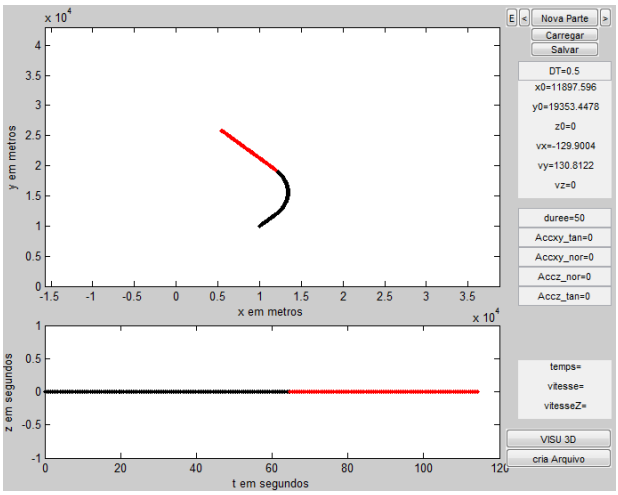


Figura 5.75: Trajetória 7

A duração dos cenários aqui gerados é estabelecida a ser 200 seg, independente da duração das trajetórias contidas em cada um dos cenários.

i. Cenário 1:

O cenário 1 é composto pelas trajetórias 1 (figura 5.69), 2 (figura 5.70) e 3 (figura 5.71). A trajetória 2 é utilizada duas vezes, inserida em locais diferentes no programa gerador de cenário. A figura 5.76 mostra as trajetórias inseridas no programa gerador de cenários e o cenário gerado é apresentado na figura 5.77.

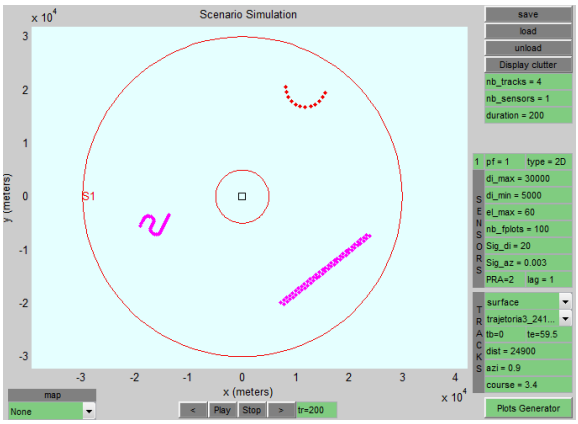


Figura 5.76: Trajetórias inseridas no programa gerador de cenários

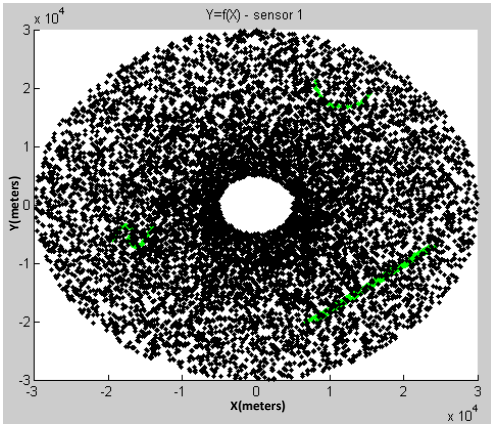


Figura 5.77: Cenário gerado múltiplos alvos 1

ii. Cenário 2:

O cenário 2 é composto pelas trajetórias 4 (figura 5.72), 5 (figura 5.73), 6 (figura 5.74) e 7 (figura 5.75). A figura 5.78 mostra as trajetórias inseridas no programa gerador de cenários e o cenário gerado é apresentado na figura 5.79.

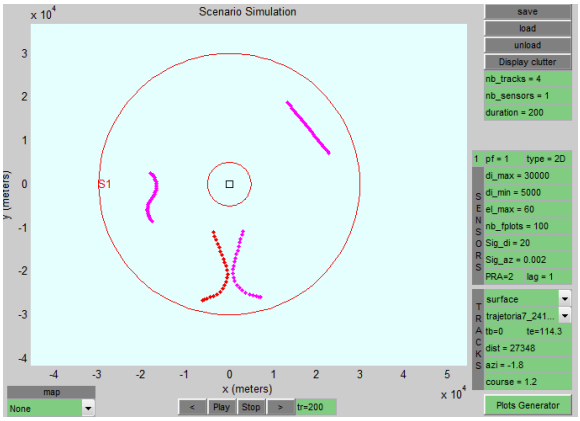


Figura 5.78: Trajetórias inseridas no programa gerador de cenários

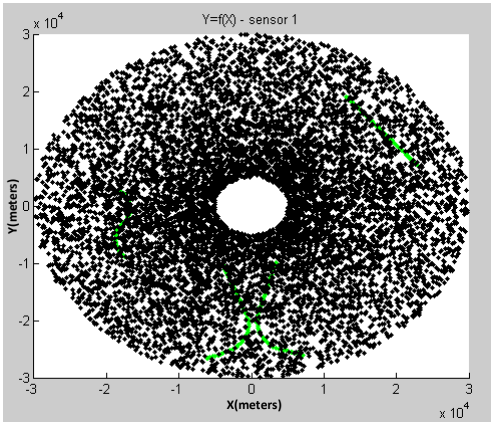


Figura 5.79: Cenário gerado múltiplos alvos 2

iii. Cenário 3:

O cenário 3 é composto pelas mesmas trajetórias utilizadas no cenário 2, inseridas no programa gerador de cenários como apresentado na figura 5.78. Nestas trajetórias são introduzidas perdas de detecção em intervalos de tempo apresentados na tabela 5.12. O cenário gerado é apresentado na figura 5.80.

Tabela 5.12: Trajetórias, intervalos de tempo e varreduras com perda de detecção no cenário 3

Trajetória	Tempo (seg)	Varredura
5	$[46-52]$	24 a 25
	$[78-86]$	40 a 42
6	$[28-32]$	15
	$[48-56]$	25 a 27
	$[168-174]$	85 a 86
7	$[14-18]$	8
	$[98-104]$	50 a 51

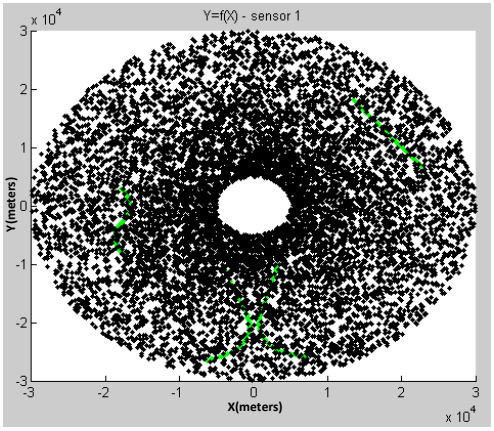


Figura 5.80: Cenário gerado múltiplos alvos 3

iv. Cenário 4:

O cenário 4 é composto pelas trajetórias 1 (figura 5.69), 3 (figura 5.71) e 5 (figura 5.73), sendo que a trajetória 3 é utilizada duas vezes, inserida em locais diferentes, no programa gerador de cenário. A figura 5.81 mostra as trajetórias inseridas no programa gerador de cenários e o cenário gerado é apresentado na figura 5.82.

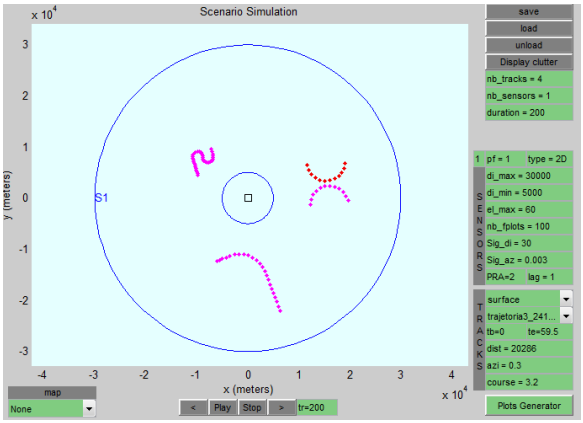


Figura 5.81: Trajetórias inseridas no programa gerador de cenários

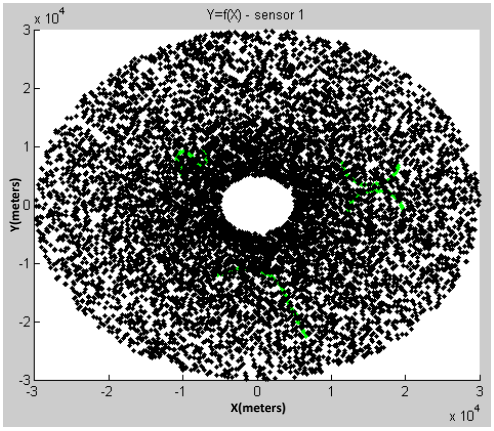


Figura 5.82: Cenário gerado múltiplos alvos 4

v. Cenário 5:

O cenário 5 é composto pelas mesmas trajetórias utilizadas no cenário 4, mas são introduzidas perdas de detecção nas trajetórias em intervalos de tempo apresentados na tabela 5.13. Dado que neste cenário a trajetória 3 é utilizada em duas ocasiões, renomeamos por 3_a e 3_b como forma de diferenciá-la, sendo que a trajetória 3_a é representada pela cor vermelho na figura 5.81. O cenário gerado é apresentado na figura5.83.

Tabela 5.13: Trajetórias, intervalos de tempo e varreduras com perda de detecção no cenário 5

Trajetória	Tempo (seg)	Varredura
1	$[18-22]$	10
	$[38-44]$	20 a 21
	$[92-96]$	47
	$[114-122]$	58 a 60
3_a	$[20-26]$	11 a 12
	$[46-50]$	24
3_b	$[16-20]$	9
	$[30-36]$	16 a 17

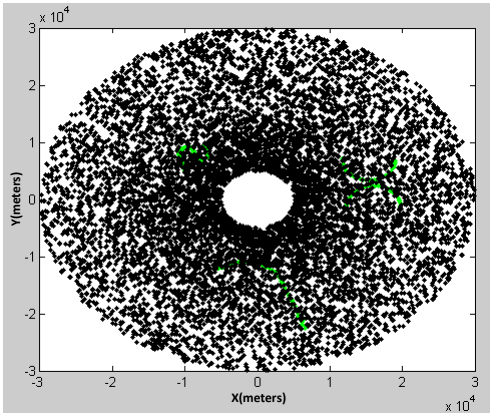


Figura 5.83: Cenário gerado múltiplos alvos 5

vi. Cenário 6:

O cenário 6 é composto pelas trajetórias 1 (figura 5.69), 2 (figura 5.70), 3 (figura 5.71) e 5 (figura 5.73), sendo que a trajetória 3 é utilizada três vezes, inserida em locais diferentes no programa gerador de cenário. A figura 5.84 mostra as trajetórias inseridas no programa gerador de cenários e o cenário gerado é apresentado na figura 5.85.

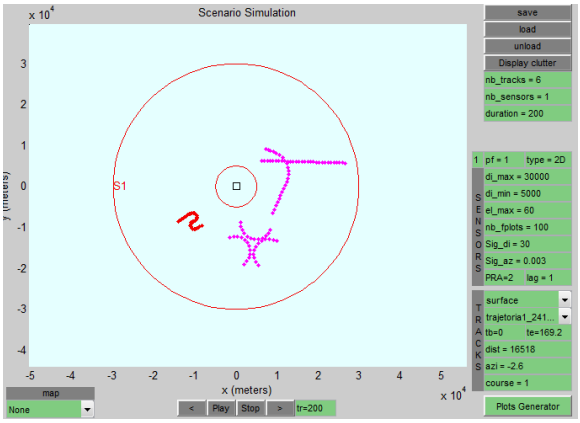


Figura 5.84: Trajetórias inseridas no programa gerador de cenários

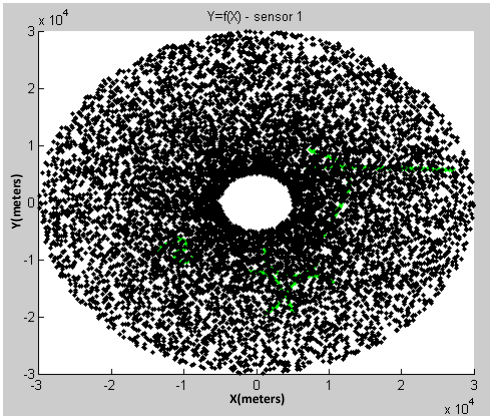


Figura 5.85: Cenário gerado múltiplos alvos 6

5.3 Criação dos acompanhamentos preliminares

De forma análoga ao processo descrito na seção 2.1.1, é criada uma janela circular entorno de qualquer medida não associada a um acompanhamento existente. Esta janela terá limites estabelecidos por

$$D.min = (70\frac{m}{s} * 2s) - (3 * 30m) = 50\ m$$
$$D.max = (200\frac{m}{s} * 2s) + (3 * 30m) = 490\ m \quad .$$

Se na varredura seguinte aparece uma medida dentro desta janela, é criado um novo acompanhamento. Devido à densidade de clutter, é possível que dentro de uma mesma janela apareçam mais do que uma medida, criando acompanhamentos com medidas em comum, os quais, no decorrer do tempo, muito provávelmente serão excluídos, permanecendo unicamente aqueles que representam um objeto de interesse. Os acompanhamentos criados são armazenados e atualizados nas seguintes varreduras por meio da rede neural.

5.4 Rede neural

A rede neural é a abordagem principal do método proposto, já que dela depende a correta atualização de cada um dos acompanhamentos. As redes neurais são famosas e amplamente utilizadas por serem uma ferramenta poderosa de apoio à tomada de decisão em diversas aplicações como, por exemplo, classificação. No entanto, essas técnicas de aprendizagem são extremamente sensíveis à diferentes fatores. Pelo fato de o treinamento da rede ser baseado em exemplos, independentemente da eficiência do algoritmo de aprendizagem, o seu desempenho vai depender da relevância das variáveis de entrada e da qualidade dos dados utilizados. Assim,

definir adequadamente as variáveis de entrada e fazer um pre-processamento dos dados se tornam tarefas fundamentais. Para o problema de rastreamento de alvos marítimos, foco desta dissertação, onde se tem informações provenientes do sensor Radar, e partindo do pressuposto de que o movimento do alvo apresenta um padrão coerente, ou seja, respeita limites de cinemática pré-estabelecidos (aceleração e velocidades máximas e mínimas), opta-se por utilizar informações de cinemática como critério para identificar se um contato deve ser considerado para atualizar o acompanhamento de um alvo existente. Desta forma, tendo em conta os diferentes tipos de movimento que um alvo marítimo pode experimentar, foram selecionadas como variáveis de entrada,

$$\Delta V(t_k) = V(t_k) - V(t_{k-1}) \tag{5.4}$$

$$\Delta\alpha(t_k) = \frac{\alpha(t_k)}{\Delta T} \tag{5.5}$$

$$\Delta T = t_k - t_{k-1} \tag{5.6}$$

onde $V(t_k)$ é a velocidade necessária para percorrer a distancia entre dois contatos, definida como $V(t_k) = \frac{d(t_k)}{\Delta T}$, $d(t_k)$ é a distancia entre os contatos, $\alpha(t_k)$ representa a variação do rumo do alvo em graus e ΔT é o tempo entre medições. Estas informações são inferidas a partir das medidas fornecidas pelo sensor Radar, descritas na seção anterior. O cálculo destas variáveis é feito para cada um dos contatos disponíveis para associação com um acompanhamento existente. No caso apresentado na figura 5.86, os contatos 1 a 3 representam um acompanhamento existente e os contatos 4 a 5, gerados na mesma varredura, são aqueles que estão em avaliação para serem associados ao acompanhamento. Só um destes contatos será selecionado para associação.

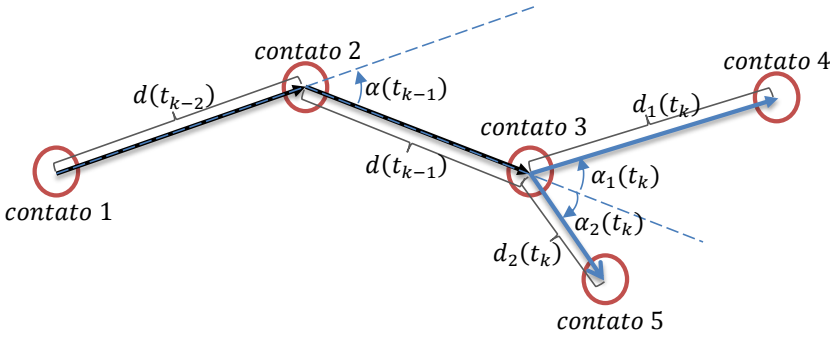


Figura 5.86: Exemplo do calculo das entradas da RNA

Devido à grande quantidade de contatos gerados numa única varredura e à grande área de vigilância do radar, se opta por definir uma heurística para não testar contatos que estejam fora de alcance do alvo que se quer acompanhar. Assim, a

área de vigilância é dividida em 16 setores conforme mostra figura 5.87, portanto, a separação mínima entre dois setores não vizinhos é de aproximadamente 1.96 km, distância impossível de percorrer em 8 seg (tempo máximo entre medições considerando as perdas consecutivas) considerando a velocidade máxima dos alvos estabelecida nesta dissertação. Com isso, define-se um critério segundo o qual, associação entre acompanhamentos e contatos em setores não vizinhos são ignoradas.

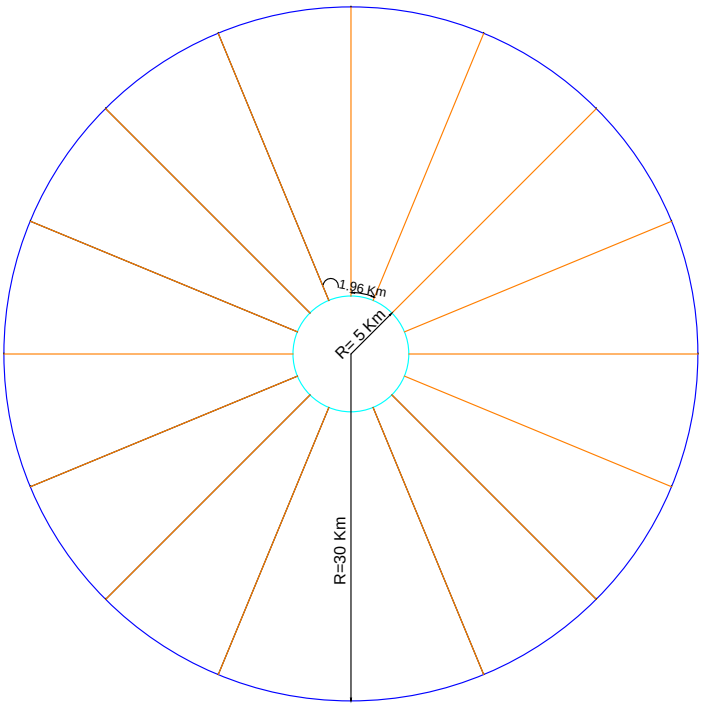


Figura 5.87: Divisão da área de vigilância em setores

Uma vez selecionados os contatos em setores vizinhos, calcula-se a velocidade necessária para percorrer a distância entre o acompanhamento (track) e cada um destes contatos. A partir desta informação, são selecionados candidatos à associação apenas os contatos que tenham velocidade coerente com a descrita pelo alvo. Desta forma, foi estabelecido que um alvo com velocidade superior a 400 m/seg ou inferior a 30 m/seg não pode ser considerado candidato à associação.

A geração das entradas para a aplicação do treinamento da rede, é feita assumindo que as medidas fornecidas pelo sensor até o instante de tempo $k-1$ pertencem à trajetória de um determinado alvo e não a clutter correlacionado. Em outras palavras, são ignorados todos falsos acompanhamentos, já que o objetivo no processo de treinamento é medir o erro na associação entre um contato e um acompanhamento de um alvo existente.

A informação do tipo de contato, fornecida pelo programa gerador de cenário, onde o valor 1 indica que o contato é gerado por um alvo e -1 indica que o contato é clutter, é utilizada para compor o vetor de saídas desejadas, que serve para calcular

o erro na saída da rede, já que é assumido que o contato identificado como tipo 1 é o mais adequado ou coerente para atualizar o acompanhamento existente do mesmo alvo.

Resumindo, o que se espera ao final do treinamento da rede é que ela consiga associar ao acompanhamento existente o contato que faça maior coerência em termos de cinemática. Sob este ponto de vista, se o acompanhamento é composto unicamente por clutter correlacionado, associar clutter que apresente coerência com este acompanhamento não seria um erro de associação da rede. Este caso é analisado no processo de validação do método proposto.

A partir dos cenários gerados para o treinamento da rede e da descrição sobre como é calculado cada um dos pares entrada-saída de treinamento, obtém-se 2766 pares de dados dos quais, 831 representam a associação do acompanhamento com o contato gerado pelo mesmo alvo e 1935 representam associação com clutter. Conforme referido no capítulo 3, estes dados devem passar por um processo de pre-processamento como forma de facilitar o trabalho da rede neural. Assim, é feito um processo de remoção da média e escalamento dos dados, utilizando a toolbox de pre-processamento de dados do *software* Matlab. O conjunto de entradas é dividido em 3 subconjuntos nomeadamente treinamento, validação e teste, definidos aleatoriamente com proporções de 60%, 20% e 20% respectivamente. Cada um destes subconjuntos é avaliado no processo de treinamento para evitar a perda de generalização da rede neural.

5.4.1 Estrutura e Treinamento da Rede

Conforme referido no capítulo 3 o algoritmo de treinamento *backpropagation* com otimização Levenberg-Marquart é uma das abordagens que apresentam melhor desempenho no treinamento de redes neurais alimentadas adiante (*feedforward*) [29]. Embora este algoritmo misture alta velocidade de convergência com estabilidade, seu custo computacional pode ser elevado, por ser um método de segunda ordem. Neste trabalho se compara o uso deste algoritmo com o *backpropagation* com gradiente descendente para treinar uma rede MLP *feedforward*, utilizando como critério de erro a ser minimizado o erro quadrático médio (MSE, sigla em Inglês). O número de neurônios na camada intermediária é ajustado entre 1 e 10 neurônios sendo de tipo tangente hiperbólico e na camada de saída é utilizado um único neurônio de tipo tangente hiperbólico. Finalmente, é estabelecido um máximo de 1000 épocas para atingir o menor erro possível.

Para a implementação da rede neural é utilizada a *toolbox* de redes neurais artificiais que integra o *software* Matlab.

5.5 Confirmação e eliminação de acompanhamentos

Para a confirmação ou eliminação do acompanhamento é utilizado numa fase inicial o critério M/N descrito no capítulo 2, onde M e N são empiricamente definidos com valores iguais a 3 e 4 respectivamente. Se o critério é satisfeito e o acompanhamento confirmado, este é levado à fase seguinte, que determinará se no decorrer do tempo o mesmo deve ou não ser eliminado. Com este propósito foi estabelecido o seguinte: se um acompanhamento apresentar mais do que três perdas consecutivas é considerado como perdido e posteriormente eliminado.

5.6 Filtragem

A última etapa do nosso modelo proposto consiste na filtragem dos acompanhamentos a qual fornece uma aproximação da trajetória real dos alvos de interesse a partir das informações ruidosas contidas nesses acompanhamentos. Com este propósito, foi selecionada a filtragem IMM, descrita no capítulo 4.

Em alguns problemas de rastreamento, o algoritmo IMM é implementado considerando três modelos de filtro para descrever a cinemática do alvo [43, 48]. No entanto, para nosso trabalho foram considerados apenas dois modelos de filtro nomeadamente o modelo de Movimento Retilíneo Uniforme (MRU) e modelo de Giro Coordenado (*Coordinated Turn Model*, CT) [49, 50, 51].

A escolha dos modelos MRU e CT se justifica ao fato de que um navio, na maior parte do tempo, efetua um movimento parcimonioso que permite economizar o consumo de combustível, exceto em alguns casos em que há necessidade de alterar o movimento para executar alguma manobra [52], nos quais o MRU não seria o modelo adequado para descrever a cinemática do alvo.

Uma questão fundamental na construção destes modelos de KF's é a escolha do sistema de coordenadas utilizado na modelagem do movimento do alvo. No caso de movimiento bidimensional, tanto as coordenadas polares quanto as coordenadas cartesianas podem servir de base para a modelagem, embora, na prática, segundo BAR-SHALOM e LI [4], estas últimas são a melhor opção para modelar o movimento do alvo. Assim, neste trabalho, opta-se por utilizar o sistema de coordenadas cartesianas. A justificativa para esta opção é que, além da simplicidade de interpretação geométrica da trajetória, geralmente, o modelo físico utilizado para caracterizar a cinemática do alvo é estabelecido neste sistema de coordenadas.

5.6.1 Definição da Equação do processo

A seguir são detalhados os componentes da equação do processo para cada um dos modelos de filtro utilizados.

Modelo MRU:

Para o modelo MRU definimos o vetor de estados, $X(k)$, por

$$X(k) = \begin{bmatrix} x(k) \\ V_x(k) \\ y(k) \\ V_y(k) \end{bmatrix}. \tag{5.7}$$

A matriz de transição de estados deste modelo é dada por

$$F(k) = \begin{bmatrix} 1 & T_k & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T_k \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{5.8}$$

O ruído de processo é definido como ruído branco na aceleração ("WNA"white noise acceletarion). Assim, o vetor do ruído de processo, $W(k)$, é dado por

$$W(k) = \begin{bmatrix} \frac{T_k^2}{2} & 0 \\ T_k & 0 \\ 0 & \frac{T_k^2}{2} \\ 0 & T_k \end{bmatrix} \cdot \begin{bmatrix} w_x \\ w_y \end{bmatrix}, \tag{5.9}$$

onde T_k é o período de amostragem, considerado variável, por depender do tempo de rotação da antena, da velocidade e percurso do alvo, w_x e w_y representam o ruído de processo em cada um dos eixos coordenados. Assumindo w_x e w_y independentes com média zero e variâncias constantes e iguais ($\sigma_x^2 = \sigma_y^2 = \sigma_q^2$) [47, 52], a matriz de covariância associada ao ruído do processo, $Q(k)$, é dada por

$$Q(k) = \begin{bmatrix} \frac{T_k^4}{4}\sigma_q^2 & \frac{T_k^3}{2}\sigma_q^2 & 0 & 0 \\ \frac{T_k^3}{2}\sigma_q^2 & T_k^2\sigma_q^2 & 0 & 0 \\ 0 & 0 & \frac{T_k^4}{4}\sigma_q^2 & \frac{T_k^3}{2}\sigma_q^2 \\ 0 & 0 & \frac{T_k^3}{2}\sigma_q^2 & T_k^2\sigma_q^2 \end{bmatrix}. \tag{5.10}$$

Finalmente a equação do processo deste modelo é escrita da seguinte maneira

$$X(k+1) = \begin{bmatrix} 1 & T_k & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T_k \\ 0 & 0 & 0 & 1 \end{bmatrix} * X(k) + \begin{bmatrix} \frac{T_k^2}{2} & 0 \\ T_k & 0 \\ 0 & \frac{T_k^2}{2} \\ 0 & T_k \end{bmatrix} \cdot \begin{bmatrix} w_x \\ w_y \end{bmatrix}. \tag{5.11}$$

Observe que esta equação é linear e, por isso, não é preciso fazer qualquer outra manipulação.

Modelo CT:

Para o modelo CT definimos o vetor de estados, $X(k)$, por

$$X(k) = \begin{bmatrix} x(k) \\ V_x(k) \\ y(k) \\ V_y(k) \\ \omega(k) \end{bmatrix} \tag{5.12}$$

e a sua respectiva matriz de transição de estados dada por

$$F(k) = \begin{bmatrix} 1 & T_k & 0 & -\omega(k) * \frac{T_k^2}{2} & 0 \\ 0 & 1 - \omega(k)^2 * \frac{T_k^2}{2} & 0 & -\omega(k) * T_k & 0 \\ 0 & \omega(k) * \frac{T_k^2}{2} & 1 & T_k & 0 \\ 0 & \omega(k) * T_k & 0 & 1 - \omega(k)^2 * \frac{T_k^2}{2} & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{5.13}$$

Neste modelo são considerados dois níveis de ruído: um para a parcela do estado linear e outro para a parcela de giro. Assim, o vetor do ruído de processo, $W(k)$, é dado por,

$$W(k) = \begin{bmatrix} \frac{T_k^2}{2} & 0 & 0 \\ T_k & 0 & 0 \\ 0 & \frac{T_k^2}{2} & 0 \\ 0 & T_k & 0 \\ 0 & 0 & T_k \end{bmatrix} * \begin{bmatrix} w_x \\ w_y \\ w_\omega \end{bmatrix}, \tag{5.14}$$

onde w_x e w_y são as componentes do ruído de processo vistas como uma perturbação na aceleração e w_ω é o ruído na taxa de giro. Todas estes componentes são assumidos independentes, com media zero e variâncias σ_x^2, σ_y^2 e σ_ω^2 respectivamente. Desta

forma, a matriz da covariância associada ao ruído do processo, $Q(k)$, é dada por

$$Q(k) = \begin{bmatrix} \frac{T_k^4}{4} * \sigma_q^2 & \frac{T_k^3}{2} * \sigma_q^2 & 0 & 0 & 0 \\ \frac{T_k^3}{2} * \sigma_q^2 & \frac{T_k^2}{2} * \sigma_q^2 & 0 & 0 & 0 \\ 0 & 0 & \frac{T_k^4}{4} * \sigma_q^2 & \frac{T_k^3}{2} * \sigma_q^2 & 0 \\ 0 & 0 & \frac{T_k^3}{2} * \sigma_q^2 & \frac{T_k^2}{2} * \sigma_q^2 & 0 \\ 0 & 0 & 0 & 0 & \frac{T_k^2}{2} * \sigma_\omega^2 \end{bmatrix}, \tag{5.15}$$

onde $\sigma_q^2 = \sigma_x^2 = \sigma_y^2$.

Finalmente a equação do processo deste modelo é dada por

$$\begin{aligned} X(k+1) = & \begin{bmatrix} 1 & T_k & 0 & -\omega(k) * \frac{T_k^2}{2} & 0 \\ 0 & 1 - \omega(k)^2 * \frac{T_k^2}{2} & 0 & -\omega(k) * T_k & 0 \\ 0 & \omega(k) * \frac{T_k^2}{2} & 1 & T_k & 0 \\ 0 & \omega(k) * T_k & 0 & 1 - \omega(k)^2 * \frac{T_k^2}{2} & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} * X(k) \\ & + \begin{bmatrix} \frac{T_k^2}{2} & 0 & 0 \\ T_k & 0 & 0 \\ 0 & \frac{T_k^2}{2} & 0 \\ 0 & T_k & 0 \\ 0 & 0 & T_k \end{bmatrix} * \begin{bmatrix} w_x \\ w_y \\ w_\omega \end{bmatrix}. \end{aligned} \tag{5.16}$$

Observe que a equação 5.16 é não linear, pelo que devemos utilizar as equações do EKF, descritas na seção 4.2, para tratar o problema da não linearidade. Desta forma, a equação 5.16 pode ser reescrita como

$$\begin{aligned} X(k+1) &= f(k, X(k)) + W(k) \\ &= \begin{bmatrix} x(k) + V_x(k) * T_k - V_y(k) * \omega(k) * \frac{T_k^2}{2} \\ V_x(k) - V_x(k) * \omega(k)^2 * \frac{T_k^2}{2} - V_y(k) * \omega(k) * T_k \\ y(k) + V_y(k) * T_k + V_x(k) * \omega(k) * \frac{T_k^2}{2} \\ V_y(k) - V_y(k) * \omega(k)^2 * \frac{T_k^2}{2} + V_x(k) * \omega(k) * T_k \\ \omega(k) \end{bmatrix} \\ &+ \begin{bmatrix} \frac{T_k^2}{2} & 0 & 0 \\ T_k & 0 & 0 \\ 0 & \frac{T_k^2}{2} & 0 \\ 0 & T_k & 0 \\ 0 & 0 & T_k \end{bmatrix} * \begin{bmatrix} w_x \\ w_y \\ w_\omega \end{bmatrix}. \end{aligned} \tag{5.17}$$

Devido ao processo de linearização proposto na equação 4.29, é necessário calcular

a matriz jacobiana da função $f(k, \hat{X}(k|k))$. Para manter a mesma notação utilizada na seção 4.2, esta matriz é denotada por $F(k)$ e calculada conforme referido nesta mesma seção. Desta forma, teremos que (pela simplicidade de notação omitiu-se o índice (k|k) na equação abaixo)

$$F(k) = \begin{bmatrix} 1 & T_k & 0 & -\hat{\omega} * \frac{T_k^2}{2} & -\hat{V}_y * \frac{T_k^2}{2} \\ 0 & 1 - \hat{\omega}^2 * \frac{T_k^2}{2} & 0 & -\hat{\omega} * T_k & -\hat{V}_y * T_k - \hat{V}_x * \hat{\omega} * T_k^2 \\ 0 & \hat{\omega} * \frac{T_k^2}{2} & 1 & T_k & \hat{V}_x * \frac{T_k^2}{2} \\ 0 & \hat{\omega} * T_k & 0 & 1 - \hat{\omega}^2 * \frac{T_k^2}{2} & \hat{V}_x * T_k - \hat{V}_y * \hat{\omega} * T_k^2 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \quad (5.18)$$

A dedução das matrizes de transição de estados, equações 5.8 e 5.13, dos vetores de ruído do processo, equações 5.9 e 5.14, as matrizes da covariância associadas ao ruído do processo, equações 5.10 e 5.15 e as equações do processo 5.11 e 5.17 são detalhadas no apêndice A.

5.6.2 Definição da Equação de medida

Conforme referido na seção 5.1, o sistema Radar fornece o vetor de medidas ($Z(k)$) em coordenadas polares, pelo que uma transformação não linear deve ser feita para relacionar este vetor com o vetor de estados do alvo $X(k)$ (definido em coordenadas cartesianas) de cada um dos filtros. Para isso existem dois enfoques: o primeiro consiste em transformar a medida em coordenadas cartesianas antes da entrada do filtro e o segundo enfoque é considerar a medida na forma original como entrada do filtro. No entanto, devido à natureza não linear da transformação, o último enfoque exige a utilização das equações do filtro de Kalman Estendido (EKF), para tratar esta não linearidade. Foi justamente este enfoque considerado no nosso trabalho.

Através de uma função de transformação trigonométrica, as medidas distância e azimuth são calculados em função das componentes de posição em coordenadas cartesianas ($x(k)$ e $y(k)$) da seguinte maneira

$$\text{distancia}(k) = \sqrt{x(k)^2 + y(k)^2} = fh_1 \quad (5.19)$$

$$\text{azimuth}(k) = \arctan\left(\frac{x(k)}{y(k)}\right) = fh_2. \quad (5.20)$$

Assim, a função vetorial não linear que relaciona as variáveis de estado às medidas, denotada por $fh(k+1, X(k+1))$ (mantendo a mesma notação utilizada na seção 4.2, é dada por

$$fh(k+1, X(k+1)) = \begin{bmatrix} fh_1 = \sqrt{x(k+1)^2 + y(k+1)^2} \\ fh_2 = \arctan\left(\frac{x(k+1)}{y(k+1)}\right) \end{bmatrix}. \quad (5.21)$$

Esta função é igual para os dois modelos. Assim, a equação da medida é definida como

$$Z(k+1) = \begin{bmatrix} \sqrt{x(k+1)^2 + y(k+1)^2} \\ \arctan\left(\frac{x(k+1)}{y(k+1)}\right) \end{bmatrix} + e(k+1) \tag{5.22}$$

Devido ao processo de linearização proposto na equação 4.29, é necessário calcular a matriz jacobiana da função $fh(k+1, \hat{X}(k+1|k))$ para cada um dos modelos de filtro. Esta matriz é denotada por $H1(k+1)$ no caso do filtro com modelo MRU e $H2(k+1)$ no caso do modelo CT. Elas são definidas como

$$\begin{aligned} H1(k+1) &= \begin{bmatrix} \frac{dfh_1}{dx} & \frac{dfh_1}{dV_x} & \frac{dfh_1}{dy} & \frac{dfh_1}{dV_y} \\ \frac{dfh_2}{dx} & \frac{dfh_2}{dV_x} & \frac{dfh_2}{dy} & \frac{dfh_2}{dV_y} \end{bmatrix} \\ &= \begin{bmatrix} \frac{\hat{x}(k+1|k)}{\sqrt{(\hat{x}(k+1|k)^2+\hat{y}(k+1|k)^2)}} & 0 & \frac{\hat{y}(k+1|k)}{\sqrt{(\hat{x}(k+1|k)^2+\hat{y}(k+1|k)^2)}} & 0 \\ \frac{\hat{y}(k+1|k)}{\hat{x}(k+1|k)^2+\hat{y}(k+1|k)^2} & 0 & -\frac{\hat{x}(k+1|k)}{\hat{x}(k+1|k)^2+\hat{y}(k+1|k)^2} & 0 \end{bmatrix} \end{aligned} \tag{5.23}$$

$$\begin{aligned} H2(k+1) &= \begin{bmatrix} \frac{dfh_1}{dx} & \frac{dfh_1}{dV_x} & \frac{dfh_1}{dy} & \frac{dfh_1}{dV_y} & \frac{dfh_1}{d\omega} \\ \frac{dfh_2}{dx} & \frac{dfh_2}{dV_x} & \frac{dfh_2}{dy} & \frac{dfh_2}{dV_y} & \frac{dfh_2}{d\omega} \end{bmatrix} \\ &= \begin{bmatrix} \frac{\hat{x}(k+1|k)}{\sqrt{(\hat{x}(k+1|k)^2+\hat{y}(k+1|k)^2)}} & 0 & \frac{\hat{y}(k+1|k)}{\sqrt{(\hat{x}(k+1|k)^2+\hat{y}(k+1|k)^2)}} & 0 & 0 \\ \frac{\hat{y}(k+1|k)}{\hat{x}(k+1|k)^2+\hat{y}(k+1|k)^2} & 0 & -\frac{\hat{x}(k+1|k)}{\hat{x}(k+1|k)^2+\hat{y}(k+1|k)^2} & 0 & 0 \end{bmatrix} \end{aligned} \tag{5.24}$$

Adicionalmente a partir da definição do vetor de ruídos $e(k)$ feita na seção 5.1, a matriz da covariância dos ruídos associados à medida, $R(k+1)$, é dada por

$$R(k+1) \;=\; \begin{bmatrix} \sigma_d^2 & 0 \\ 0 & \sigma_\theta^2 \end{bmatrix}. \tag{5.25}$$

Esta matriz também é a mesma para ambos os filtros. Os valores de σ_d e σ_θ serão estabelecidos mais adiante.

Resumindo, para o modelo MRU temos equação do processo linear e equação da medida não linear e no modelo CT temos ambas as equações não lineares, por conseguinte, em ambos os modelos será utilizado o filtro de Kalman Estendido.

5.6.3 Inicialização dos modelos de filtro

Para executar o ciclo recursivo do KF e do EKF é preciso definir os valores da estimativa inicial do estado ($\hat{X}(0|0)$), da matriz de covariância desta estimativa inicial ($P(0|0)$) e das covariâncias dos ruídos associados ao processo (σ_q^2) e à medida

$(\sigma_d^2, \sigma_\theta^2)$. Diversas abordagens de inicialização de parâmetros têm sido propostas, mas não existe uma solução geral para este problema, visto que o processo de inicialização depende do modelo do processo empregado, intervalo de amostragem e outros fatores inerentes.

Segundo WHYTE [38], no modelo de KF linear, a escolha das estimativas iniciais, $\hat{X}(0|0)$ e $P(0|0)$, não afeta a convergência para o estado real, uma vez que a sua influência sobre o modelo diminui com o tempo. Por esta razão, o modelo de KF linear é considerado estável sob qualquer perturbação entorno do estado real. Por outro lado, no modelo de EKF abordado neste trabalho, a inicialização dos parâmetros é uma das questões práticas mais difíceis de se resolver em problemas reais. Neste modelo, que é baseado em diversos processos de linearização, a convergência é sensível à escolha das estimativas iniciais dos parâmetros. Portanto, uma escolha razoável destas estimativas é apresentada a seguir.

a) **Estimativa inicial do estado**

Um método simples de obter a estimativa inicial do estado é utilizar as primeiras observações do alvo fornecidas pelo sensor [38, 53]. Assim, para inicialização dos nossos modelos, foram consideradas as duas primeiras medidas, conforme o procedimento apresentado em [49]. É importante lembrar que estas observações são dadas em coordenadas polares, $Z_m = \{d_m,\ \theta_m\}$, e o azimuth (marcação) é medido com respeito ao Norte geográfico. Por conveniência, neste caso, o azimuth será recalculado tomando como referência o Este geográfico (eixo X). Desta forma, podemos calcular a posição e a velocidade em cada um dos eixos cartesianos a partir destas duas medidas da seguinte maneira:

$$x_{m(j)} = d_{m(j)} * \cos\theta_{m(j)}, \tag{5.26}$$

$$y_{m(j)} = d_{m(j)} * \sin\theta_{m(j)}, \tag{5.27}$$

$$Vx_{m(j)} = \frac{x_{m(j)} - x_{m(j-1)}}{t_{m(j)} - t_{m(j-1)}}, \tag{5.28}$$

$$Vy_{m(j)} = \frac{y_{m(j)} - y_{m(j-1)}}{t_{m(j)} - t_{m(j-1)}}, \tag{5.29}$$

onde $j = 2$ é o número de medidas consideradas, $d_{m(j)}$ e $\theta_{m(j)}$ são componentes da j-ésima medida e o termo $t_{m(j)} - t_{m(j-1)} = T$ é o tempo entre amostras.

No modelo MRU, onde o vetor de estados é definido por 5.7, a estimativa do

estado inicial é dada por

$$\hat{X}(0|0) = \begin{bmatrix} \hat{x}(0) = x_{m(2)} = d_{m(2)} * \cos\theta_{m(2)} \\ \hat{V}_x(0) = \frac{x_{m(2)} - x_{m(1)}}{T} \\ \hat{y}(0) = y_{m(2)} = d_{m(2)} * \sin\theta_{m(2)} \\ \hat{V}_y(0) = \frac{y_{m(2)} - y_{m(1)}}{T} \end{bmatrix}, \tag{5.30}$$

e no modelo CT, onde o vetor de estados é definido por 5.12), a estimativa inicial é dada por

$$\hat{X}(0|0) = \begin{bmatrix} \hat{x}(0) = x_{m(2)} = d_{m(2)} * \cos\theta_{m(2)} \\ \hat{V}_x(0) = \frac{x_{m(2)} - x_{m(1)}}{T} \\ \hat{y}(0) = y_{m(2)} = d_{m(2)} * \sin\theta_{m(2)} \\ \hat{V}_y(0) = \frac{y_{m(2)} - y_{m(1)}}{T} \\ \hat{\omega}(0) = 0 \end{bmatrix}. \tag{5.31}$$

A estimativa inicial da taxa de giro ($\hat{\omega}(0)$) é igual a zero na medida em que não se tem conhecimento do sentido para o qual o alvo está girando (à direta, $\omega > 0$, ou à esquerda, $\omega < 0$). A partir desta estimativa inicial, o algoritmo é capaz de estimar a taxa de giro incluindo o sinal, pelo que não é necessário criar um modelo diferenciado para cada sentido.

b) **Matriz de covariância da estimativa inicial**

Um método simples de inicializar a matriz de covariância é defini-la como $P(0|0) = \alpha^2 * Q(k)$, sendo o valor típico de $\alpha = 10$ e lembrando que $Q(k)$ representa a covariância dos erros associados ao processo. Como regra geral, o valor inicial da covariância deve ser o suficientemente grande para englobar o alto nível de incerteza na estimação inicial do estado. Uma escolha mais elaborada da matriz de covariância pode ser encontrada computando o valor esperado dos erros da estimação, isto é, o erro quadrático médio [38, 53],

$$P(0|0) = E[(\hat{X}(0|0) - X(0))(\hat{X}(0|0) - X(0))^T] \tag{5.32}$$

onde $X(0)$ é o estado real inicial. Sabemos que a medida fornecida pelo sensor radar (d_m e θ_m) é definida em função da posição real do contato, d_r e θ_r, acrescida pelo seus respectivos erros, e_d e e_θ, assumidos independentes com média zero e variância σ_d^2 e σ_θ^2 , i.e,

$$d_m = d_r + e_d \tag{5.33}$$

$$\theta_m = \theta_r + e_\theta. \tag{5.34}$$

Intuitivamente podemos expressar a posição e velocidade real do alvo nas componentes cartesianas, como

$$x_{r(j)} = d_r * \cos\theta_{r(j)} \tag{5.35}$$

$$y_{r(j)} = d_r * \sin\theta_{r(j)} \tag{5.36}$$

$$Vx_{r(j)} = \frac{x_{r(j)} - x_{r(j-1)}}{T} \tag{5.37}$$

$$Vy_{r(j)} = \frac{y_{r(j)} - y_{r(j-1)}}{T} \tag{5.38}$$

onde $x_{r(j)}$ e $y_{r(j)}$ denotam a posição real do alvo em coordenadas cartesianas.

Para evitar a não linearidade das equações 5.26 e 5.35, foi feita uma aproximação linear com os termos de primeira ordem da expansão em series de Taylor (mais discussões e detalhes sobre aproximações destas funções podem ser encontradas em [4, 54, 55]). Com isto, podemos definir o erro na medição em coordenadas cartesianas como

$$x_{m(j)} - x_{r(j)} \approx e_d * \cos\theta_{m(i)} - e_\theta * d_{m(i)} * \sin\theta_{m(i)} = x_{L(j)} \tag{5.39}$$

$$y_{m(j)} - y_{r(j)} \approx e_d * \sin\theta_{m(i)} + e_\theta * d_{m(i)} * \cos\theta_{m(i)} = y_{L(j)} \tag{5.40}$$

$$Vx_{m(j)} - Vx_{r(j)} \approx \frac{x_{L(j)} - x_{L(j-1)}}{T} \tag{5.41}$$

$$Vy_{m(j)} - Vy_{r(j)} \approx \frac{y_{L(j)} - y_{L(j-1)}}{T} \tag{5.42}$$

A média dos erros definidos nas equações acima é zero.

De forma análoga à equação 5.31 podemos definir o vetor inicial de estados reais para o modelo MRU da seguinte maneira

$$X(0) = \begin{bmatrix} x(0) = x_{r(2)} = d_{r(2)} * \cos\theta_{r(2)} \\ V_x(0) = \frac{x_{r(2)} - x_{r(1)}}{T} \\ y(0) = y_{r(2)} = d_{r(2)} * \sin\theta_{r(2)} \\ V_y(0) = \frac{y_{r(2)} - y_{r(1)}}{T} \end{bmatrix}. \tag{5.43}$$

Por conseguinte, podemos definir a matriz de covariância simétrica de dimensão 4×4 para o modelo MRU, cujos elementos, considerando a equação

5.32, são dados por:

$$P_{11} = E[(\hat{x}(0) - x(0))^2] = \sigma_d^2 * \cos\theta_{m(2)}{}^2 + \sigma_\theta^2 * d_{m(2)}^2 * \sin\theta_{m(2)}{}^2$$
$$= \sigma_x^2 \tag{5.44}$$

$$P_{12} = P_{21} = E[(\hat{x}(0) - x(0))(\hat{V}_x(0) - V_x(0))] = \frac{\sigma_x^2}{T} \tag{5.45}$$

$$P_{13} = P_{31} = E[(\hat{x}(0) - x(0))(\hat{y}(0) - y(0))]$$
$$= (\sigma_d^2 - \sigma_\theta^2 * d_{m(2)}) * \cos\theta_{m(2)} * \sin\theta_{m(2)} = \sigma_{xy}^2 \tag{5.46}$$

$$P_{14} = P_{41} = E[\hat{x}(0) - x(0))(\hat{V}_y(0) - V_y(0))] = \frac{\sigma_{xy}^2}{T} \tag{5.47}$$

$$P_{22} = E[(\hat{V}_x(0) - V_x(0))^2] = \frac{2\sigma_x^2}{T^2} \tag{5.48}$$

$$P_{23} = P_{32} = E[(\hat{V}_x(0) - V_x(0))(\hat{y}(0) - y(0))] = \frac{\sigma_{xy}^2}{T} \tag{5.49}$$

$$P_{24} = P_{42} = E[(\hat{V}_x(0) - V_x(0))(\hat{V}_y(0) - V_y(0))] = \frac{2\sigma_{xy}^2}{T^2} \tag{5.50}$$

$$P_{33} = E[(\hat{y}(0) - y(0))^2] = \sigma_y^2 \tag{5.51}$$

$$P_{34} = P_{43} = E[\hat{y}(0) - y(0))(\hat{V}_y(0) - V_y(0))] = \frac{\sigma_y^2}{T} \tag{5.52}$$

$$P_{44} = E[(\hat{V}_y(0) - V_y(0))^2] = \frac{2\sigma_y^2}{T^2} \tag{5.53}$$

reorganizando estes componentes em forma de matriz teremos

$$P(0|0) = \begin{bmatrix} \sigma_x^2 & \frac{\sigma_x^2}{T} & \sigma_{xy}^2 & \frac{\sigma_{xy}^2}{T} \\ \frac{\sigma_x^2}{T} & \frac{2\sigma_x^2}{T^2} & \frac{\sigma_{xy}^2}{T} & \frac{2\sigma_{xy}^2}{T^2} \\ \sigma_{xy}^2 & \frac{\sigma_{xy}^2}{T} & \sigma_y^2 & \frac{\sigma_y^2}{T} \\ \frac{\sigma_{xy}^2}{T} & \frac{2\sigma_{xy}^2}{T^2} & \frac{\sigma_y^2}{T} & \frac{2\sigma_y^2}{T^2} \end{bmatrix}. \tag{5.54}$$

A matriz inicial da covariância dos erros da estimação do modelo CT é calculada de forma análoga ao modelo MRU, sendo única diferença a inclusão da variável de taxa de giro (ω) no vetor de estados. Portanto, o vetor inicial de estados reais é dado por

$$X(0) = \begin{bmatrix} x(0) = x_{r(2)} = d_{r(2)} * \cos\theta_{r(2)} \\ V_x(0) = \frac{x_{r(2)} - x_{r(1)}}{T} \\ y(0) = y_{r(2)} = d_{r(2)} * \sin\theta_{r(2)} \\ V_y(0) = \frac{y_{r(2)} - y_{r(1)}}{T} \\ \omega(0) = 0 \end{bmatrix} \tag{5.55}$$

e a sua respectiva a matriz de covariância definida por

$$P(0|0) = \begin{bmatrix} \sigma_x^2 & \frac{\sigma_x^2}{T} & \sigma_{xy}^2 & \frac{\sigma_{xy}^2}{T} & 0 \\ \frac{\sigma_x^2}{T} & \frac{2\sigma_x^2}{T^2} & \frac{\sigma_{xy}^2}{T} & \frac{2\sigma_{xy}^2}{T^2} & 0 \\ \sigma_{xy}^2 & \frac{\sigma_{xy}^2}{T} & \sigma_y^2 & \frac{\sigma_y^2}{T} & 0 \\ \frac{\sigma_{xy}^2}{T} & \frac{2\sigma_{xy}^2}{T^2} & \frac{\sigma_y^2}{T} & \frac{2\sigma_y^2}{T^2} & 0 \\ 0 & 0 & 0 & 0 & P_\omega \end{bmatrix}, \tag{5.56}$$

onde $\frac{\omega \max}{2} \leq \sqrt{P_\omega(0|0)} \leq \omega_{\max}$ [49] onde $\omega_{\max} = 0.087$ rad/seg, assim $P_\omega(0|0) = 0.0076$.

c) **Ruído do processo**
A seleção dos níveis de ruído para cada modelo é parte importante do desenho. Deve-se lembrar que o ruído do processo para ambos os modelos é definido como ruído branco na aceleração ("WNA"white noise acceletarion), com a diferença de que no modelo CT é adicionado ruído na parcela de giro. Desta forma, o desvio padrão do ruído de processo para o modelo MRU é definido como mostrado em [4], $\sigma_q = \alpha * a_{max}$ onde $0 < \alpha \leq 1$ e a_{max} é a aceleração máxima esperada. Geralmente para este modelo é selecionado um nível de ruído baixo [45]. Desta forma, são definidos empíricamente $\alpha = 0.7$ e $a_{max} = 5\frac{m}{seg^2}$, com isso, $\sigma_q = 3.5\frac{m}{seg^2}$.

Geralmente no modelo CT o nível de ruído é maior que no modelo MRU, visto que o nível de incerteza é maior no momento das manobras. Assim, definiremos o desvio padrão dos ruídos do processo como: $0.5 * a_{max} \leq \sigma_q \leq a_{max}$ e $0.5 * \omega \leq \sigma_\omega \leq \omega$ onde $a_{max} = 5\frac{m}{seg^2}$ e $\omega = 0.0873\frac{rad}{seg}$, com isto, definimos empíricamente $\sigma_q = 5\frac{m}{seg^2}$ e $\sigma_\omega = 0.044\frac{rad}{seg}$.

d) **Ruído da medida**
Os ruídos de medida dependem diretamente das características do sensor modelado, assim, para o nosso caso são selecionados um desvio padrão na distancia de $\sigma_d = 30m$ e no azimuth $\sigma_\theta = 3mrad$.

Agora para a execução do algoritmo IMM também é necessário estabelecer alguns parâmetros iniciais que são apresentados a seguir:

a) **Definição da matriz de transição** Esta matriz é definida seguindo o processo descrito na seção 4.3, assim, para o nosso caso foi considerado o tempo de rotação da antena de 2 segundos, o tempo esperado de execução do MRU de 200 segundos e o tempo esperado de execução do CT de 20 segundos.

Desta forma teremos

$$p_{11} = 1 - \frac{2}{200} = 0.99 \tag{5.57}$$

$$p_{12} = 1 - p_{11} = 0.01 \tag{5.58}$$

$$p_{22} = 1 - \frac{2}{20} = 0.9 \tag{5.59}$$

$$p_{12} = 1 - p_{22} = 0.1 \tag{5.60}$$

A matriz de transição resultante é expressa da seguinte forma:

$$P = \begin{bmatrix} 0.99 & 0.01 \\ 0.1 & 0.9 \end{bmatrix}. \tag{5.61}$$

b) **Probabilidade inicial do modelo,** $\mu_i(0|0)$ A probabilidade inicial do modelo, $\mu_i(0|0)$, onde i representa o i-ésimo modelo do filtro, é definida a ser 0.5 para cada um dos modelos utilizados (MRU e CT), assim,

$$\mu_1(0|0) = 0.5 \tag{5.62}$$

$$\mu_2(0|0) = 0.5 \tag{5.63}$$

5.7 Métricas

Como critérios de comparação de desempenho entre a rede treinada pelo algoritmo Levenberg-Marquart e a rede treinada pelo método de gradiente descendente são utilizados a taxa de acertos e de erros, a sensibilidade e especificidade, entre outros. Observe que, no nosso contexto, acerto se refere ao fato de o contato selecionado para associação ser gerado por o alvo de interesse.

Para a validação do método proposto e comparação com o método clássico descrito no capitulo 2, serão utilizadas como métricas o número de tracks confirmados, o número de acertos e de erros, e o tempo médio por iteração. Entende-se por tempo de iteração, o tempo que o programa leva em analisar os dados contidos numa única varredura de antena do radar.

Capítulo 6

Resultados

Neste capitulo serão apresentados os resultados obtidos pela rede neural aplicando os algoritmos de treinamento Levenberg-Marquart e gradiente descendente, fazendo uma analise comparativa entre eles. Seguidamente serão apresentados os resultados da execução do método proposto e do método clássico. Finalmente é feita uma avaliação da filtragem realizada.

6.1 Resultados do Treinamento da Rede Neural

Os resultados aqui apresentados são produto do treinamento de uma rede neural MLP que possui 5 neurônios na camada intermediária e 1 na camada de saída, todos eles são do tipo tangente hiperbólica. O número de neurônios na camada intermediária é selecionado depois de se realizar testes variando a topologia desde 1 até 10 neurônios, encontrando que a rede com 5 neurônios na camada intermediária é aquela que apresenta em media menor MSE. É importante lembrar que a saída da rede pode assumir valores entre -1 e 1, sendo que 1 indica que o contato, que gerou a entrada, deve ser considerado como atualização do acompanhamento e -1 no caso contrario. Adicionalmente, o contato que é classificado pelo programa gerador de cenários como gerado pelo alvo é o mais coerente para atualizar o acompanhamento. Com isto em mente, pode-se inferir que a rede consegue diferenciar quando um contato é gerado pelo alvo ou se representa clutter. Por simplicidade definiremos que, quando a saída da rede for maior do que zero, a entrada pertencera à classe alvo, casso contrario a entrada pertencera à classe clutter. A partir desta definição, são estabelecidas as matrizes de confusão apresentadas nesta seção.

A figura 6.1 mostra a evolução do erro de treinamento, teste e validação a cada época de treinamento da rede utilizando o método de gradiente descendente. Sendo que a rede atinge o número máximo de épocas estabelecido (1000) antes de alcançar o menor erro possível.

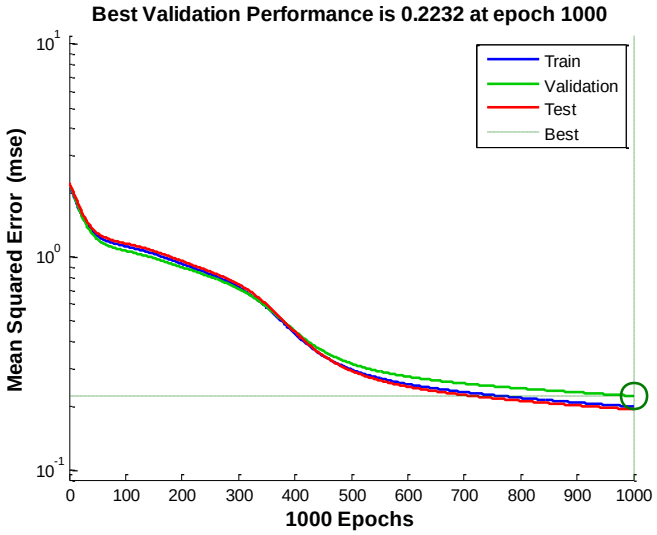


Figura 6.1: Evolução do erro de treinamento, teste e validação, da rede treinada pelo método de gradiente descendente

Na tabela 6.1 são apresentados os resultados da rede treinada pelo método de gradiente descendente. Nela podemos observar que, no total, dos 831 dados de entrada que representam a associação do acompanhamento com o contato gerado pelo mesmo alvo, 784 são classificados corretamente, ou seja, o contato que gerou esta entrada é considerado como atualização do acompanhamento, no entanto, 47 destas entradas são classificadas erroneamente. Enquanto que, dos 1935 dados de entrada que representam associações do acompanhamento com clutter, 1819 são classificados corretamente, ou seja, não são considerados para atualização do acompanhamento. A figura 6.2 mostra a evolução do erro de treinamento, teste e validação a cada época de treinamento da rede utilizando o algoritmo de Levenberg-Marquart. Sendo que neste caso a rede apresenta uma parada prematura do treinamento na época 35 para evitar a perda de generalização.

Na tabela 6.2 são apresentados os resultados da rede treinada pelo algoritmo Levenberg-Marquart. Pode-se observar que, no total, dos 831 dados de entrada que representam a associação do acompanhamento com o contato gerado pelo mesmo alvo, 824 são classificados corretamente e apenas 7 destas entradas são classificadas erroneamente. Enquanto que, dos 1935 dados de entrada que representam associações do acompanhamento com clutter, 1919 são classificados corretamente e 16 destes entradas são classificadas de forma errônea.

Tabela 6.1: Matriz de Confusão da Rede treinada pelo método de gradiente descendente

Treinamento			
		Previsto	
		Alvo	Clutter
Real	Alvo	476	24
	Clutter	71	1089
Validação			
		Previsto	
		Alvo	Clutter
Real	Alvo	149	9
	Clutter	26	369
Teste			
		Previsto	
		Alvo	Clutter
Real	Alvo	159	14
	Clutter	19	361
Total			
		Previsto	
		Alvo	Clutter
Real	Alvo	784	47
	Clutter	116	1819

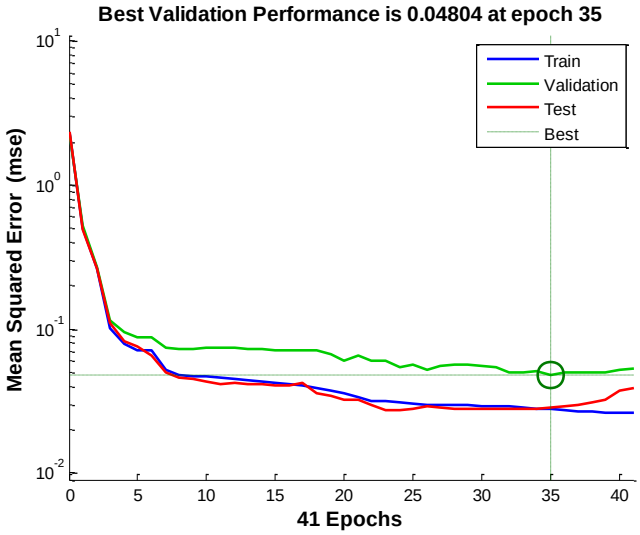


Figura 6.2: Evolução do erro de treinamento, teste e validação, da rede treinada pelo algoritmo Levenberg-Marquart

Da informação contida nas matrizes de confusão acima pode-se calcular a taxa de acerto ou acurácia para cada um dos tipos de treinamento, assim, a taxa de acerto para a rede treinada pelo método de gradiente descendente é de 94.1%, enquanto que, para a rede treinada pelo algoritmo Levenberg-Marquart é de 99.2%. Mas por

Tabela 6.2: Matriz de Confusão da Rede treinada por Levenberg-Marquart

Treinamento			
		Previsto	
		Alvo	Clutter
Real	Alvo	500	4
	Clutter	8	1148
Validação			
		Previsto	
		Alvo	Clutter
Real	Alvo	171	1
	Clutter	6	375
Teste			
		Previsto	
		Alvo	Clutter
Real	Alvo	153	2
	Clutter	2	396
Total			
		Previsto	
		Alvo	Clutter
Real	Alvo	824	7
	Clutter	16	1919

se tratar de um problema com classes desbalanceadas (approximandamente 70% dos dados de entrada representam associações do acompanhamento com clutter) é desejável utilizar uma medida de desempenho diferente da taxa de acerto, por este motivo, serão calculadas a sensibilidade, especificidade e a taxa de falsos positivos, além disso, é construída a curva ROC (Receiver Operating Characteristic).

Entende-se por sensibilidade a probabilidade de classificar corretamente uma entrada cujo estado real é definido como positivo, no nosso contexto o estado positivo refere-se à classe alvo, a qual é a classe de maior interesse, este parâmetro é calculado por

$$S = \frac{V_P}{V_P + F_N}, \tag{6.1}$$

onde V_P é o número de entradas da classe alvo que foram classificadas corretamente e F_N é o número de entradas da classe alvo que foram classificadas erroneamente. A especificidade é a probabilidade de classificar corretamente uma entrada cujo estado real é definido como negativo, no nosso contexto o estado negativo refere-se à classe

clutter, este parâmetro é calculado por

$$E = \frac{V_N}{V_N + F_P}, \tag{6.2}$$

onde V_N é o número de entradas da classe clutter que foram classificados corretamente e F_P é o número de entradas da classe clutter que foram classificados erroneamente. Finalmente, a taxa de falsos positivos, refere-se a porcentagem de amostras erroneamente classificadas como positivas dentre todas as entradas cujo estado real é negativo.

$$FPR = \frac{F_P}{V_N + F_P}. \tag{6.3}$$

A partir das equações anteriores é construída a tabela 6.3, dela podemos destacar que em todos os aspectos a rede treinada pelo algoritmo Levenberg-Marquart apresenta melhor desempenho que a rede treinada pelo método de gradiente descendente, em outras palavras, a probabilidade de classificar corretamente aquelas entradas cujo estado real é considerado tanto positivo como negativo é maior utilizando o algoritmo Levenberg-Marquart.

Tabela 6.3: Medidas de desempenho dos algoritmos de treinamento da rede neural

	Gradiente descendente	Levenberg-Marquart
Verdadeiros positivos (V_P)	784	824
Falsos Negativos (F_N)	47	7
Verdadeiros Negativos (V_N)	1819	1919
Falsos Positivos (F_P)	116	16
Sensibilidade	0.9434	0.9916
Especificidade	0.94	0.9917
Taxa de Falsos positivos	0.0599	0.0083

Nas figuras 6.3 e 6.4 são apresentadas as curvas ROC obtidas para cada um destes tipos de treinamento, esta curva serve como uma forma alternativa de testar o desempenho da rede. A curva é construída plotando a Taxa de Falsos Positivos (TFP) vs a Sensibilidade (S), sendo que o ponto onde se obtém o melhor desempenho da classificação é quando TFP é igual a 0 e a sensibilidade é a igual a 1, neste caso todos os pontos são classificados corretamente. Como é observado a rede treinada com o algoritmo Levenberg-Marquart se encontra perto deste objetivo. Outra forma de medir o desempenho destas redes utilizando a curva ROC é calcular a área abaixo da curva, quanto maior seja esta área melhor é o desempenho da rede. Assim, a rede treinada pelo método do gradiente tem uma área de 0.9790, já a curva ROC da rede treinada pelo algoritmo Levenberg-Marquart tem uma área

de 0.9981, confirmando assim o desempenho superior deste tipo de treinamento. Por conseguinte a rede treinada pelo algoritmo Levenberg-Marquart é a escolhida para a validação do método de acompanhamento de múltiplos alvos proposto nesta dissertação.

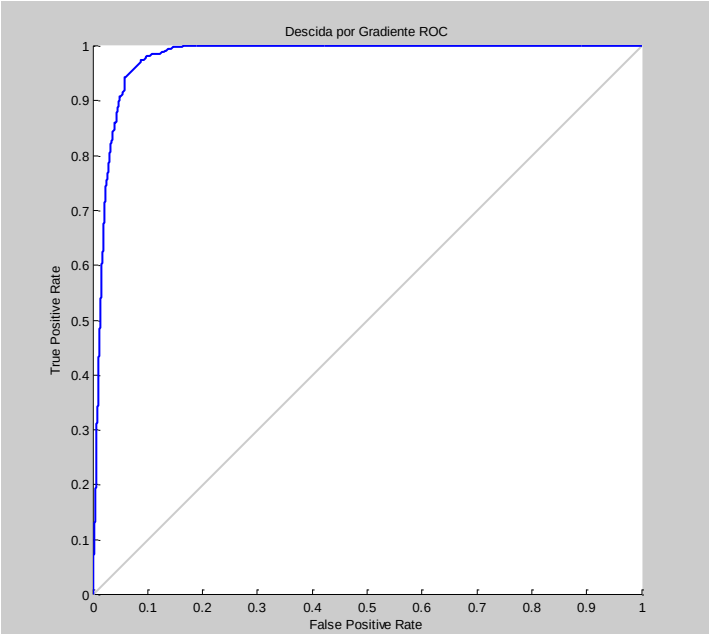


Figura 6.3: Curva ROC, da rede treinada pelo método de gradiente descendente

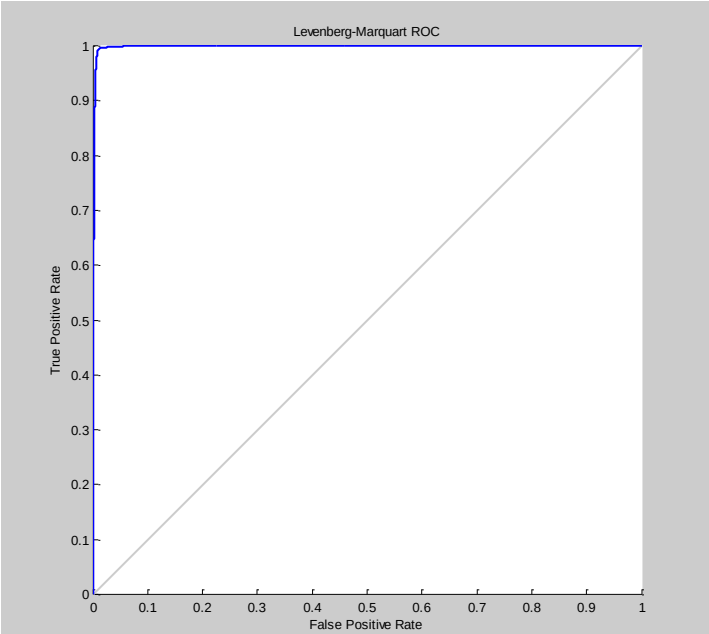


Figura 6.4: Curva ROC, da rede treinada pelo algoritmo Levenberg-Marquart

6.2 Resultados da execução do método proposto e do método clássico

Os cenários aqui utilizados são descritos na seção 5.2 , a duração de cada um destes cenários é de 200 segundos, sendo que, as trajetórias que os compõem apresentam uma duração inferior a este limite. A execução destes algoritmos é feita num computador com processador Intel Core i5-2450M, 2.50 GHz e 4 Gb de memória RAM.

Nas figuras que apresentam os resultados obtidos da execução do método clássico para cada um dos cenários os pontos de cor verde representam contatos que foram utilizados para atualizar algum acompanhamento, os pontos de cor preta representam contatos detectados até 10 varreduras após a atual e os pontos azuis representam os contatos da última varredura, a linha vermelha representa a trajetória filtrada pelo IMM. No caso das figuras que apresentam os resultados obtidos pela execução do método proposto os pontos de cor preta representam os contatos utilizados para atualizar um acompanhamento que tenha sido confirmado previamente, os pontos de cor rosa representam contatos utilizados para atualizar algum acompanhamento antes da sua confirmação, os pontos de cor vermelha representam contatos de possível associação com um acompanhamento existente mas que não são selecionados para atualiza-lo, os pontos verdes representam os contatos que de fato são gerados pelos alvos e a linha azul representa a trajetória filtrada pelo IMM.

i. Cenário 1:

   A figura 6.5 apresenta tanto as trajetórias sugeridas para os alvos contidos no cenário 1 como os falsos acompanhamentos que foram criados pelo método clássico. Uma aproximação das trajetórias sugeridas para os alvos contidos no cenário 1 é apresentada nas figuras 6.6, 6.7 e 6.8.

   A figura 6.9 apresenta tanto as trajetórias sugeridas para os alvos contidos no cenário 1, como os falsos acompanhamentos que foram criados pelo método proposto. Uma aproximação das trajetórias sugeridas para os alvos contidos no cenário 1 é apresentada nas figuras 6.10, 6.11 e 6.12.

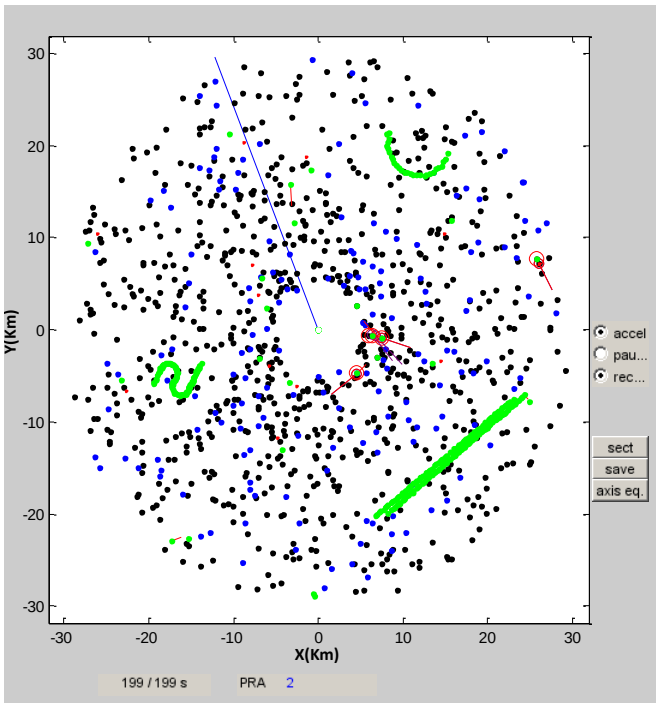


Figura 6.5: Resultados do Cenário 1 pelo método clássico

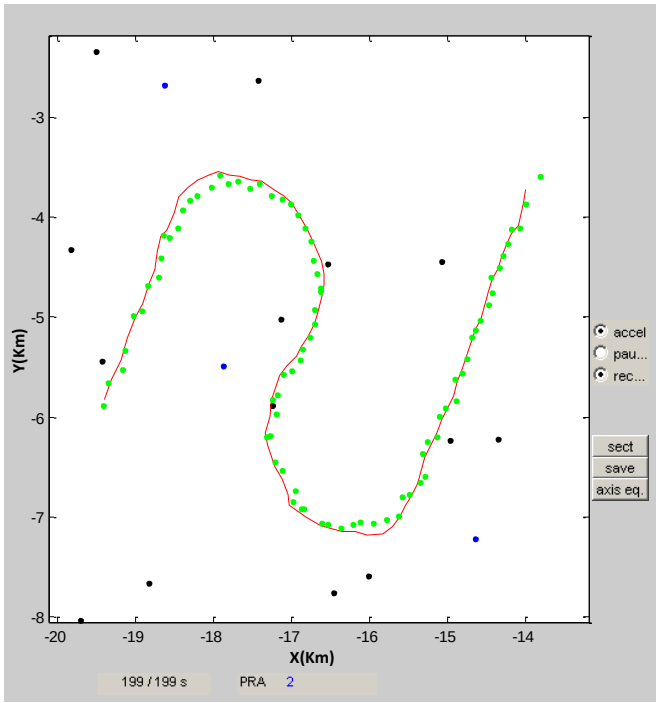


Figura 6.6: Acompanhamento da trajetória 1 feito pelo método clássico no cenário 1

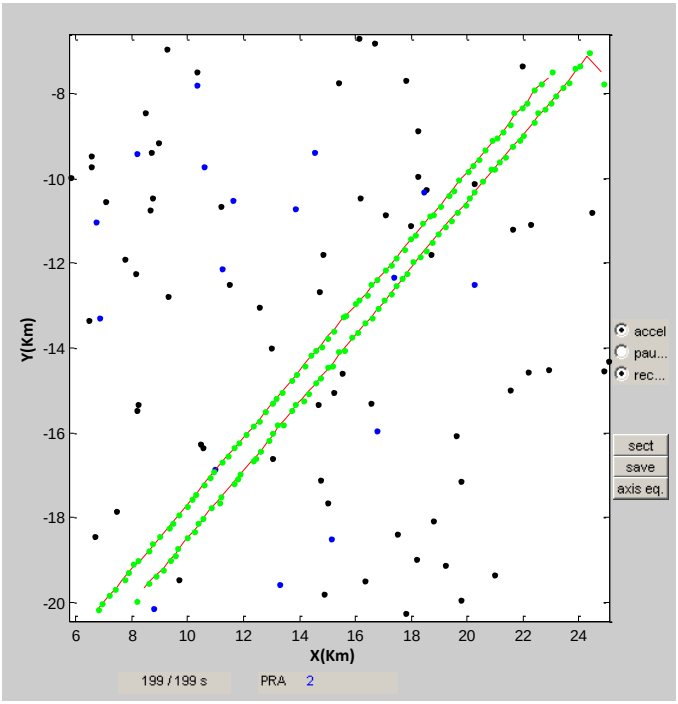


Figura 6.7: Acompanhamentos da trajetória 2 feitos pelo método clássico no cenário 1

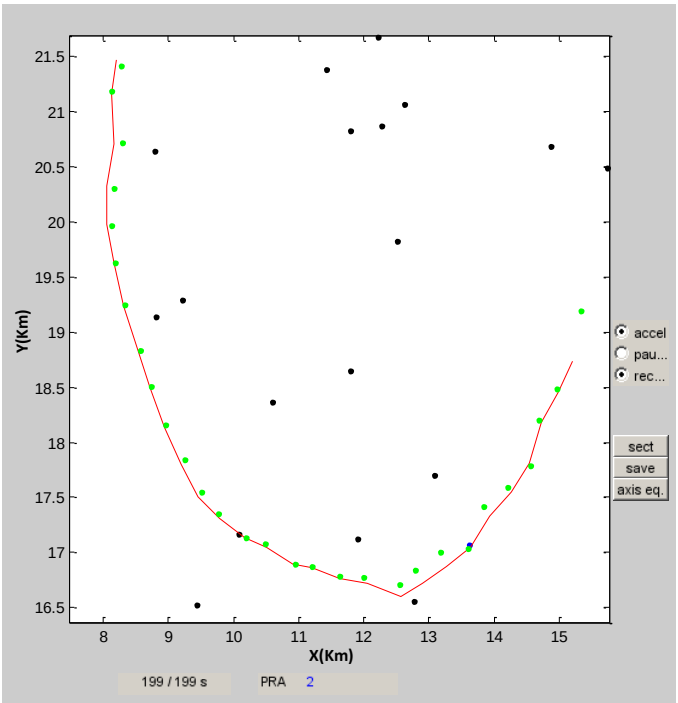


Figura 6.8: Acompanhamento da trajetória 3 feito pelo método clássico no cenário 1

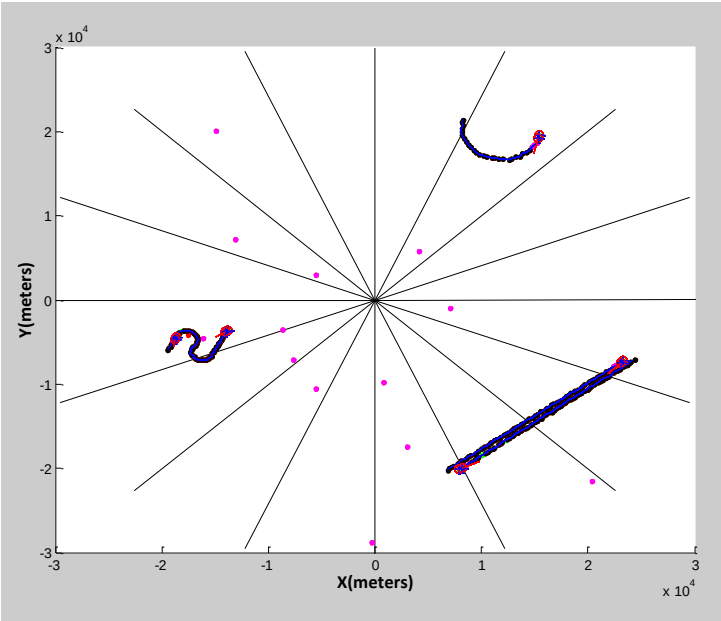


Figura 6.9: Resultados do Cenário 1 pelo método proposto

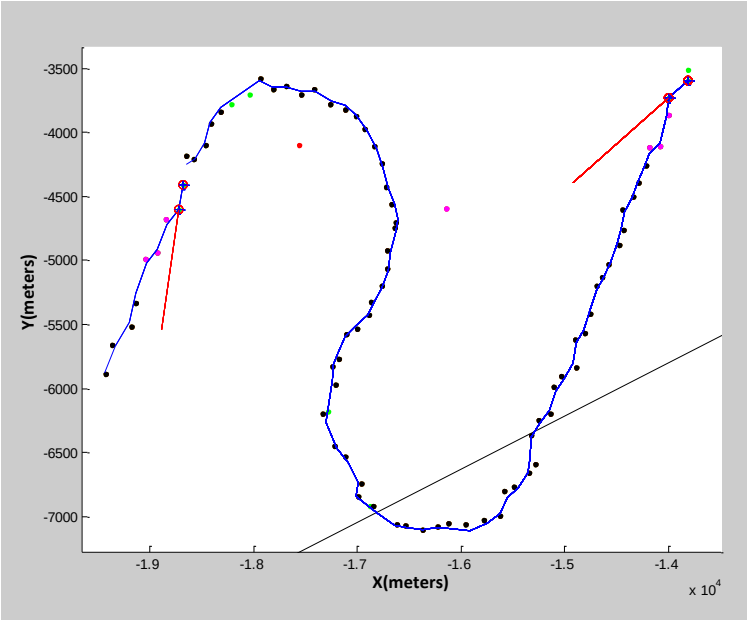


Figura 6.10: Acompanhamento da trajetória 1 feito pelo método proposto no cenário 1

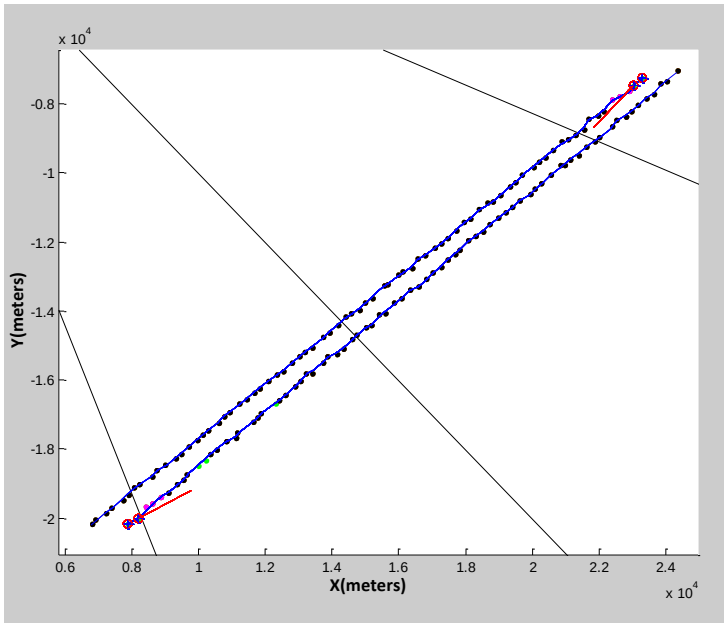


Figura 6.11: Acompanhamentos da trajetória 2 feitos pelo método proposto no cenário 1

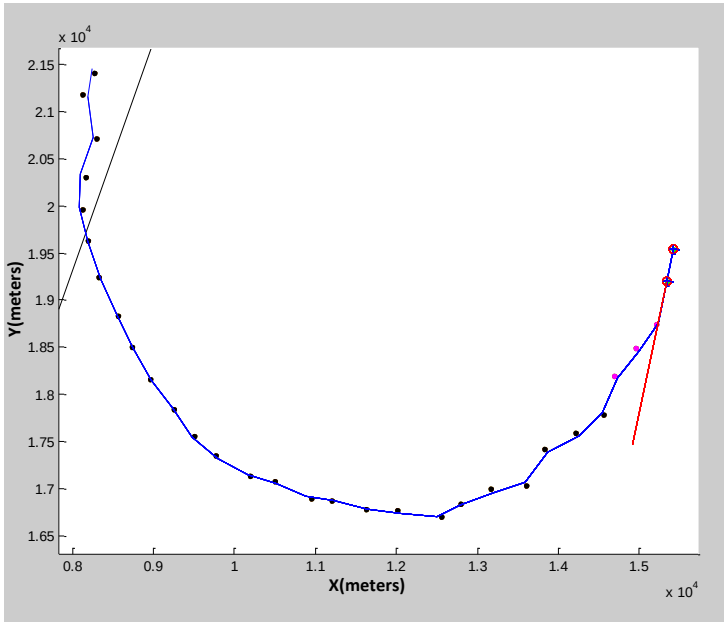


Figura 6.12: Acompanhamento da trajetória 3 feito pelo método proposto no cenário 1

ii. Cenário 2:

A figura 6.13 apresenta tanto as trajetórias sugeridas para os alvos contidos no cenário 2, como os falsos acompanhamentos que foram criados pelo método clássico. Uma ampliação das trajetórias sugeridas para os alvos contidos no cenário 2 é apresentada nas figuras 6.14, 6.15 e 6.16.

A figura 6.17 apresenta tanto as trajetórias sugeridas para os alvos contidos no cenário 2, como os falsos acompanhamentos que foram criados pelo método proposto. Uma ampliação das trajetórias sugeridas para os alvos contidos no cenário 2 é apresentada nas figuras 6.18, 6.19 e 6.20.

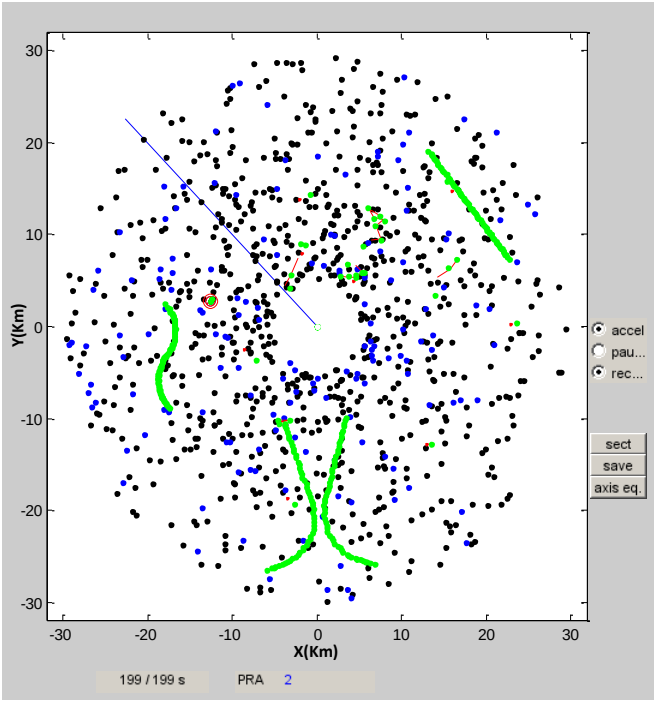


Figura 6.13: Resultados do Cenário 2 pelo método clássico

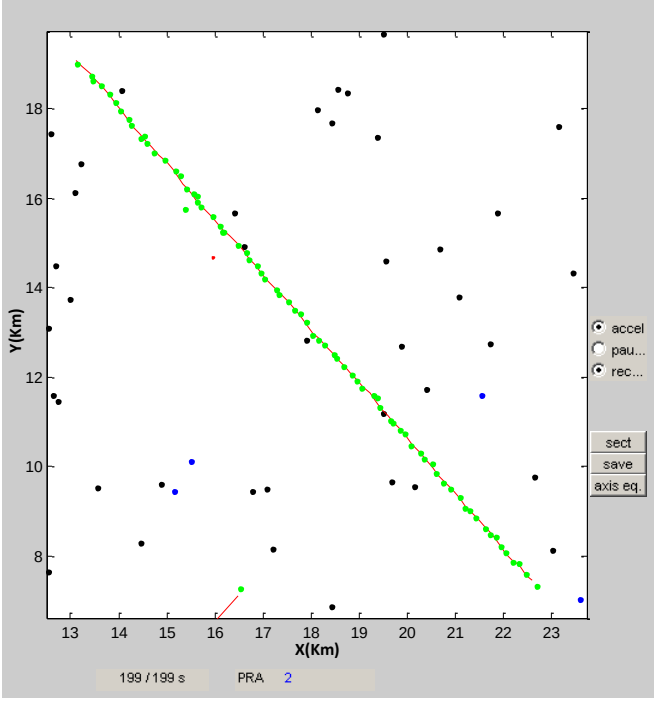


Figura 6.14: Acompanhamento da trajetória 4 feito pelo método clássico no cenário 2

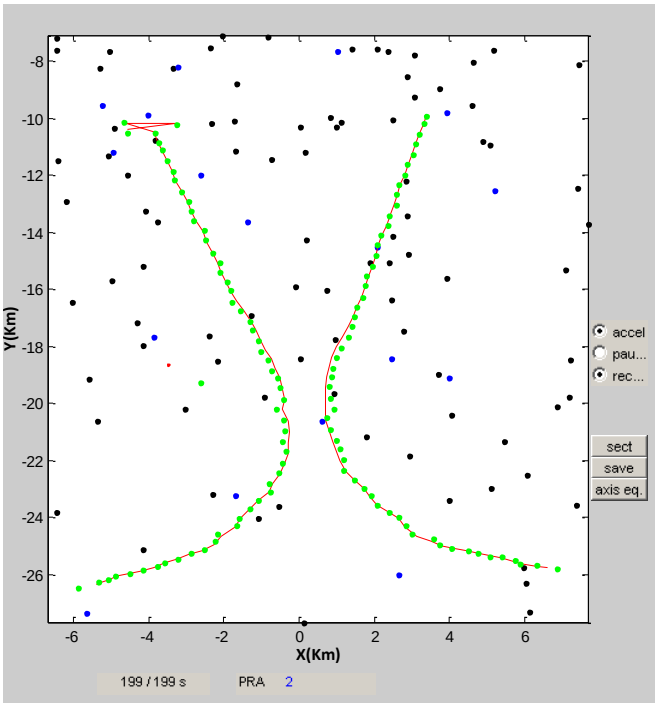


Figura 6.15: Acompanhamentos das trajetórias 5 e 7 feitos pelo método clássico no cenário 2

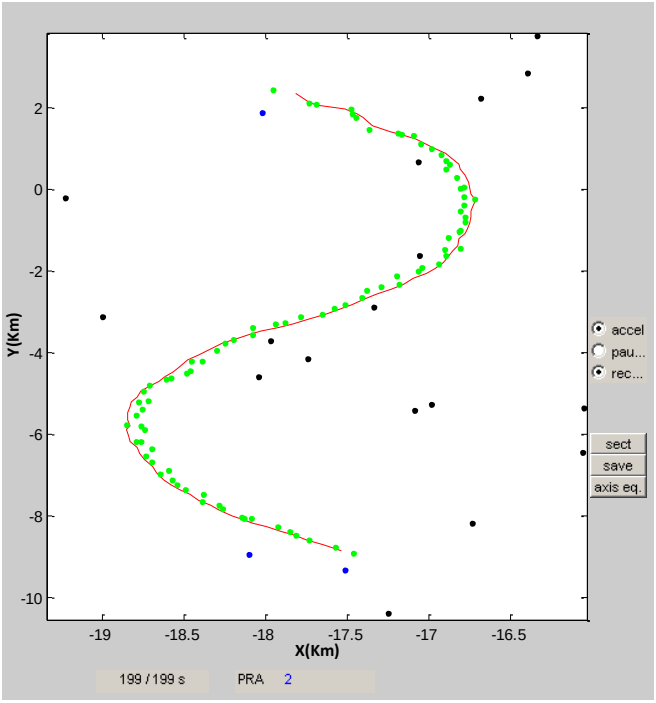


Figura 6.16: Acompanhamento da trajetória 6 feito pelo método clássico no cenário 2

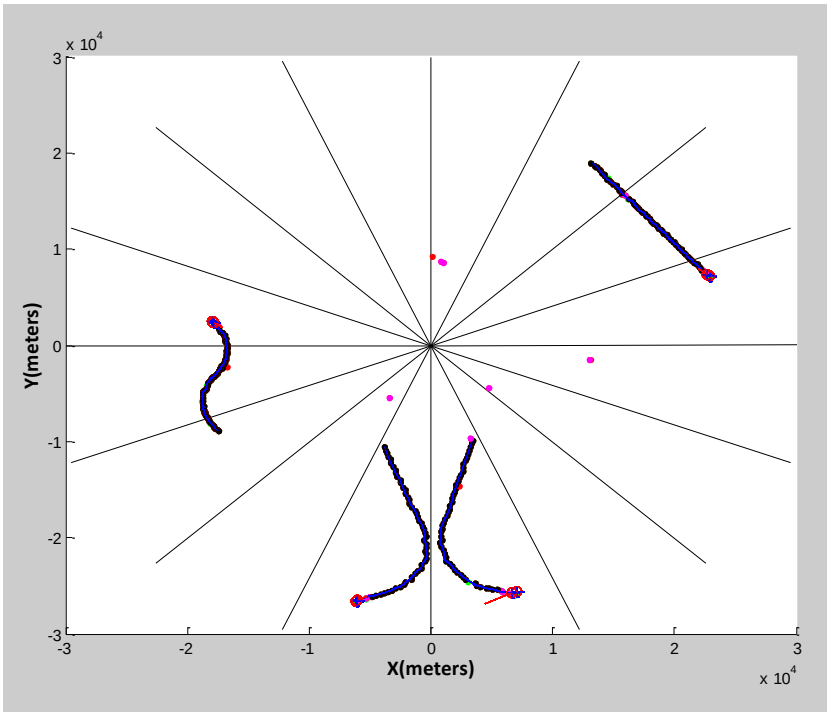


Figura 6.17: Resultados do Cenário 2 pelo método proposto

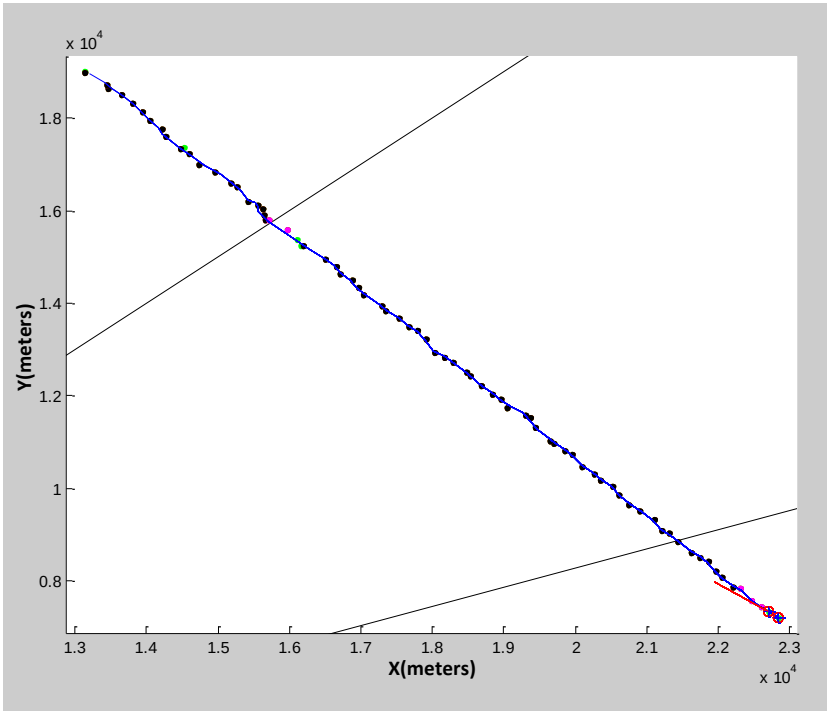


Figura 6.18: Acompanhamento da trajetória 4 feito pelo método proposto no cenário 2

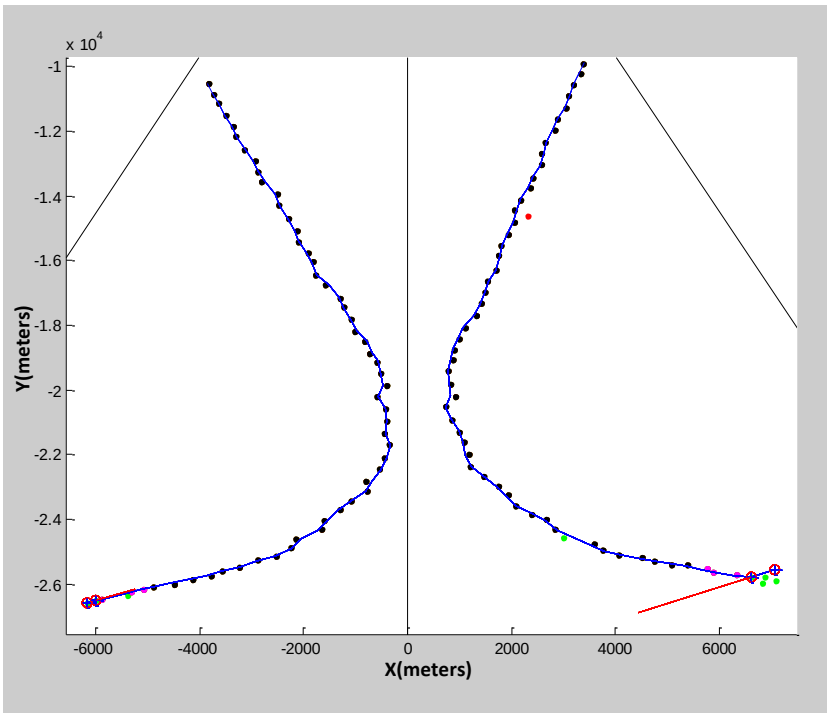


Figura 6.19: Acompanhamentos das trajetórias 5 e 7 feitos pelo método proposto no cenário 2

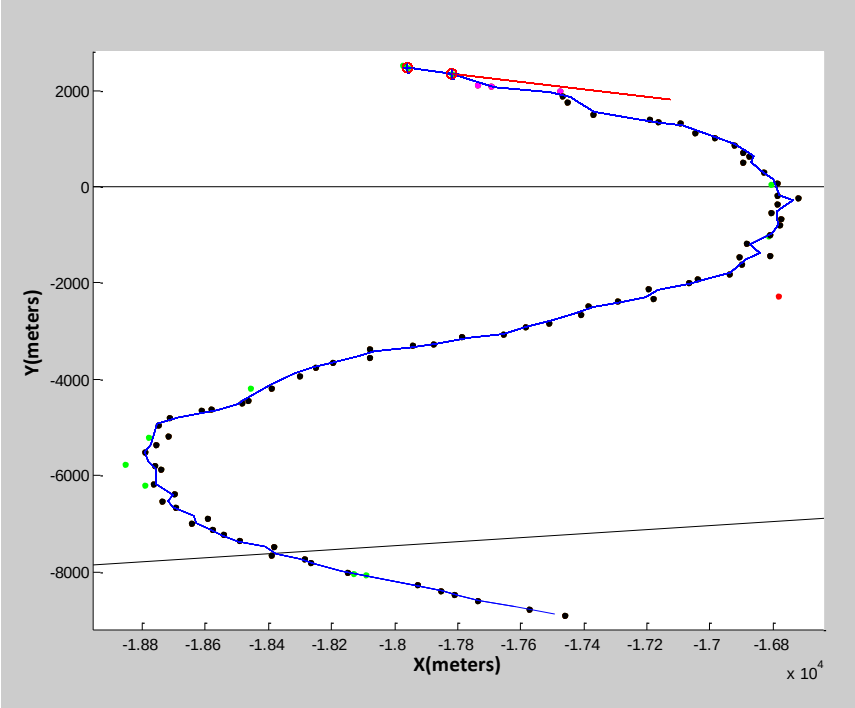


Figura 6.20: Acompanhamento da trajetória 6 feito pelo método proposto no cenário 2

iii. Cenário 3:

A figura 6.21 apresenta tanto as trajetórias sugeridas para os alvos contidos no cenário 3, como os falsos acompanhamentos que foram criados pelo método clássico. Uma ampliação das trajetórias sugeridas para os alvos contidos no cenário 3 é apresentada nas figuras 6.22, 6.23 e 6.24.

A figura 6.25 apresenta tanto as trajetórias sugeridas para os alvos contidos no cenário 3, como os falsos acompanhamentos que foram criados pelo método proposto. Uma ampliação das trajetórias sugeridas para os alvos contidos no cenário 3 é apresentada nas figuras 6.26, 6.27 e 6.28.

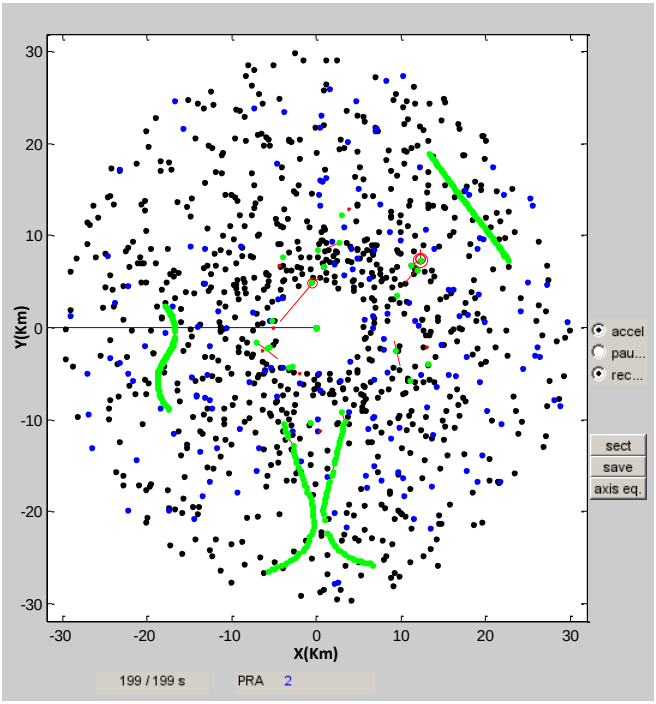


Figura 6.21: Resultados do Cenário 3 pelo método clássico

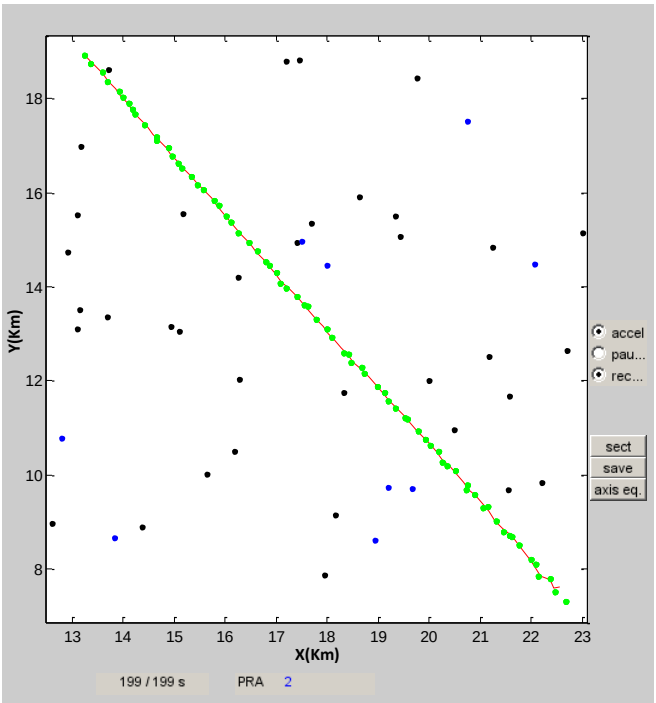


Figura 6.22: Acompanhamento da trajetória 4 feito pelo método clássico no cenário 3

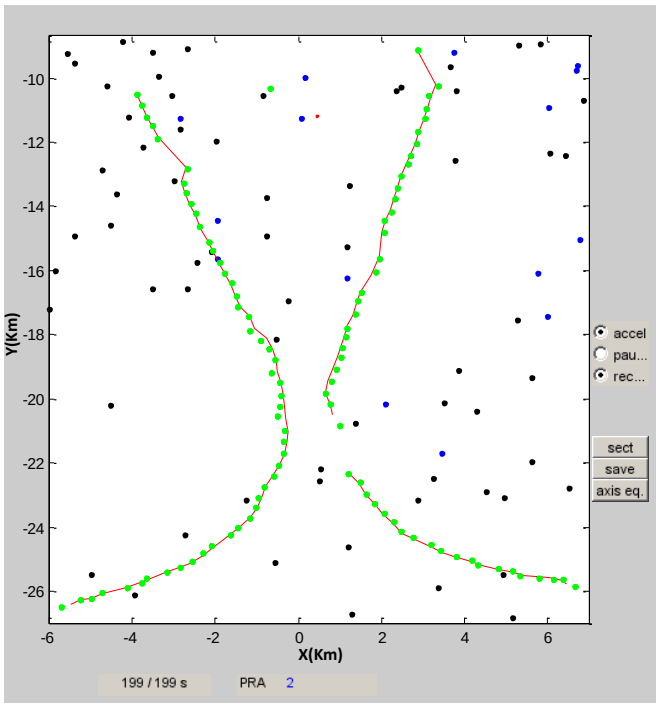


Figura 6.23: Acompanhamentos das trajetórias 5 e 7 feitos pelo método clássico no cenário 3

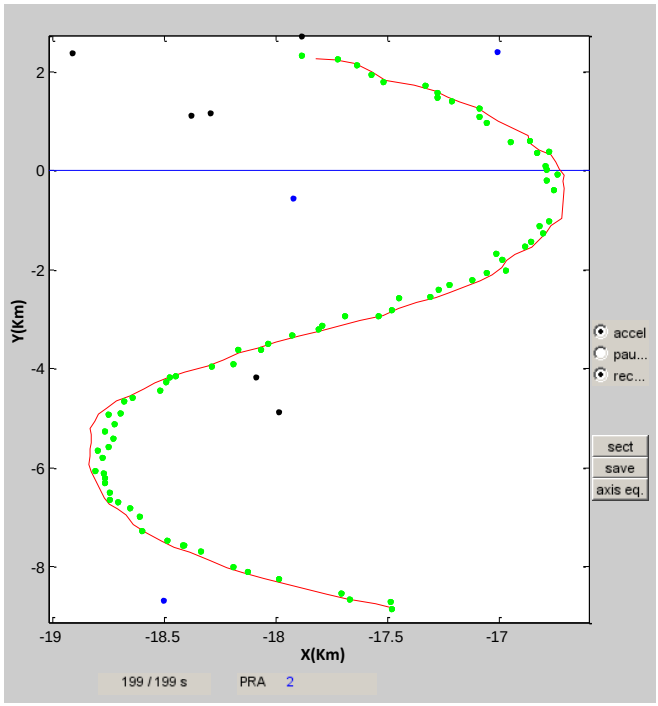


Figura 6.24: Acompanhamento da trajetória 6 feito pelo método clássico no cenário 3

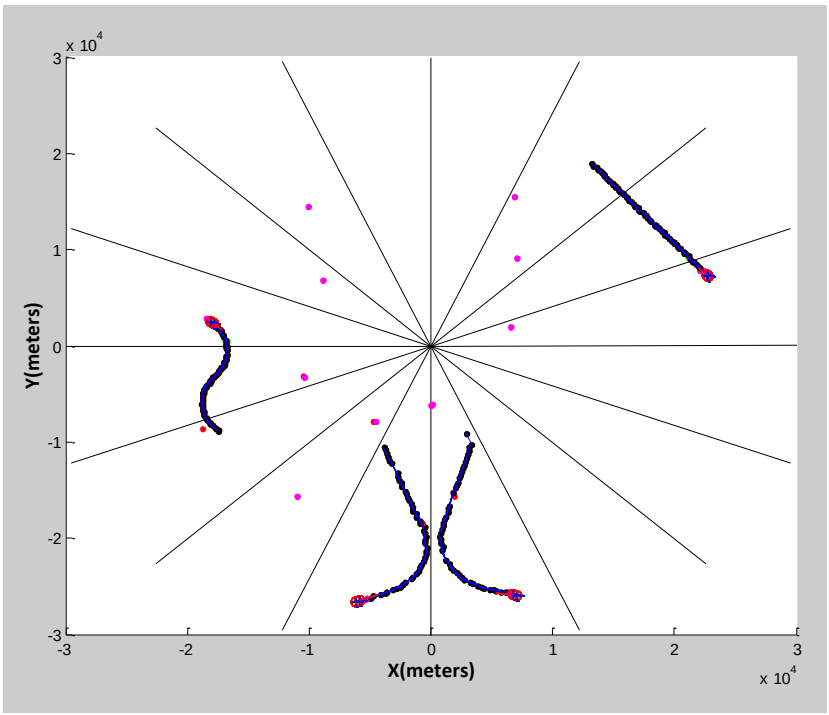


Figura 6.25: Resultados do Cenário 3 pelo método proposto

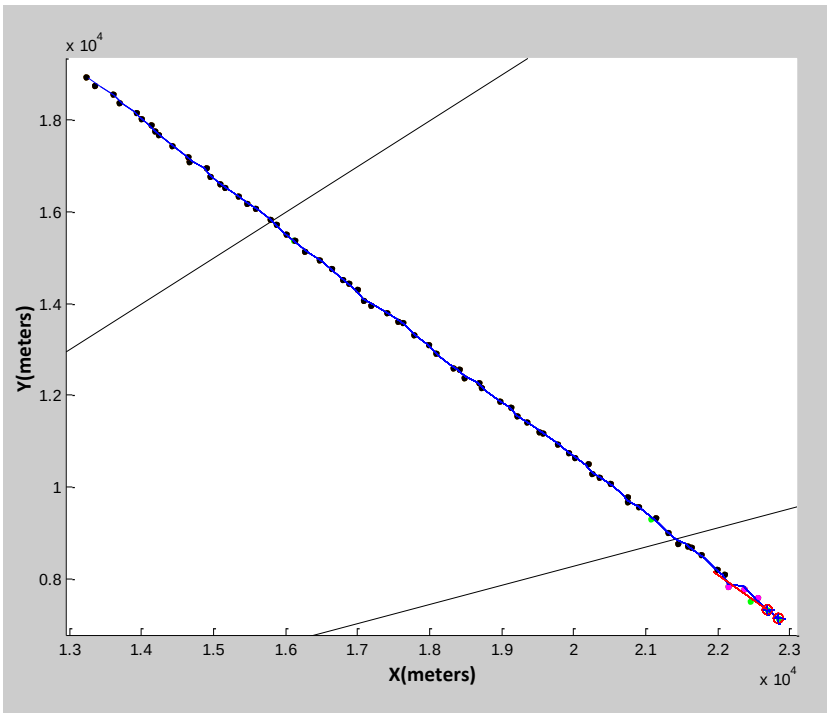


Figura 6.26: Acompanhamento da trajetória 4 feito pelo método proposto no cenário 3

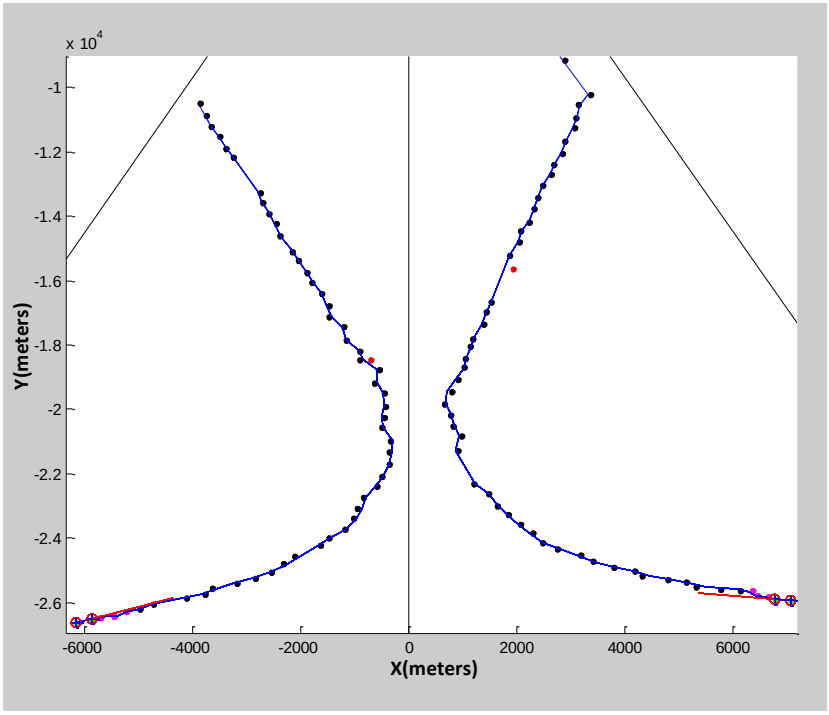


Figura 6.27: Acompanhamentos das trajetórias 5 e 7 feitos pelo método proposto no cenário 3

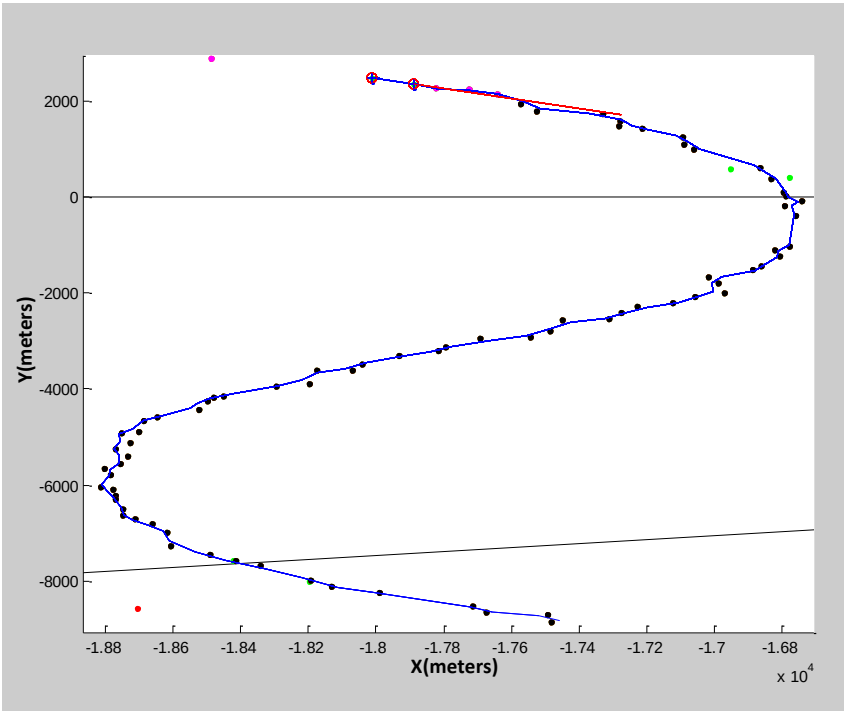


Figura 6.28: Acompanhamento da trajetória 6 feito pelo método proposto no cenário 3

iv. Cenário 4:

A figura 6.29 apresenta tanto as trajetórias sugeridas para os alvos contidos no cenário 4, como os falsos acompanhamentos que foram criados pelo método clássico. Uma ampliação das trajetórias sugeridas para os alvos contidos no cenário 4 é apresentada nas figuras 6.30, 6.31 e 6.32

A figura 6.33 apresenta tanto as trajetórias sugeridas para os alvos contidos no cenário 4, como os falsos acompanhamentos que foram criados pelo método proposto. Uma ampliação das trajetórias sugeridas para os alvos contidos no cenário 4 é apresentada nas figuras 6.34, 6.35 e 6.36.

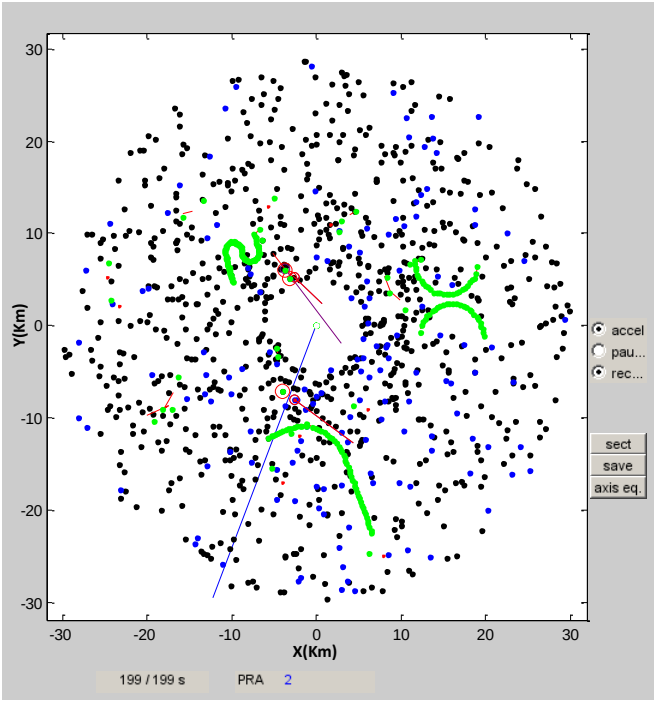


Figura 6.29: Resultados do Cenário 4 pelo método clássico

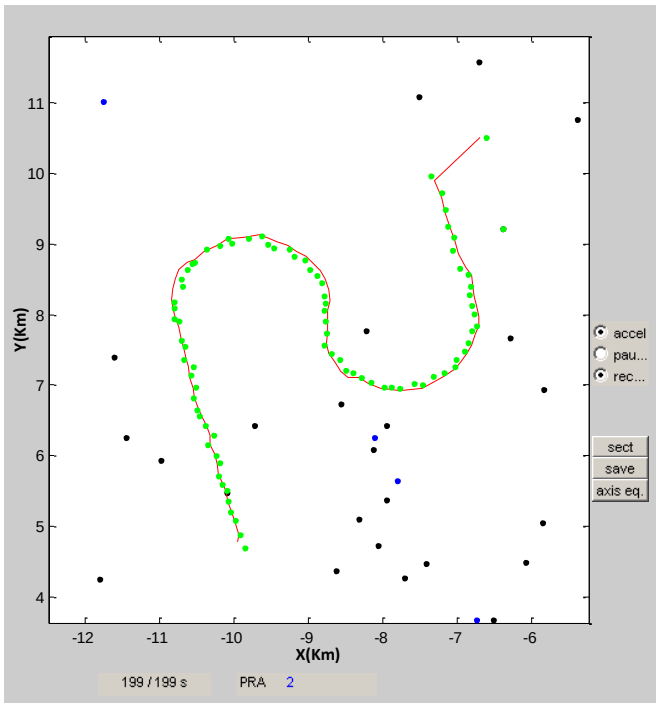


Figura 6.30: Acompanhamento da trajetória 1 feito pelo método clássico no cenário 4

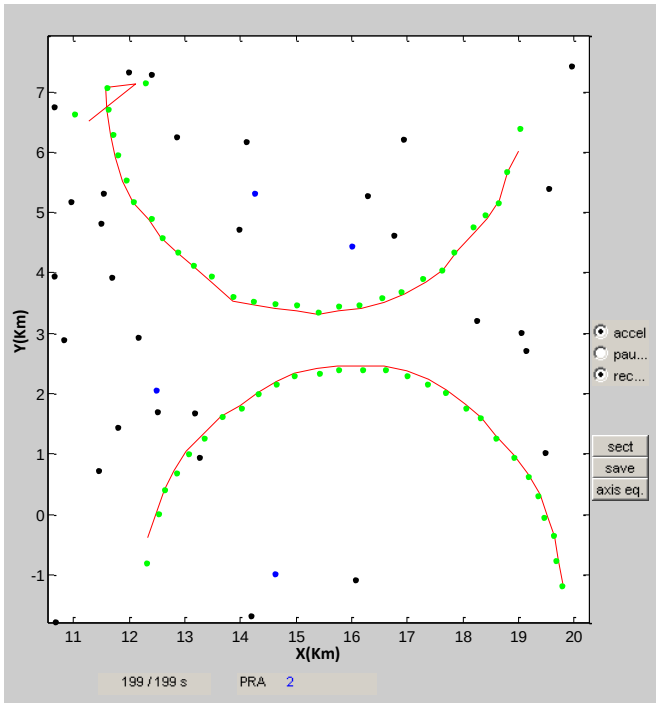


Figura 6.31: Acompanhamentos da trajetória 3 feitos pelo método clássico no cenário 4

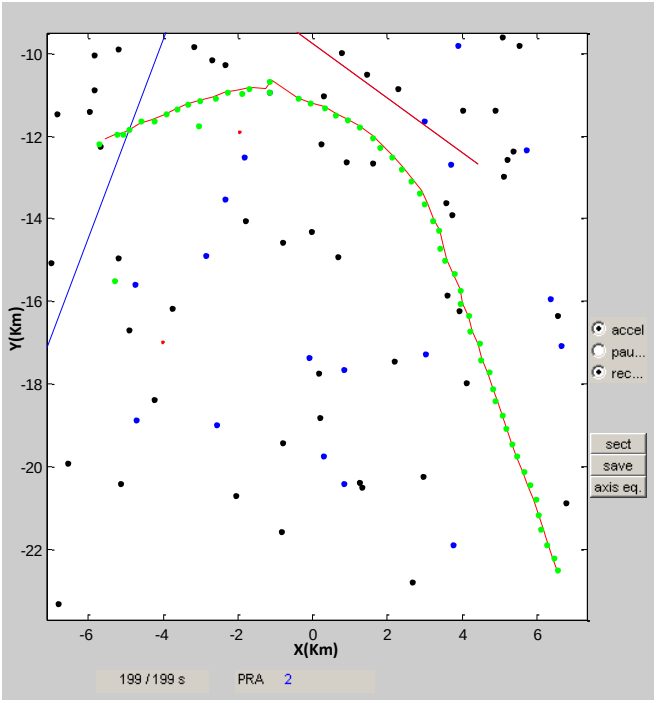


Figura 6.32: Acompanhamento da trajetória 5 feito pelo método clássico no cenário 4

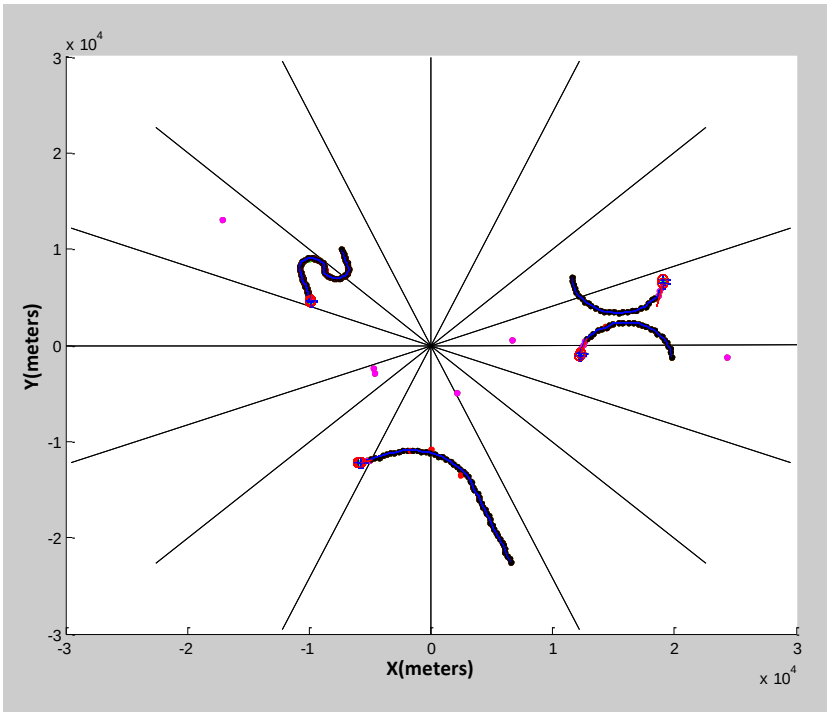


Figura 6.33: Resultados do Cenário 4 pelo método proposto

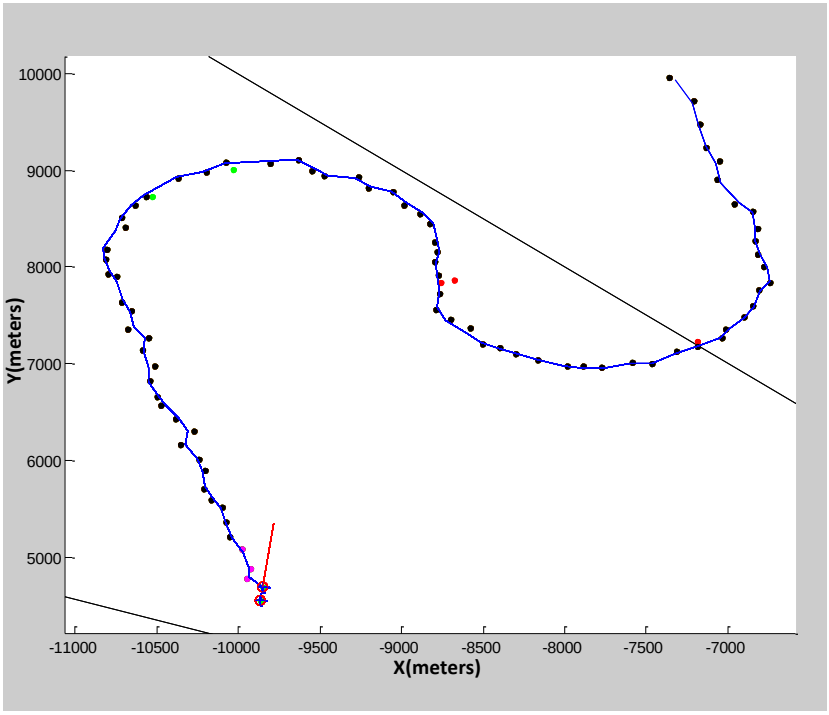


Figura 6.34: Acompanhamento da trajetória 1 feito pelo método proposto no cenário 4

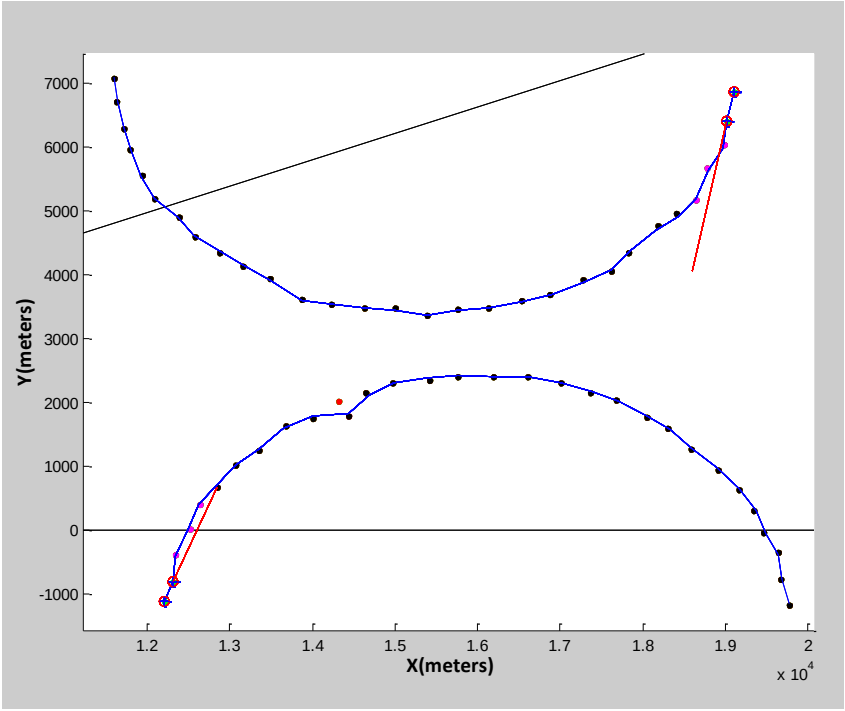


Figura 6.35: Acompanhamentos da trajetória 3 feitos pelo método proposto no cenário 4

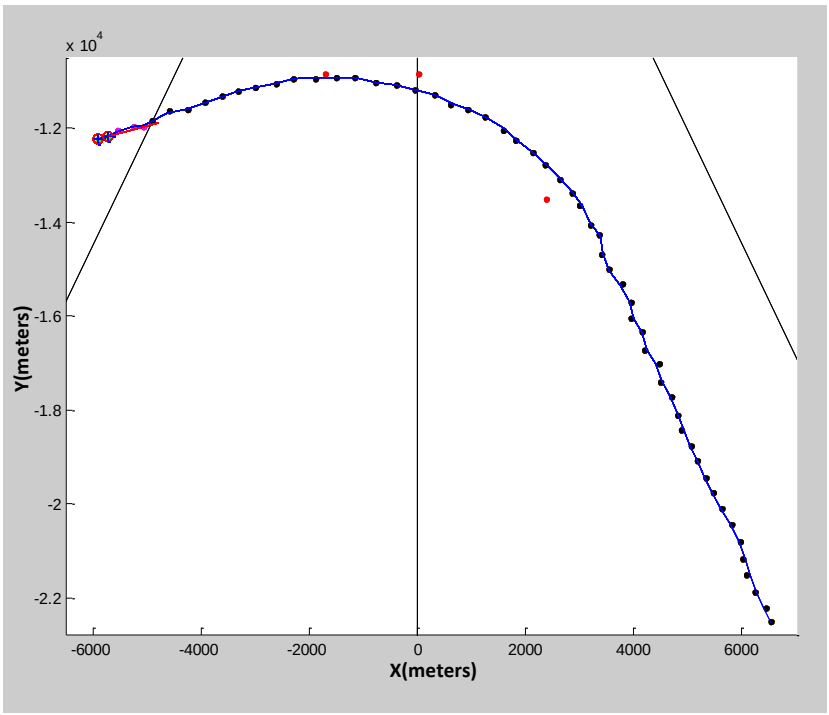


Figura 6.36: Acompanhamento da trajetória 5 feito pelo método proposto no cenário 4

v. Cenário 5:

A figura 6.37 apresenta tanto as trajetórias sugeridas para os alvos contidos no cenário 5, como os falsos acompanhamentos que foram criados pelo método clássico. Uma ampliação das trajetórias sugeridas para os alvos contidos no cenário 5 é apresentada nas figuras 6.38, 6.39 e 6.38.

A figura 6.41 apresenta tanto as trajetórias sugeridas para os alvos contidos no cenário 5, como os falsos acompanhamentos que foram criados pelo método proposto. Uma ampliação das trajetórias sugeridas para os alvos contidos no cenário 5 é apresentada nas figuras 6.42, 6.43 e 6.44.

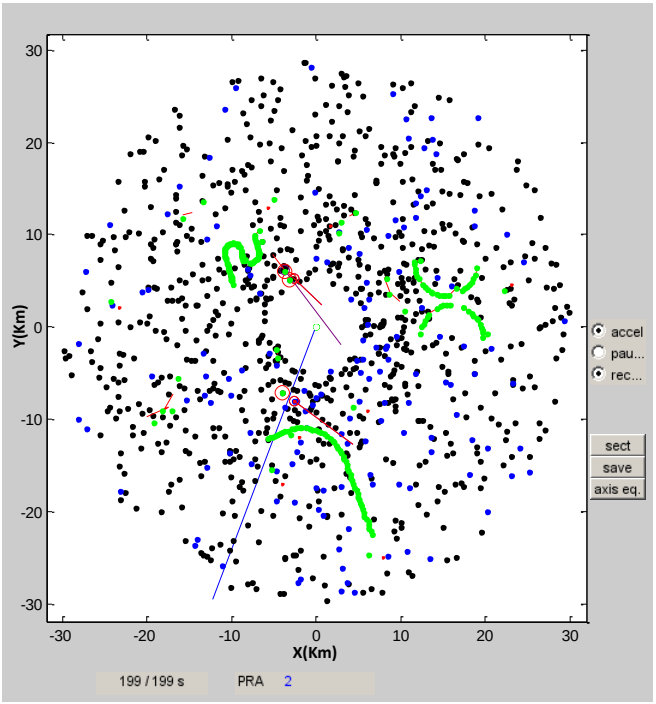


Figura 6.37: Resultados do Cenário 5 pelo método clássico

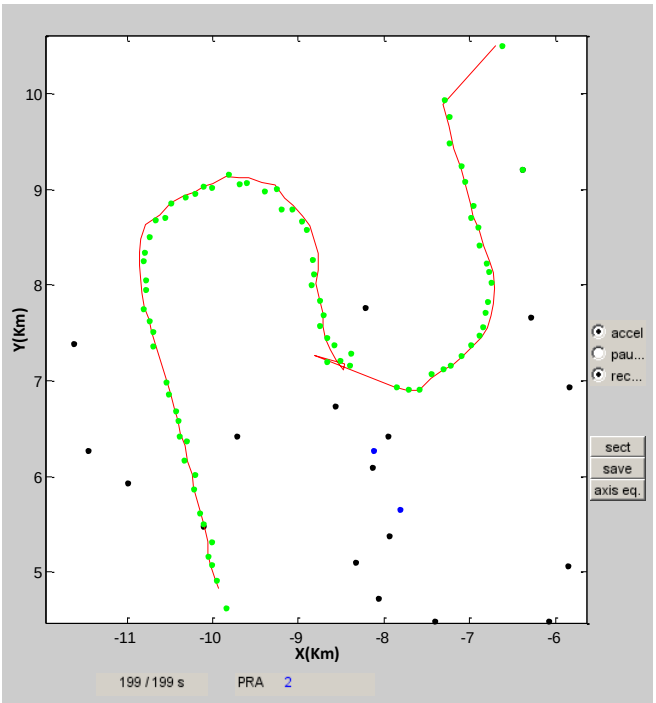


Figura 6.38: Acompanhamento da trajetória 1 feito pelo método clássico no cenário 5

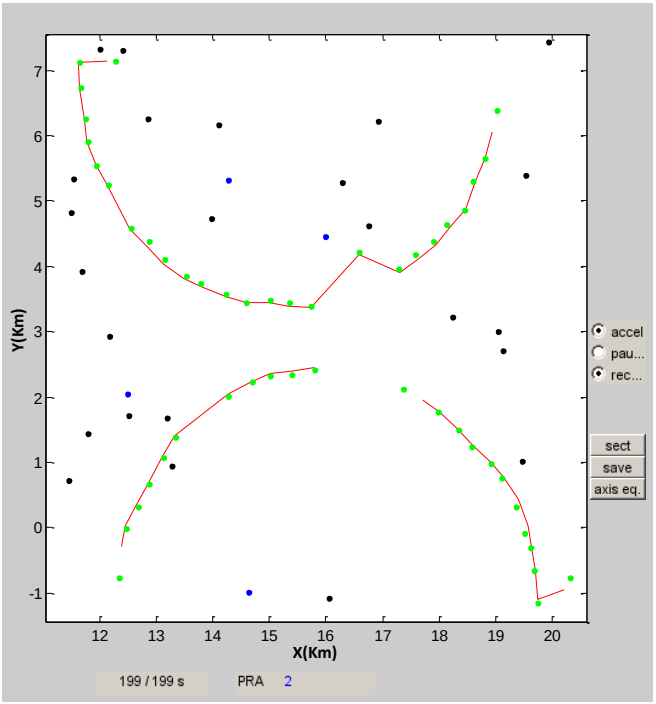


Figura 6.39: Acompanhamentos da trajetória 3 feitos pelo método clássico no cenário 5

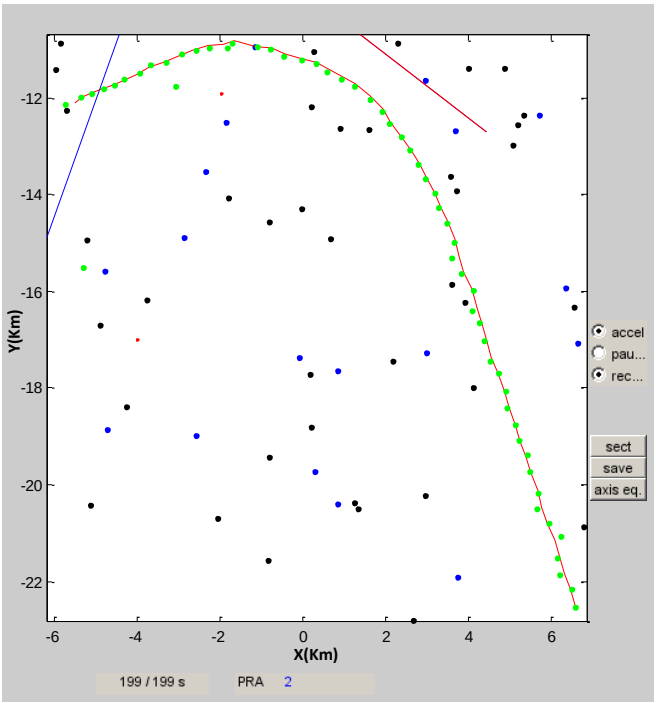


Figura 6.40: Acompanhamento da trajetória 5 feito pelo método clássico no cenário 5

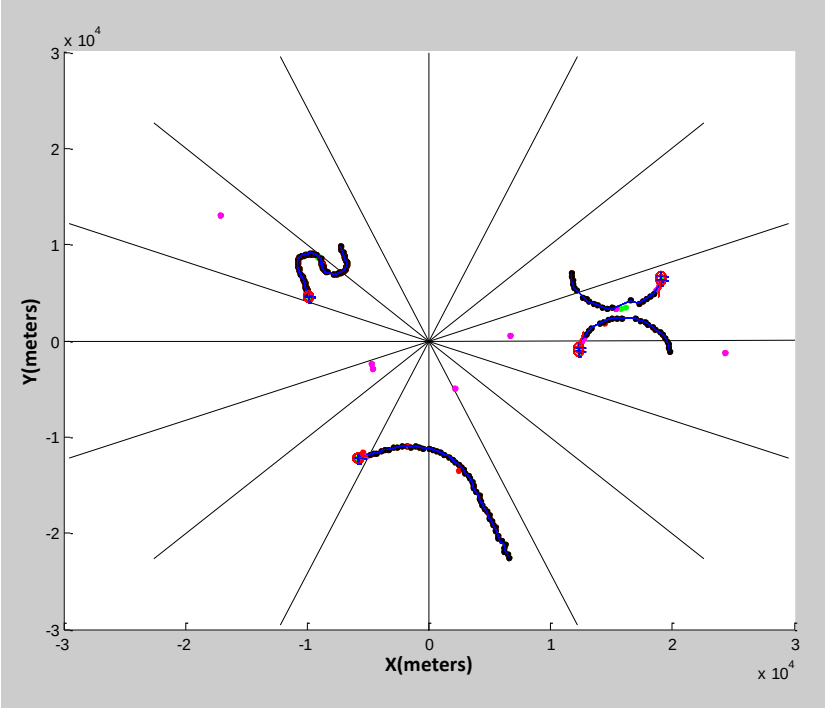


Figura 6.41: Resultados do Cenário 5 pelo método proposto

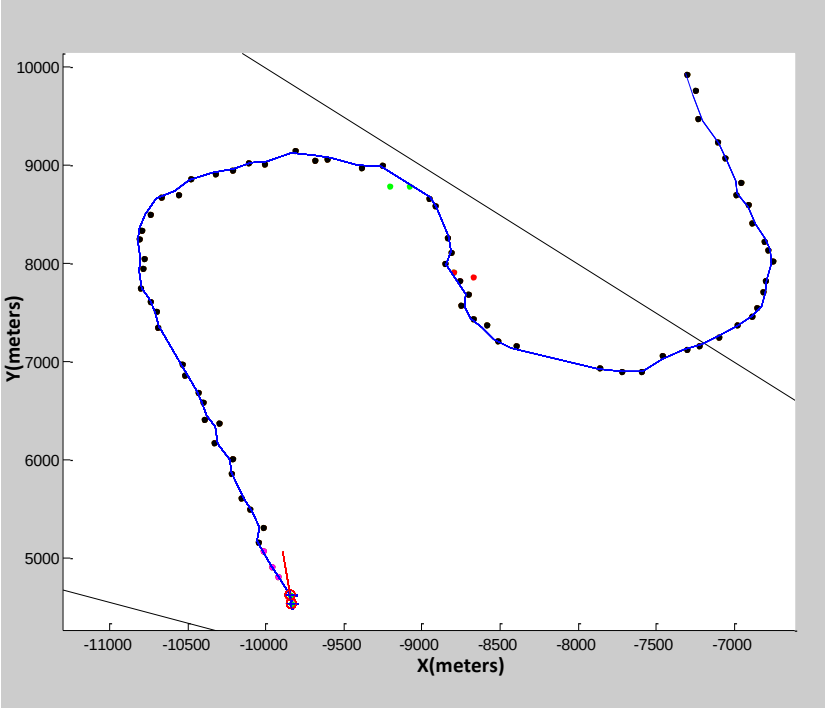


Figura 6.42: Acompanhamento da trajetória 1 feito pelo método proposto no cenário 5

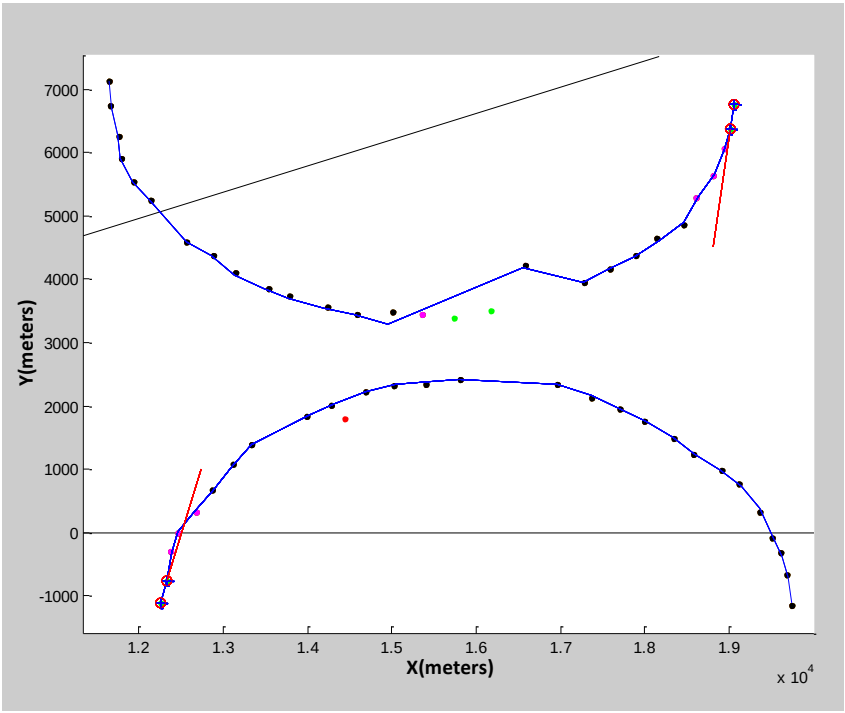


Figura 6.43: Acompanhamentos da trajetória 3 feitos pelo método proposto no cenário 5

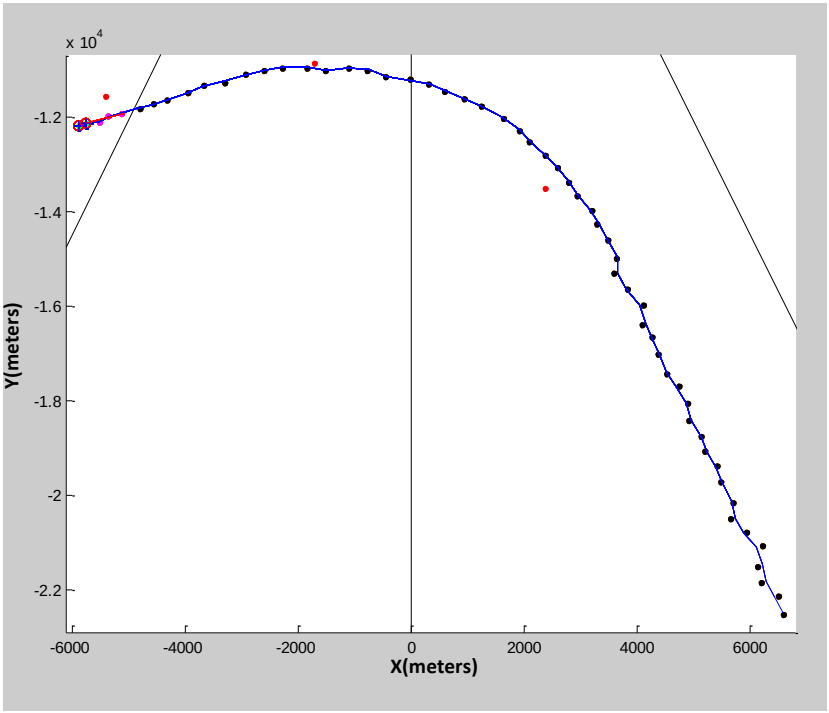


Figura 6.44: Acompanhamento da trajetória 5 feito pelo método proposto no cenário 5

vi. Cenário 6:

A figura 6.45 apresenta tanto as trajetórias sugeridas para os alvos contidos no cenário 6, como os falsos acompanhamentos que foram criados pelo método clássico. Uma ampliação das trajetórias sugeridas para os alvos contidos no cenário 6 é apresentada nas figuras 6.46, 6.47 e 6.48.

A figura 6.49 apresenta tanto as trajetórias sugeridas para os alvos contidos no cenário 6, como os falsos acompanhamentos que foram criados pelo método proposto. Uma ampliação das trajetórias sugeridas para os alvos contidos no cenário 6 é apresentada nas figuras 6.50, 6.51 e 6.52.

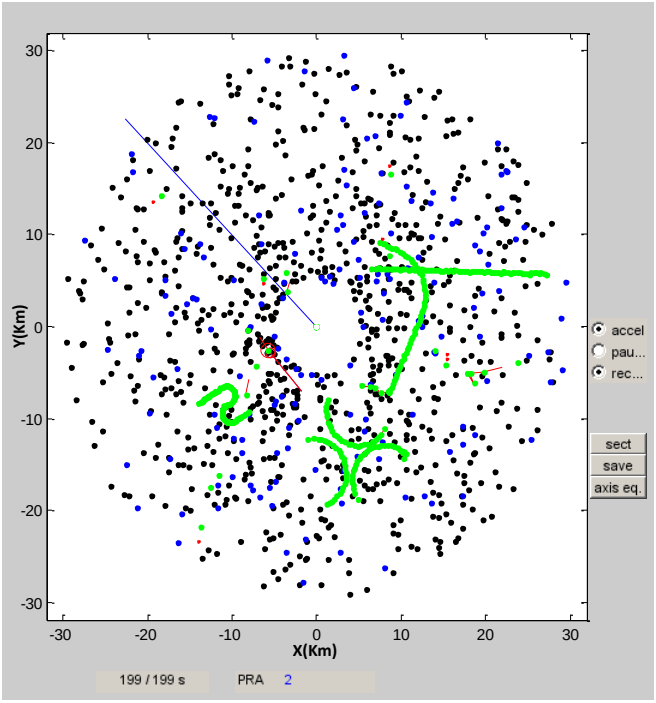


Figura 6.45: Resultados do Cenário 6 pelo método clássico

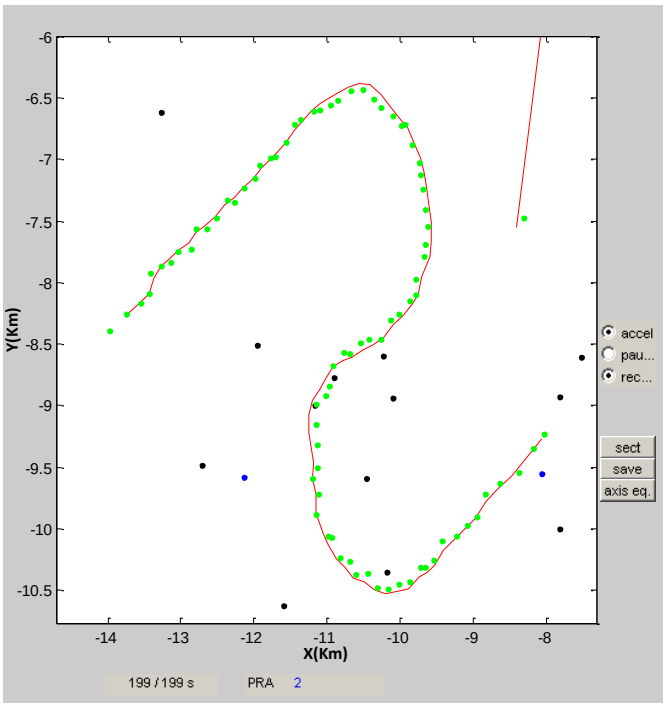


Figura 6.46: Acompanhamento da trajetória 1 feito pelo método clássico no cenário 6

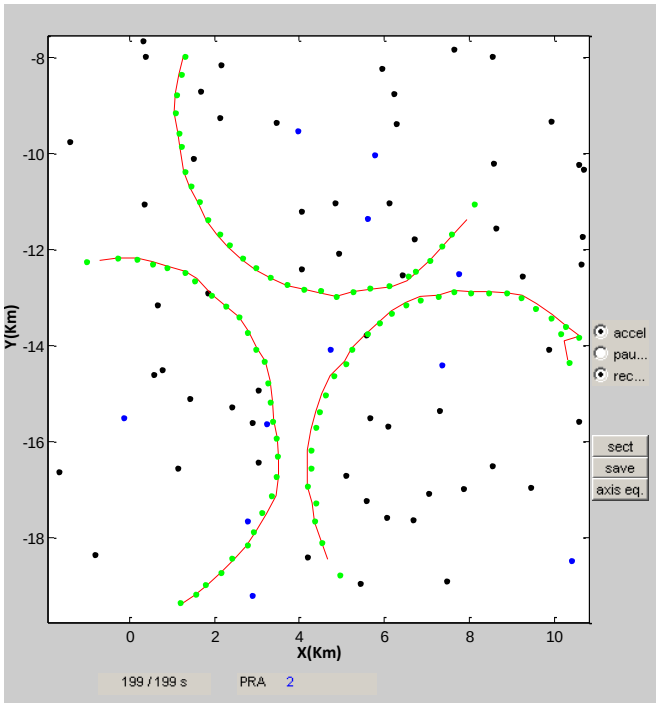


Figura 6.47: Acompanhamentos da trajetória 3 feitos pelo método clássico no cenário 6

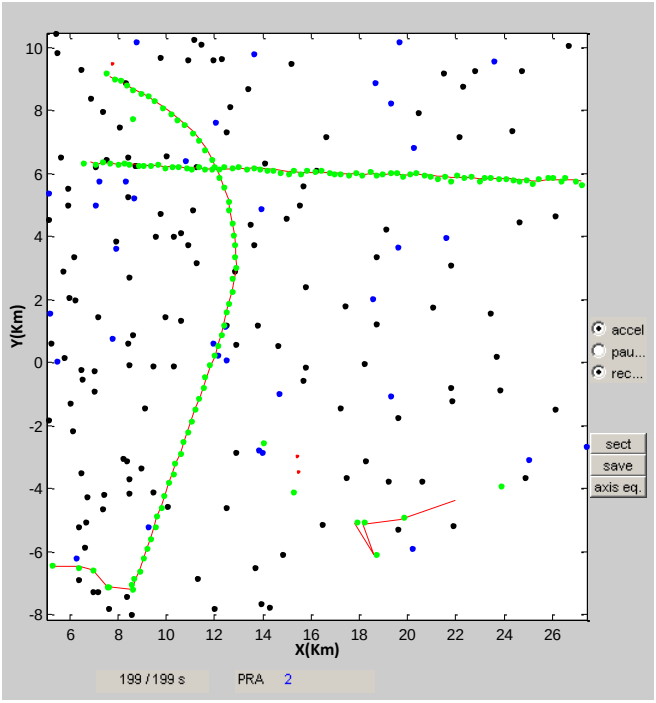


Figura 6.48: Acompanhamento das trajetórias 2 e 5 pelo método clássico no cenário 6

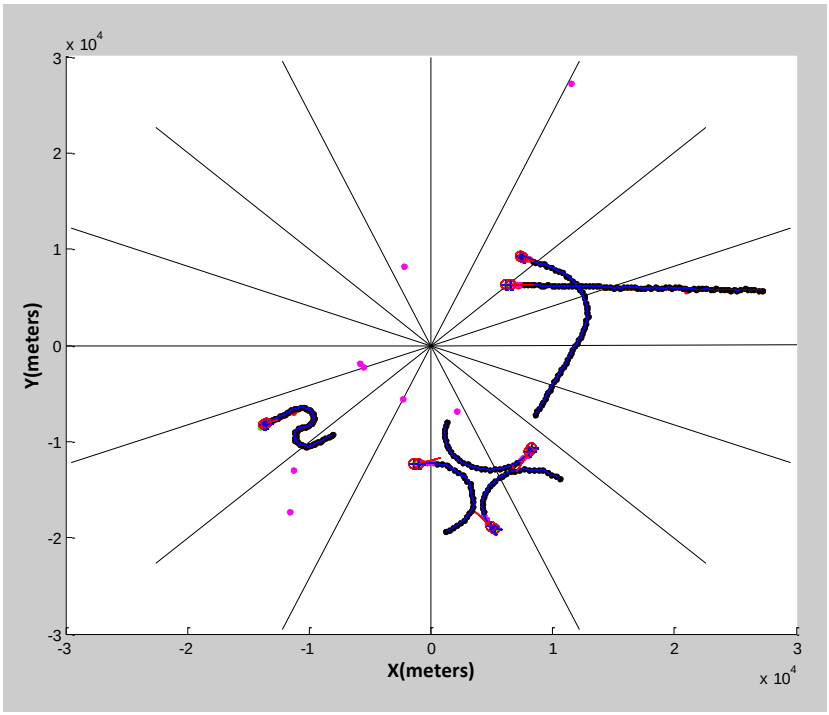


Figura 6.49: Resultados do Cenário 6 pelo método proposto

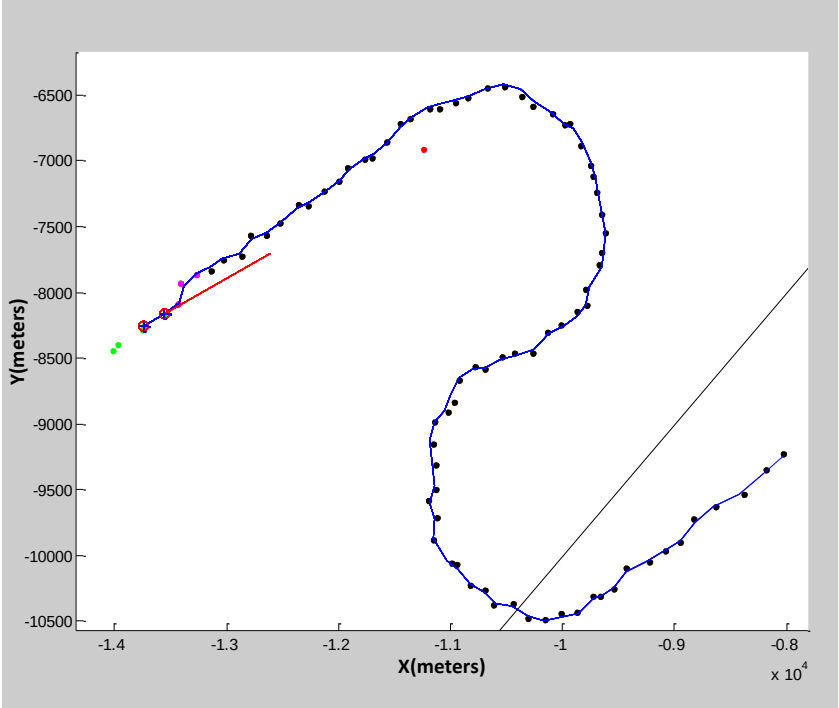


Figura 6.50: Acompanhamento da trajetória 1 feito pelo método proposto no cenário 6

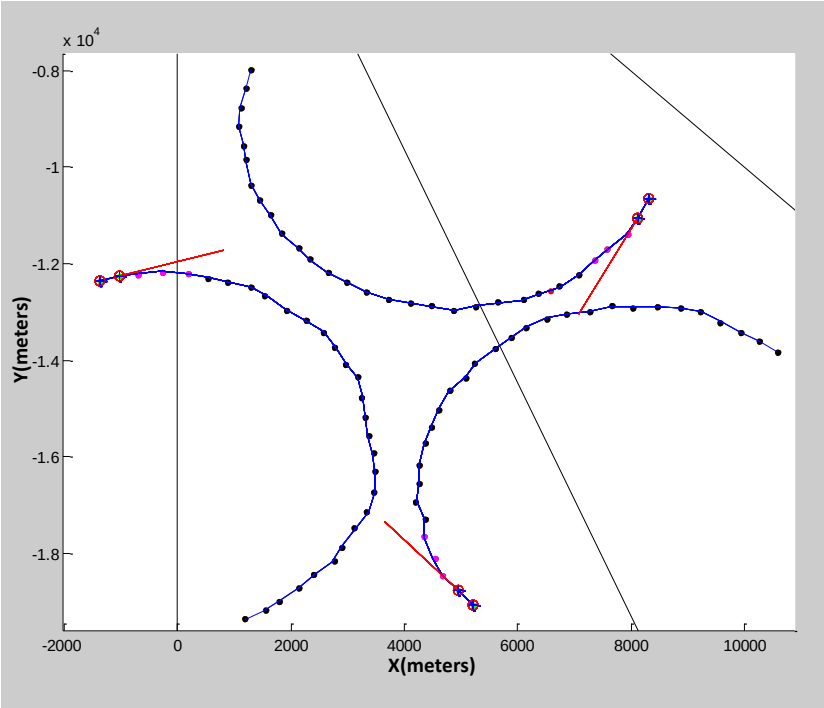


Figura 6.51: Acompanhamentos da trajetória 3 feitos pelo método proposto no cenário 6

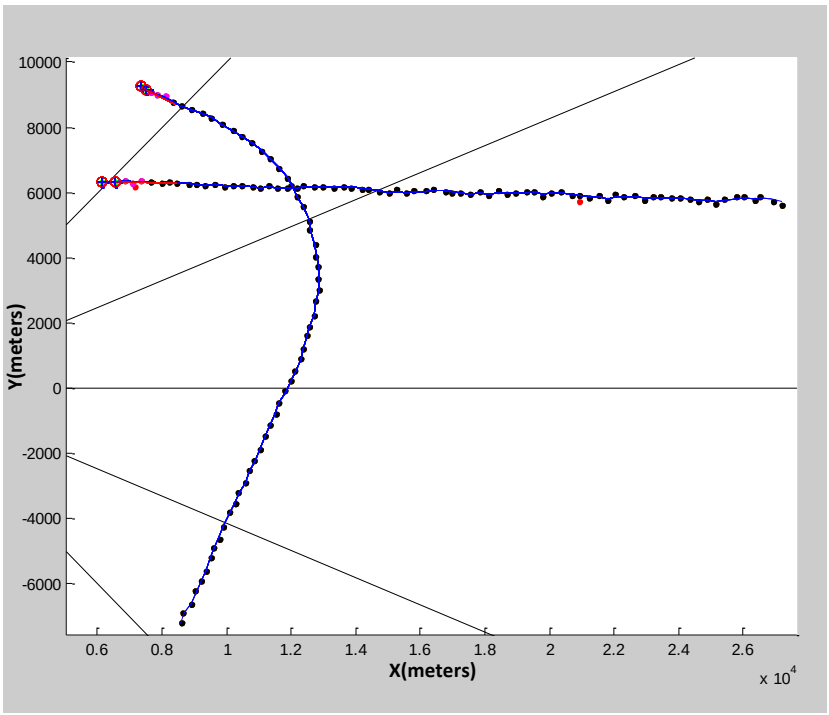


Figura 6.52: Acompanhamento das trajetórias 2 e 5 pelo método proposto no cenário 6

Nas tabelas abaixo são apresentadas comparações dos resultados obtidos a partir da execução dos dois métodos para cada um dos cenários gerados, levando em consideração o número de acertos, número de perdas de detecção, associações com clutter dos acompanhamentos que representam os alvos contidos neste cenário, taxa de acerto, tempo medio da Iteração e número de acompanhamentos confirmados pelos algoritmos.

Tabela 6.4: Resultados Cenário 1

Cenário 1		
	Método Clássico	Método Proposto
Número de Contatos gerados pelos alvos	266	266
Número de acertos	266	258
Perdas de detecção	0	8
Clutter's associados aos acompanhamentos	1	0
Taxa de acerto (%)	99.63	96.99
Tempo medio da Iteração (seg)	1.96281	0.14504
Número de acompanhamentos confirmados	6	5

Tabela 6.5: Resultados Cenário 2

Cenário 2		
	Método Clássico	Método Proposto
Número de Contatos gerados pelos alvos	281	281
Número de acertos	279	262
Perdas de detecção	2	18
Clutter's associados aos acompanhamentos	2	1
Taxa de acerto (%)	98.59	93.24
Tempo medio da Iteração (seg)	2.21144	0.15713
Número de acompanhamentos confirmados	8	4

Tabela 6.6: Resultados Cenário 3

Cenário 3		
	Método Clássico	Método Proposto
Número de Contatos gerados pelos alvos	265	265
Número de acertos	261	256
Perdas de detecção	4	8
Clutter's associados aos acompanhamentos	4	2
Taxa de acerto (%)	97.03	96.24
Tempo medio da Iteração (seg)	2.00664	0.16002
Número de acompanhamentos confirmados	10	4

Tabela 6.7: Resultados Cenario 4

Cenário 4		
	Método Clássico	Método Proposto
Número de Contatos gerados pelos alvos	201	201
Número de acertos	199	198
Perdas de detecção	0	2
Clutter's associados aos acompanhamentos	4	1
Taxa de acerto (%)	98.03	98.51
Tempo medio da Iteração (seg)	1.79256	0.11164
Número de acompanhamentos confirmados	8	4

Tabela 6.8: Resultados Cenário 5

Cenário 5		
	Método Clássico	Método Proposto
Número de Contatos gerados pelos alvos	188	188
Número de acertos	184	182
Perdas de detecção	2	5
Clutter's associados aos acompanhamentos	8	2
Taxa de acerto (%)	94.85	96.29
Tempo medio da Iteração (seg)	1.8421	0.10776
Número de acompanhamentos confirmados	11	4

Tabela 6.9: Resultados Cenário 6

Cenário 6		
	Método Clássico	Método Proposto
Número de Contatos gerados pelos alvos	308	308
Número de acertos	307	305
Perdas de detecção	0	3
Clutter's associados aos acompanhamentos	4	0
Taxa de acerto (%)	98.71	99.03
Tempo medio da Iteração (seg)	2.20885	0.174040
Número de acompanhamentos confirmados	9	6

6.2.1 Resultados da Filtragem IMM

Nesta seção foram selecionadas algumas das trajetórias sugeridas nos diferentes cenários pelo método proposto com o propósito de analisar a filtragem feita pelo algoritmo IMM. Assim teremos que:

- Trajetória 1 sugerida no cenário 4:

  A figura 6.53 mostra a mesma trajetória apresentada na figura 6.34 mas é adicionada a trajetória real do alvo (5.69) representada de cor vermelha. Lembrando que nesta figura a linha azul representa a trajetória filtrada pelo filtro IMM e os pontos pretos representam a trajetória sugerida pelo método proposto.

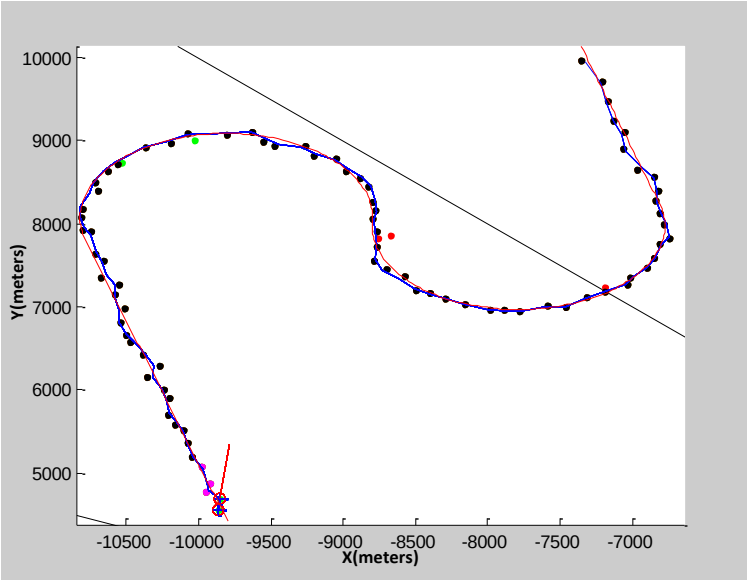


Figura 6.53: Trajetória 1 sugerida no cenário 4

A figura 6.54 apresenta os erros de posição nos eixos x e y em relação à posição real do alvo. Na cor azul é apresentado o erro da trajetória filtrada e na cor vermelha o erro da trajetória sugerida pelo método proposto.

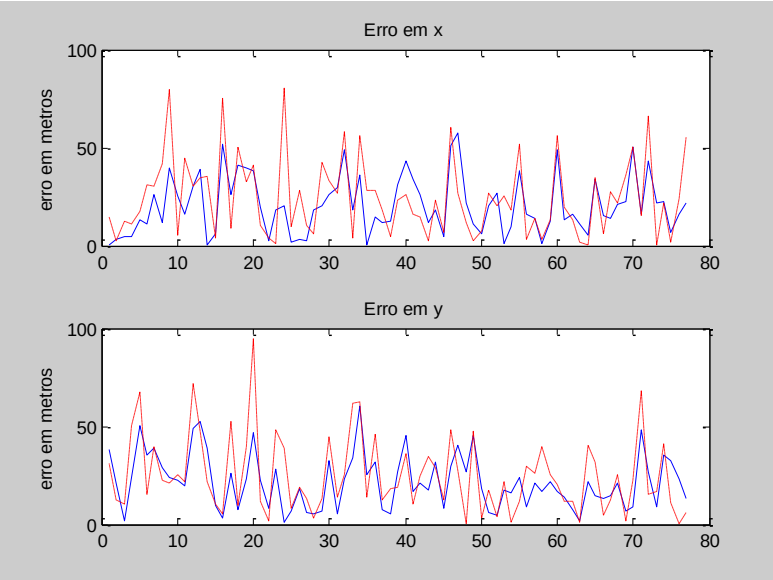


Figura 6.54: Erro de posição nos eixos x e y cometido pela trajetória 1 sugerida no cenário 4, em relação à posição real do alvo.

- Trajetória 2 sugerida no cenário 1:

  No cenário 1 a trajetória 2 (5.70) é utilizada duas vezes, assim, a figura 6.55 mostra na cor preta os acompanhamentos da trajetória 2 feitos pelo método proposto, na cor azul a saída do filtro IMM e na cor vermelha a trajetória real dos alvos. Com o propósito de diferenciar estas trajetórias as chamaremos de trajetória 2.1 e trajetória 2.2.

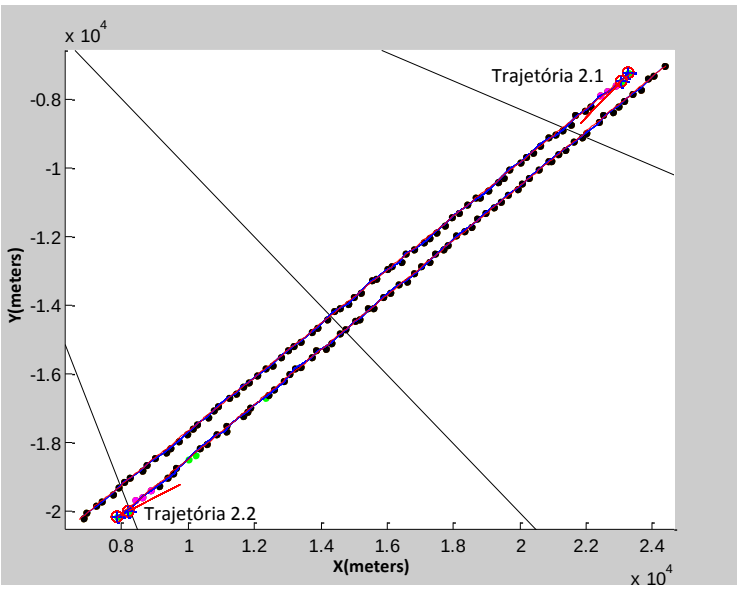


Figura 6.55: Trajetória 2 sugerida no cenário 1

As figuras 6.56 e 6.57, representam os erros de posição nos eixos x e y cometidos

nas trajetórias 2.1 e 2.2 respectivamente, em relação à posição real do alvo.

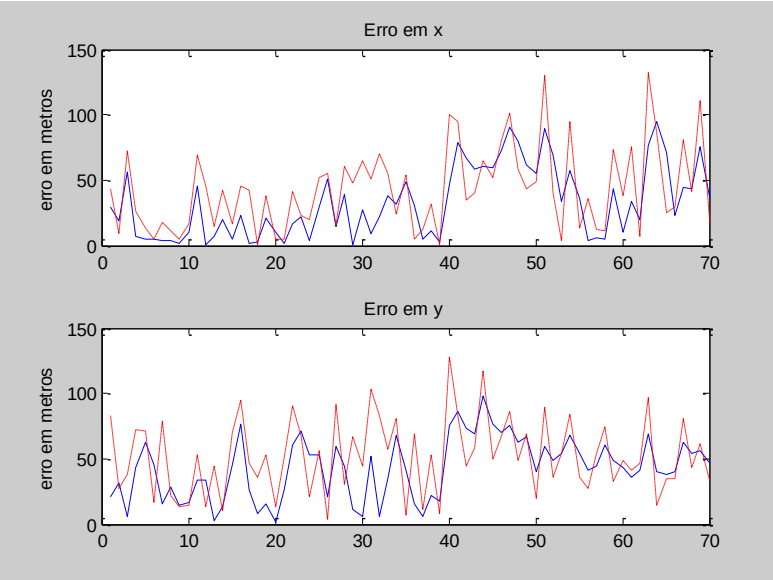


Figura 6.56: Erro de posição nos eixos x e y cometido na trajetória 2.1 sugerida no cenário 1, em relação à posição real do alvo.

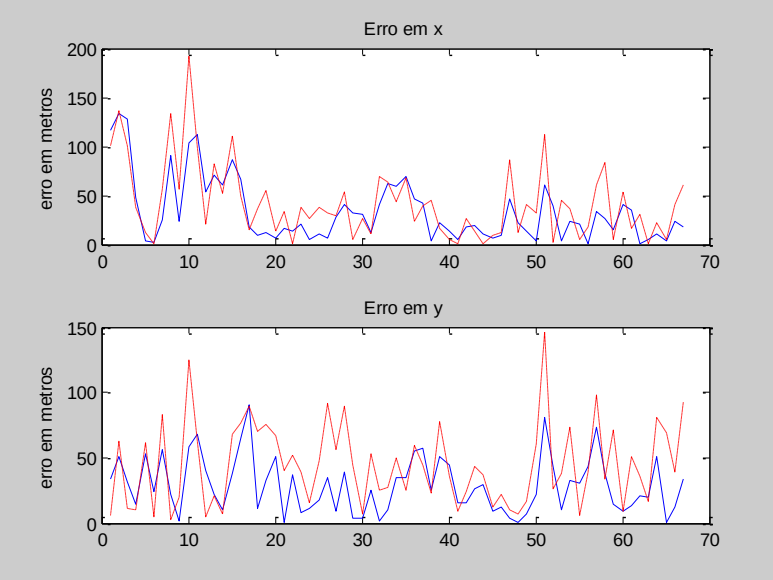


Figura 6.57: Erro de posição nos eixos x e y cometido na trajetória 2.2 sugerida no cenário 1, em relação à posição real do alvo.

- Trajetória 3 sugerida no cenário 6:

  A figura 6.58 apresenta na cor preta um dos acompanhamentos da trajetória 3 sugeridos pelo método proposto no cenário 6 (ver figura 6.51), na cor azul a trajetória filtrada pelo algoritmo IMM e na cor vermelha a trajetória real do alvo.

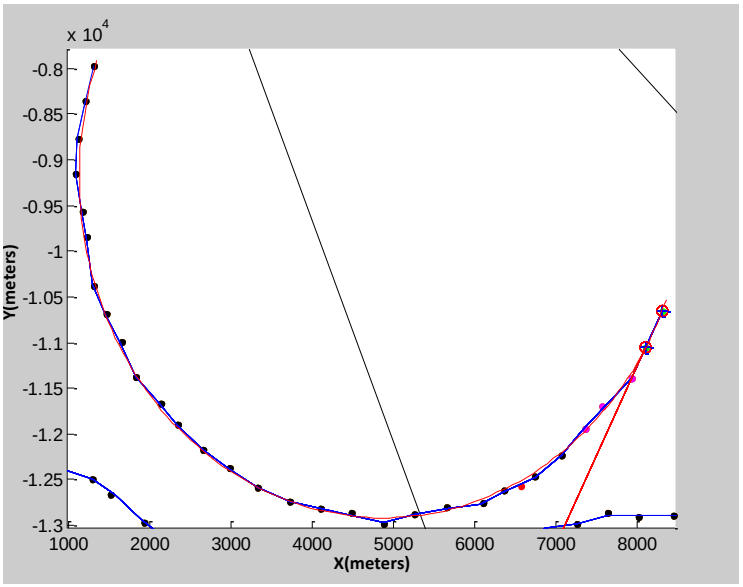


Figura 6.58: Trajetória 3 sugerida no cenário 6

A figura 6.59 apresenta o erro de posição em cada um dos eixos cartesianos em relação à posição real do alvo.

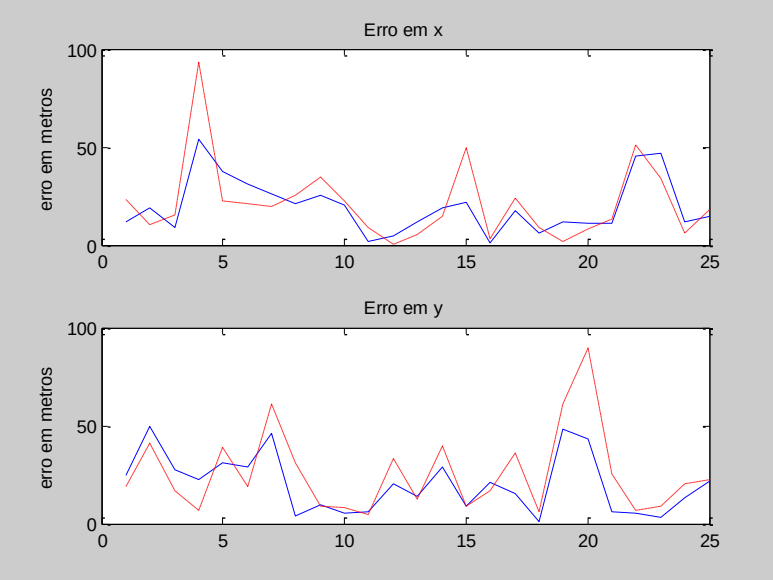


Figura 6.59: Erro de posição nos eixos x e y cometido na trajetória 3 sugerida no cenário 6, em relação à posição real do alvo.

- Trajetória 6 sugerida no cenário 2:

  A figura 6.60 apresenta na cor preta um dos acompanhamentos da trajetória 5 sugeridos pelo método proposto no cenário 2 (ver figura 6.16), na cor azul a trajetória filtrada pelo algoritmo IMM e na cor vermelha a trajetória real do alvo.

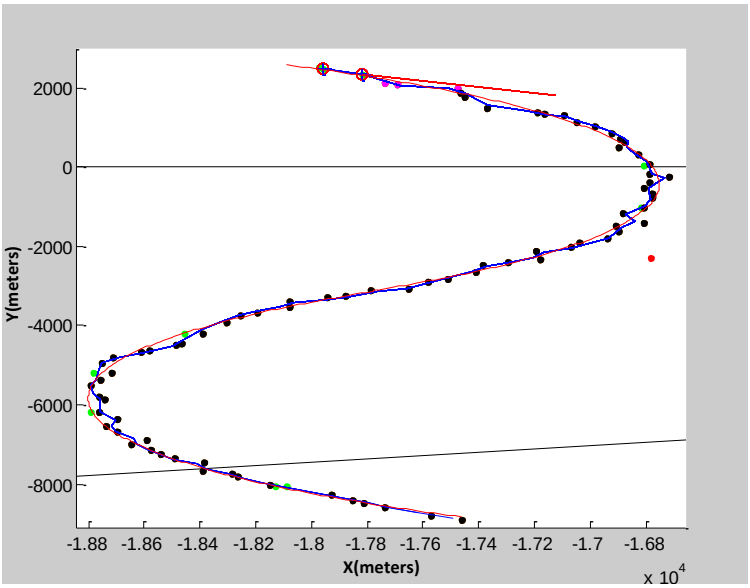


Figura 6.60: Trajetória 6 sugerida no cenário 2

A figura 6.61 apresenta o erro de posição em cada um dos eixos cartesianos em relação à posição real do alvo.

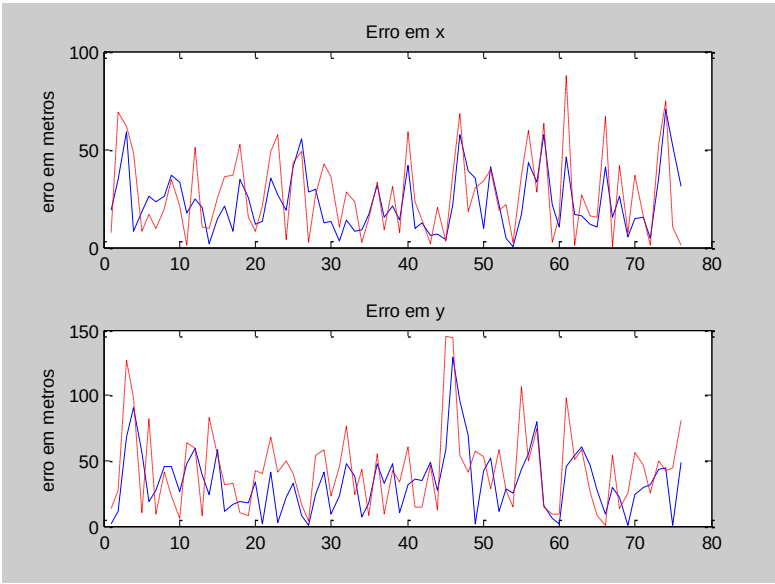


Figura 6.61: Erro de posição nos eixos x e y cometido pela trajetória 6 sugerida no cenário 2, em relação à posição real do alvo.

6.3 Análise comparativa de resultados

A partir das informações apresentadas nas tabelas 6.4, 6.5, 6.6, 6.7, 6.8 e 6.9, pode-se observar que nos cenários 1, 2 e 3 o método clássico apresenta uma taxa de acerto um pouco superior ao método proposto, isto se deve ao fato de o método proposto ser mais restritivo na hora de selecionar os contatos, o que gera um número maior de perdas de detecção e menor número de associações com clutter em comparação com o método clássico em todos os cenários. É importante destacar que apesar de apresentar maior número de perdas isto não se traduz na perda do acompanhamento dos alvos.

Nos cenários onde foram introduzidas as perdas de detecção (cenários 3 e 5) o método clássico não conseguiu manter o acompanhamento de dois dos alvos, o que pode ser considerado como um ponto fraco deste método.

A grande diferença entre os dois métodos é o tempo médio de execução de uma iteração. Pode-se observar que o tempo médio gasto pelo método proposto é inferior a 1/10 do tempo gasto pelo método clássico, o que fornece uma grande vantagem para o método proposto na hora da sua implementação em aplicações em tempo real. Conforme discutido em [2], este resultado justifica-se pelo fato de que um cenário com alta densidade de clutter gera um número elevado de hipóteses, o que aumenta o custo computacional deste método.

Em relação ao número de acompanhamentos confirmados que não representam um alvo de interesse (falsos acompanhamentos), o método clássico apresenta um número maior em comparação com o método proposto, isto é devido à busca intensiva feita pelo método clássico e ao processo de inicialização descrito na seção 2.1. Embora estes acompanhamentos sejam eliminados nas varreduras seguintes, os mesmos podem causar uma queda do desempenho, aumentando o tempo de execução por iteração.

Quanto ao processo de filtragem nas figuras 6.54, 6.56, 6.57, 6.59 e 6.61 é possível observar que o algoritmo IMM consegue filtrar grande parte do ruido introduzido nas trajetórias, independentemente das características de cinemática descritas por cada uma delas. Este filtro mostra-se capaz de se adaptar rapidamente às variações do movimento produzidas ao longo de uma mesma trajetória.

Capítulo 7

Conclusões e trabalhos futuros

7.1 Conclusões

Este trabalho consistiu na comparação de desempenho entre duas metodologias de acompanhamento de múltiplos alvos numa plataforma estática monosensor, a partir de informação extraída do sensor radar em ambientes marítimos com alta densidade de *clutter*. A metodologia clássica para abordar este problema é baseada no algoritmo MHT, como fonte de decisão de associação dos contatos, enquanto a metodologia proposta utiliza uma rede neural MLP para realizar esta mesma tarefa. Adicionalmente, ambas as metodologias fazem uso de uma etapa de inicialização e de uma etapa de filtragem, esta ultima é a encarregada de estimar a trajetória dos alvos a partir das informações perturbadas pelos ruídos provenientes do processo e do sensor, para ambos os casos foi utilizada a filtragem IMM, sendo que, na metodologia clássica esta filtragem também é utilizada para prever a posição futura do alvo.

Este trabalho foi divido em duas etapas, na primeira fase foram avaliados o algoritmo de treinamento Backpropagation com otimização Levenberg Marquart e o Backpropagation com gradiente descendente visando a selecionar o melhor método de treinamento e arquitetura da rede. Na segunda fase, foi construído o método proposto com base na seleção da rede neural feita na fase anterior e comparado com o método clássico. Com o propósito de realizar esta comparação, foram criados diversos cenários e estabelecidas algumas métricas, como o número de acertos e de erros e tempo medio por iteração, entre outras. Assim, foram desenvolvidos dois programas, um que realiza o treinamento da rede neural e compara os algoritmos de treinamento e o outro que executa o método proposto e calcula cada uma das métricas necessárias para a comparação com o método clássico. Além destes dois programas, foram utilizados outros dois programas fornecidos pelo Instituto de Pesquisa da Marinha (IPqM), um deles tornou possível a criação das trajetórias dos

alvos e de cada um dos cenários utilizados nesta dissertação, o outro tem a função de executar o método clássico e calcular cada uma das métricas estabelecidas.

A tarefa da rede neural utilizada neste trabalho, consiste, basicamente em inferir a partir de informações de cinemática se um contato é coerente para ser considerado como atualização de um acompanhamento existente. Ao utilizar esta metodologia se elimina o pressuposto de gaussanidade sobre as medições feito pelo algoritmo MHT. Considerando os resultados obtidos com os diferentes tipos de treinamento (seção 6.1) é possível afirmar que uma rede neural MLP com uma unica camada intermediária treinada pelo algoritmo Backpropagation com otimização Levenberg Marquart é a opção mais adequada para tratar o problema relatado neste trabalho.

Dos resultados obtidos da execução do método proposto e do método clássico para cada um dos cenários construídos, pode ser verificado que o desempenho entre os dois métodos, respeito aos erros e acertos cometidos, é similar, mas o método clássico apresenta algumas dificuldades nos cenários onde foram introduzidas as perdas de detecção, sendo que alguns acompanhamentos foram perdidos e inicializados novamente como um novo alvo. Além de isso, o número de acompanhamentos confirmados é superior em todas as simulações. Enquanto ao tempo medio gasto por iteração, o método proposto apresenta uma redução desta métrica em todos os testes realizados em comparação com o método clássico, isto leva a uma redução significativa do custo computacional e à possibilidade de trabalhar com maior número de alvos nos cenários. Assim, o método proposto apresenta-se como uma alternativa eficaz na realização de acompanhamento de múltiplos alvos.

O algoritmo de filtragem construído possui a capacidade de ponderar dois modelos dinâmicos, um para alvos em movimento retilíneo uniforme e outro para alvos que efetuam manobras visando calcular uma combinação dos modelos de filtro que melhor se adapte as variações de cinemática da trajetória que esteja sendo acompanhada. É importante destacar que estes modelos de filtros são independentes uma vez que seus parâmetros podem ser ajustados individualmente. Finalmente, com relação ao comportamento do algoritmo de filtragem, fica evidente o rápido ajuste da sua saída em presencia de variações da cinemática do alvo, sem importar o tipo de movimento que esteja sendo executado.

7.2 Trabalhos futuros

Existem diferentes pontos que podem ser melhorados em trabalhos futuros entre eles temos:

A utilização de uma massa de dados reais que contenham alvos com diferentes tipos de manobra, velocidades, rumos, marcações e distâncias, sendo capturados em diferentes condições ambientais, isto permitira avaliar de maneira mais eficaz o

método proposto. Tal massa de dados não se encontra disponível.

Analises em cenários com múltiplos sensores, com a finalidade de reproduzir o que acontece realmente no ambiente militar, onde se tem disponíveis diferentes tipos de sensores que fornecem informações distintas, porém complementares, visando aumentar a certeza de uma informação. Neste caso o trabalho é considerado como de fusão de dados e de acompanhamento de múltiplos alvos.

Implementação de novas arquiteturas de RNA ou de outras técnicas de *machine learning* como logica fuzzy, SVM, entre outras, visando melhorar o desempenho obtido pela rede implementada nesta dissertação.

Finalmente, sugere-se realizar uma comparação entre diferentes métodos de filtragem que utilizem o enfoque de múltiplos modelos, entre eles se destaca o *Generalized Pseudo-Bayesian estimator*. Outras metologias utilizadas para realizar esta tarefa são apresentados em [52].

Uma outra estratégia que pode ser experimentada é a inclusão no algoritmo IMM de mais modelos de filtros com diferentes modelos de cinemática ou simplesmente fazer uso de modelos de cinemática mais complexos.

Referências Bibliográficas

[1] BLACKMAN, S. S., "Multiple hypothesis tracking for multiple target tracking", *IEEE Aerospace and Electronic Systems Magazine*, v. 19, n. 1, pp. 5–18, 2004.

[2] AMDITIS, A., THOMAIDIS, G., MAROUDIS, P., et al., "Multiple Hypothesis Tracking Implementation", In: RODRIGUEZ, J. A. M. (ed), *Laser Scanner Technology*, chap. 10, InTechOpen, 2012.

[3] HAYKIN, S., 1999, *Neural Networks a Comprehensive Foundation*. 2nd ed. Pearson Education.

[4] BAR-SHALOM, Y., LI, X.-R., 1995, *Multitarget-Multisensor Tracking: Principles and Techniques*. 3rd ed. Yaakov Bar-Shalom.

[5] BARBOSA, R. P., *Fusão de alvos utilizando grafos em ambientes de múltiplos sensores*, Tese de doutorado, COPPE/UFRJ, Rio de Janeiro, RJ, Brasil, 2012.

[6] WASSERMAN, P. D., 1989, *Neural Computing Theory and Practice*. Van Nostrand Reinhold: New York, USA.

[7] HERTZ, J., KROGH, A., PALMER, R., 1991, *Introduction to the Theory of Neural Computation*. 1st ed. Addison-Wesley Publishing Company: Reading, Massachusetts.

[8] WINTER, M., FAVIER, G., "A neural network for data association", *IEEE International Conference on Acoustics, Speech, and Signal Processing, 1999. Proceedings of the IEEE, 1999*, v. 2, pp. 1041–1044, Mar. 1999.

[9] WINTER, M., FAVIER, G., "A neural solution for multitarget tracking based on a maximum likelihood approach", *IEEE International Conference on Acoustics, Speech and Signal Processing, 1998. Proceedings of the IEEE*, v. 2, pp. 1141–1144, May. 1998.

[10] LIN, Y. J., CHEN, H. T., SU, J. D., et al., "Applying the Competitive Hopfield Neural Network to Multiple Target Tracking Systems", *Automatic Control Conference Tainan, Taiwan*, Nov. 2005.

[11] RICHARD, M. D., LIPPMANN, R. P., "Neural network classifiers estimate Bayesian a posteriori probabilities", *Neural computation*, v. 3, n. 4, pp. 461–483, 1991.

[12] GUANG-YUAN, Z., FU-JUN, W., ZHEN-SHENG, W., "Interacting Multiple Model Algorithm Used In Multi-Sensor Fusion System", *8th International Conference on Electronic Measurement and Instruments*, , n. 4, pp. 4–135 4–139, 2007.

[13] KIRUBARAJAN, T., BAR-SHALOM, Y., "Kalman Filter vs. IMM Estimator: When Do We Need the Latter?" *IEEE Transactions on Aerospace and Electronic Systems*, v. 39, n. 4, pp. 1452 1457, Oct. 2003.

[14] YEDDANAPUDI, M., BAR-SHALOM, Y., PATTIPATI, K. R., "IMM estimation for multitarget-multisensor air traffic surveillance", *Proceedings of the IEEE*, v. 85, n. 1, pp. 80–96, 1997.

[15] BLOM, H. A. P., BAR-SHALOM, Y., "The interacting multiple model algorithm for systems with Markovian switching coefficients", *IEEE Transactions on Automatic Control*, v. 33, n. 8, pp. 780–783, 1988.

[16] PETERS, D. J., *A Practical Guide to Level One Data Fusion Algorithms*, Tech. Rep. DREA TM 2001-201, Defence Research Establishment Atlantic, 2001.

[17] GINI, F., RANGASWAMY, M., 2008, *Knowledge based radar detection, tracking and classification*. John Wiley & Sons.

[18] REID, D. B., "An algorithm for tracking multiple targets", *IEEE Transactions on Automatic Control*, v. 24, n. 6, pp. 843–854, Dez. 1979.

[19] BLACKMAN, S. S., 1986, *Multiple Target Tracking with Radar Applications*. Norwood, MA: Artech House.

[20] BLACKMAN, S. S., DEMPSTER, R. J., REED, R., "Demonstration of multiple-hypothesis tracking (MHT) practical real-time implementation feasibility". In: *Signal and Data Processing of Small Targets*, v. 4473, pp. 470–475, 2001.

[21] BAR-SHALOM, Y., 1990, *Multitarget-multisensor tracking: advanced applications*. v. 1. Artech House: Norwood, MA.

[22] CYBENKO, G., "Approximation by superpositions of a sigmoidal function", *Mathematics of control, signals and systems*, v. 2, n. 4, pp. 303–314, 1989.

[23] CALÔBA, L. P., "Introdução ao uso de Redes Neurais na Modelagem de Sistemas Dinâmicos e Series Temporais", Livro de Mini Cursos do XIV Congresso Brasileiro de Automática, 2002.

[24] MCCULLOCH, W. S., PITTS, W., "A logical calculus of the ideas immanent in nervous activity", *The bulletin of mathematical biophysics*, v. 5, n. 4, pp. 115–133, 1943.

[25] ROSENBLATT, F., "The perceptron: a probabilistic model for information storage and organization in the brain", *Psychological review*, v. 65, n. 6, pp. 386–408, 1958.

[26] MINSKY, M., PAPERT, S., 1969, *Perceptrons: An Introduction to Computational Geometry*. The MIT Press: Cambridge, Massachusetts.

[27] DE FREITAS PARREIRAS, L. P. R., *Arbitragem Estatística e Inteligência Artificial*, Tese m.sc, Universidade de São Paulo, São Paulo, 2007.

[28] RUMELHART, D. E., HINTON, G. E., WILLIAMS, R. J., "Learning representations by back-propagating errors", *Nature*, v. 323, pp. 533–536, 1986.

[29] HAGAN, M. T., MENHAJ, M. B., "Training feedforward networks with the Marquardt algorithm", *Neural Networks, IEEE Transactions on*, v. 5, n. 6, pp. 989–993, Nov. 1994.

[30] MARQUARDT, D. W., "An algorithm for least-squares estimation of nonlinear parameters", *Journal of the Society for Industrial & Applied Mathematics*, v. 11, n. 2, pp. 431–441, Jun. 1963.

[31] YU, H., WILAMOWSKI, B. M., "Levenberg-marquardt training". In: *The Industrial Electronics Handbook*, v. 5, pp. 1–16, 2011.

[32] YU, H., WILAMOWSKI, B. M., "Comparison of neural network learning algorithms for prediction enhancement of a planning tool". In: *The Industrial Electronics Handbook*, 2007.

[33] HAGAN, M. T., DEMUTH, H. B., BEALE, M. H., 1996, *Neural network design*. v. 1. Pws Boston.

[34] BATTITI, R., "First-and second-order methods for learning: between steepest descent and Newton's method", *Neural computation*, v. 4, n. 2, pp. 141–166, 1992.

[35] SURATGAR, A. A., TAVAKOLI, M. B., HOSEINABADI, A., "Modified Levenberg-Marquardt Method for Neural Networks Training", *International Journal of Computer, Information, Systems and Control Engineering*, v. 1, n. 6, pp. 1748–1750, 2007.

[36] HAYKIN, S. S., 2001, *Kalman filtering and neural networks*. John Wiley & Sons, Inc.

[37] WELCH, G., BISHOP, G., "An Introduction to the Kalman Filter", *Department of Computer Science University of North Carolina at Chapel Hill*, 2006.

[38] DURRANT-WHYTE, H. F., 2001, *Introduction to estimation and the Kalman filter*. Australian Centre for Field Robotics.

[39] DING, Z., HONG, L., "An interacting multiple model algorithm with a switching Markov chain", *Mathematical and Computer Modelling*, v. 25, n. 1, pp. 1–9, 1997.

[40] CASELLA, G., BERGER, R. L., 2011, *Statistical inference*. Duxbury Thomson Learning.

[41] BAR-SHALOM, Y., FORTMANN, T. E., 1988, *Tracking and Data Association*. Academic Press.

[42] ALY, S. M., FOULY, R. E., BARAKA, H., "Extended Kalman Filtering and Interacting Multiple Model for Tracking Maneuvering Targets in Sensor Netwotrks", *Seventh Workshop on Intelligent solutions in Embedded Systems*, pp. 149–156, Jun. 2009.

[43] BLACKMAN, S. S., POPOLI, R., 1999, *Design and analysis of modern tracking systems*. Artech House.

[44] DANG, H., HAN, C., GRUYER, D., "Combining of IMM filtering and DS data association for multitarget tracking", *Fusion 2004: Seventh International Conference on Information Fusion*, 2004.

[45] BAR-SHALOM, Y., LI, X. R., KIRUBARAJAN, T., 2001, *Estimation with applications to tracking and navigation: theory algorithms and software*. John Wiley & Sons.

[46] MIGUENS, A. P., 1996, *Navegação a Ciência e a Arte: Navegação Costeira, Estimada e em Águas Restritas*. v. 1. Rio de Janeiro: DHN.

[47] KANG, E. W., 2008, *Radar system analysis, design, and simulation*. Artech House.

[48] DEMPSTER, R. J., BLACKMAN, S. S., NICHOLS, T. S., "Combining IMM filtering and MHT data association for multitarget tracking". In: *Proceedings of the Twenty-Ninth Southeastern Symposium on System Theory*, pp. 123–127, 1997.

[49] BAR-SHALOM, Y., WILLETT, P. K., TIAN, X., 2011, *Tracking and Data Fusion a Handbook of Algorithms*. Yaakov Bar-Shalom.

[50] LI, X.-R., JILKOV, V. P., "Survey of Maneuvering Target Tracking: Dynamic Models". In: *Proceedings of SPIE Conference on Signal and Data Processing of Small Targets*, v. 4048, pp. 212–235, Orlando, FL, USA, April 2000.

[51] YUAN, X., LIAN, F., HAN, C., "Models and Algorithms for Tracking Target with Coordinated Turn Motion", *Mathematical Problems in Engineering*, v. 2014, n. 649276, pp. 1–10, 2014.

[52] DE OLIVEIRA, J. R. P., *Acompanhamento de Alvos Radar Utilizando Filtragem de Kalman e Vetor de Estados com Dimensão Variável*, Tese de mestrado, PEE - COPPE/UFRJ, Rio de Janeiro, RJ, Brasil, 2005.

[53] HUSSAIN, D. M. A., AHMED, Z., "Object Tracking under High Correlation for Kalman & α-β Filter", In: HUSSAIN, D. M. A. (ed), *Advances in Computer Science and IT*, chap. 5, InTechOpen, 2009.

[54] ANDERSON, B. D. O., MOORE, J. B., 1979, *Optimal filtering*. Prentice-Hall.

[55] LERRO, D., BAR-SHALOM, Y., "Tracking with Debiased Consistent Converted Measurements Versus EKF", *IEEE Transactions on Aerospace and Electronic Systems*, v. 29, n. 3, pp. 1015–1022, 1993.

[56] KORBICZ, J., KOWALCZUK, J. M. K. Z., CHOLEWA, W., 2004, *Fault Diagnosis Models, Artificial Intelligence, Applications*. Springer-Verlag Berlin Heidelberg.

Apêndice A

Dedução dos modelos de cinemática do filtro

Os modelos (MRU e CT) considerados neste trabalho representam casos particulares do modelo de cinemática do movimento curvilíneo planar (CLM) [56]. Assim, introduzimos a abordagem deste modelo em tempo contínuo. É importante destacar que, para fins de conotação, qualquer variável definida em função de t ou de k indica respectivamente que está sendo analisada em tempo contínuo ou em tempo discreto respectivamente.

A velocidade do alvo pode ser descrita em função das coordenadas cartesianas a partir de $V(t)$ (velocidade tangencial instantânea) e $\phi(t)$ (*heading angle* ou curso do navio):

$$V_x(t) = \dot{x}(t) = V(t) * \cos\phi(t) \tag{A.1}$$

$$V_y(t) = \dot{y}(t) = V(t) * \sin\phi(t) \tag{A.2}$$

onde

$$V(t) = \sqrt{V_x^2 + V_y^2}. \tag{A.3}$$

A diferenciação das equações A.1 e A.2 resulta nas equações diferenciais de segunda ordem,

$$\begin{aligned} a_x(t) = \ddot{x}(t) &= \dot{V}(t) * \cos\phi(t) - V(t) * \dot{\phi}(t) * \sin\phi(t) \\ &= \dot{V}(t) * \cos\phi(t) - V_y(t) * \dot{\phi}(t), \end{aligned} \tag{A.4}$$

$$\begin{aligned} a_y(t) = \ddot{y}(t) &= \dot{V}(t) * \sin\phi(t) + V(t) * \dot{\phi}(t) * \cos\phi(t) \\ &= \dot{V}(t) * \sin\phi(t) + V_x(t) * \dot{\phi}(t), \end{aligned} \tag{A.5}$$

que descrevem o movimento curvilíneo do alvo. Neste modelo, considera-se que atuam sobre o alvo duas forças perpendiculares entre si nomeadamente aceleração tangencial $a_t(t)$ e aceleração normal $a_n(t)$, definidas como:

$$a_t(t) = \dot{V}(t) \tag{A.6}$$

$$a_n(t) = V(t) * \dot{\phi}(t) = V(t) * \omega(t), \tag{A.7}$$

onde $\omega(t)$ é a velocidade angular ou taxa de giro.

A figura A.1 mostra as componentes de uma trajetória para o modelo CLM.

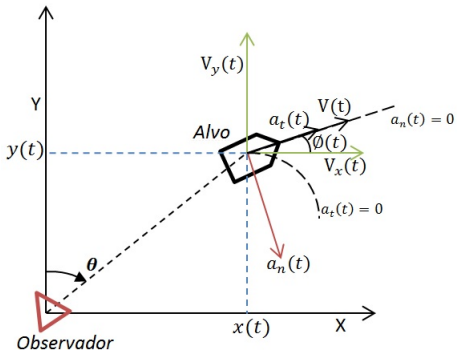


Figura A.1: Componente Movimento Curvilíneo

A.1 Dedução Modelo MRU:

O modelo MRU em tempo contínuo parte do pressuposto de que o alvo se movimenta em linha reta com velocidade constante desconhecida. Por conseguinte, as equações do modelo MRU, derivadas das equações A.1, A.2, A.4, A.5, A.6 e A.7 do modelo CLM, sao dadas por

$$Vx(t) = a \tag{A.8}$$

$$Vy(t) = b \tag{A.9}$$

$$a_n(t) = a_t(t) = 0 \tag{A.10}$$

$$a_x(t) = a_y(t) = 0 \tag{A.11}$$

onde a e b são constantes. As equações acima podem ser resumidas pela seguinte equação vetorial do processo em tempo continuo:

$$\dot{X}(t) = A * X(t) + U, \tag{A.12}$$

onde U é o ruído do processo em tempo contínuo, porém visto como uma aceleração constante, definido como

$$U = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} * w, \tag{A.13}$$

sendo w o vetor composto por ruído em cada um dos eixos cartesianos:

$$w = \begin{bmatrix} w_x \\ w_y \end{bmatrix}. \tag{A.14}$$

O vetor de estados $X(t)$ deste modelo é definido por

$$X(t) = \begin{bmatrix} x(t) \\ V_x(t) \\ y(t) \\ V_y(t) \end{bmatrix}, \tag{A.15}$$

por conseguinte

$$\dot{X}(t) = \begin{bmatrix} \dot{x}(t) \\ a_x(t) \\ \dot{y}(t) \\ a_y(t) \end{bmatrix}, \tag{A.16}$$

onde $a_x = \dot{V}_x$ e $a_y = \dot{V}_y$. A matriz de transição de estado A, é obtida substituindo A.13, A.15 e A.16 em A.12, assim,

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}. \tag{A.17}$$

A discretização da equação do processo A.12 no período de amostragem T_k é feita utilizando os três primeiros termos da expansão em series de Taylor, ou seja,

$$X(k+1) = X(k) + (T_k) * \dot{X}(k) + \frac{(T_k)^2}{2} * \ddot{X}(k), \tag{A.18}$$

onde $T_k = t_{k+1} - t_k$ é considerado variável, por depender do tempo de rotação da antena, da velocidade e do curso do alvo. A variável $X(k)$ representa o vetor de

estados A.15 no instante de tempo k, ou seja,

$$X(k) = \begin{bmatrix} x(k) \\ V_x(k) \\ y(k) \\ V_y(k) \end{bmatrix}, \tag{A.19}$$

$\dot{X}(k)$ representa a equação do modelo A.12 no instante de tempo k,

$$\dot{X}(k) = A * X(k) + U, \tag{A.20}$$

e $\ddot{X}(k)$ é a derivada da equação A.20 com respeito a k, para U constante

$$\ddot{X}(k) = A * \dot{X}(k). \tag{A.21}$$

Substituindo as equações A.20 e A.21 em A.18 obtemos

$$\begin{aligned} X(k+1) &= X(k) + T_k * (A * X(k) + U) + \frac{T_k^2}{2} * A * (A * X(k) + U) \\ &= (Id + T_k * A + \frac{T_k^2}{2} * A^2) * X(k) + (Id * T_k + \frac{T_k^2}{2} * A) * U, \end{aligned} \tag{A.22}$$

onde Id é a matriz identidade.

Sabemos da definição do KF, feita no capitulo 4, que a dinámica do alvo modelada pela equação vetorial do processo em tempo discreto é definida segundo a equação 4.3, isto é,

$$X(k+1) = F(k) * X(k) + W(k). \tag{A.23}$$

Como referido anteriormente, o ruído do processo, $W(k)$, é visto como uma aceleração constante. Assim, reescrevendo a equação A.22 em forma de A.23, teremos

$$F(k) = \left(Id + T_k * A + \frac{T_k^2}{2} * A^2 \right) \tag{A.24}$$

$$W(k) = \left(Id * T_k + \frac{T_k^2}{2} * A \right) * U. \tag{A.25}$$

Substituindo A.17 e A.13 em A.24 e A.25 teremos que

$$F(k) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} + \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \cdot T_k + \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}^2 \cdot \frac{T_k^2}{2}$$

$$= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} + \begin{bmatrix} 0 & T_k & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & T_k \\ 0 & 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

$$= \begin{bmatrix} 1 & T_k & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T_k \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{A.26}$$

e

$$W(k) = \left(\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \cdot T_k + \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \cdot \frac{T_k^2}{2} \right) \cdot \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} * \begin{bmatrix} w_x \\ w_y \end{bmatrix}$$

$$= \begin{bmatrix} T_k & \frac{T_k^2}{2} & 0 & 0 \\ 0 & T_k & 0 & 0 \\ 0 & 0 & T_k & \frac{T_k^2}{2} \\ 0 & 0 & 0 & T_k \end{bmatrix} \cdot \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{bmatrix} * \begin{bmatrix} w_x \\ w_y \end{bmatrix}$$

$$= \begin{bmatrix} \frac{T_k^2}{2} & 0 \\ T_k & 0 \\ 0 & \frac{T_k^2}{2} \\ 0 & T_k \end{bmatrix} \cdot \begin{bmatrix} w_x \\ w_y \end{bmatrix} \tag{A.27}$$

$$= B(k) \cdot w. \tag{A.28}$$

Finalmente, substituindo A.26 e A.27 em A.23 obtemos a equação do processo de nosso modelo:

$$X(k+1) = \begin{bmatrix} 1 & T_k & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & T_k \\ 0 & 0 & 0 & 1 \end{bmatrix} * X(k) + \begin{bmatrix} \frac{T_k^2}{2} & 0 \\ T_k & 0 \\ 0 & \frac{T_k^2}{2} \\ 0 & T_k \end{bmatrix} \cdot \begin{bmatrix} w_x \\ w_y \end{bmatrix} \tag{A.29}$$

Assumindo os ruído w_x e w_y independentes com média zero e variâncias constantes e iguais ($\sigma_x^2 = \sigma_y^2 = \sigma_q^2$) [52, 47], a matriz da covariância dos ruídos associados ao estado é dada por

$$\begin{aligned} Q(k) &= E[W(k) \cdot W(k)^T] \\ &= E[(B(k) \cdot w) \cdot (B(k) \cdot w)^T] \\ &= B(k) \cdot E[w \cdot w^T] \cdot B(k)^T \\ &= B(k) \cdot \begin{bmatrix} \sigma_x^2 & 0 \\ 0 & \sigma_y^2 \end{bmatrix} \cdot B(k)^T. \end{aligned} \tag{A.30}$$

Fazendo as devidas substituições temos que

$$\begin{aligned} Q(k) &= \begin{bmatrix} \frac{T_k^2}{2} & 0 \\ T_k & 0 \\ 0 & \frac{T_k^2}{2} \\ 0 & T_k \end{bmatrix} \cdot \begin{bmatrix} \sigma_q^2 & 0 \\ 0 & \sigma_q^2 \end{bmatrix} \cdot \begin{bmatrix} \frac{T_k^2}{2} & T_k & 0 & 0 \\ 0 & 0 & \frac{T_k^2}{2} & T_k \end{bmatrix} \\ &= \begin{bmatrix} \frac{T_k^4}{4}\sigma_q^2 & \frac{T_k^3}{2}\sigma_q^2 & 0 & 0 \\ \frac{T_k^3}{2}\sigma_q^2 & T_k^2\sigma_q^2 & 0 & 0 \\ 0 & 0 & \frac{T_k^4}{4}\sigma_q^2 & \frac{T_k^3}{2}\sigma_q^2 \\ 0 & 0 & \frac{T_k^3}{2}\sigma_q^2 & T_k^2\sigma_q^2 \end{bmatrix} \end{aligned} \tag{A.31}$$

A.2 Modelo CT:

No modelo CT assume-se que o alvo se movimenta com velocidade e taxa de giro constantes e desconhecidas. Portanto, atendendo estes pressupostos, as equações do modelo CT (expressas em tempo continuo) podem ser derivadas das equações A.1, A.2, A.4, A.5, A.6 e A.7 do modelo CLM, da seguinte forma

$$Vx(t) = a, \tag{A.32}$$

$$Vy(t) = b, \tag{A.33}$$

$$a_t(t) = \dot{V}(t) = 0, \tag{A.34}$$

$$a_n(t) = V(t) * \omega(t) = c, \tag{A.35}$$

$$a_x(t) = \dot{V}_x(t) = -\omega(t) * V_y(t), \tag{A.36}$$

$$a_y(t) = \dot{V}_y(t) = \omega(t) * V_x(t), \tag{A.37}$$

onde a, b, c são constantes desconhecidas. Assim, as equações acima são representadas pela seguinte equação vetorial do processo:

$$\dot{X}(t) = A(t) * X(t) + U, \tag{A.38}$$

onde U é o ruído do processo em tempo contínuo, porém visto como uma perturbação na aceleração e na taxa de giro, isto é,

$$U = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} * w, \tag{A.39}$$

sendo w o vetor composto pelos ruídos em cada um dos eixos cartesianos (w_x, w_y) e na taxa de giro (w_ω):

$$w = \begin{bmatrix} w_x \\ w_y \\ w_\omega \end{bmatrix}. \tag{A.40}$$

O vetor de estados $X(t)$ é definido por

$$X(t) = \begin{bmatrix} x(t) \\ V_x(t) \\ y(t) \\ V_y(t) \\ \omega(t) \end{bmatrix}. \tag{A.41}$$

Por conseguinte,

$$\dot{X}(t) = \begin{bmatrix} \dot{x}(t) \\ a_x(t) \\ \dot{y}(t) \\ a_y(t) \\ 0 \end{bmatrix}, \tag{A.42}$$

onde $a_x(t)$ e $a_y(t)$ são dadas respectivamente por A.36 e A.37, pelo que

$$\dot{X}(t) = \begin{bmatrix} \dot{x}(t) \\ -\omega(t) * V_y(t) \\ \dot{y}(t) \\ \omega(t) * V_x(t) \\ 0 \end{bmatrix}. \tag{A.43}$$

A matriz de transição de estados $A(t)$ é obtida ao substituir A.41 e A.43 em A.38,

$$A(t) = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\omega(t) & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & \omega(t) & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}. \tag{A.44}$$

Tal como no modelo MRU, a equação do processo A.38 em tempo discreto é obtida através da expansão em series de Taylor,

$$X(k+1) = X(k) + T_k * \dot{X}(k) + \frac{T_k^2 * \ddot{X}(k)}{2}, \tag{A.45}$$

onde $X(k)$, $\dot{X}(k)$ e $A(k)$ são respectivamente o vetor de estados A.41, a equação do modelo A.38 e a matriz de transição de estados no instante de tempo k, dados por

$$X(k) = \begin{bmatrix} x(k) \\ V_x(k) \\ y(k) \\ V_y(k) \\ \omega(k) \end{bmatrix}, \tag{A.46}$$

$$\dot{X}(k) = A(k) * X(k) + U, \tag{A.47}$$

$$A(k) = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\omega(k) & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & \omega(k) & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}, \tag{A.48}$$

e $\ddot{X}(k)$ é obtido derivando $\dot{X}(k)$, assumindo U constante, ou seja,

$$\ddot{X}(k) = A(k)^2 * X(k) + A(k) * U. \tag{A.49}$$

A partir das definições anteriores, obtemos que

$$\begin{aligned} X(k+1) &= X(k) + T_k * (A(k) * X(k) + U) \\ &\quad + \frac{T_k^2 * A(k)^2 * X(k) + A(k) * U}{2} \\ &= (Id + T_k * A(k) + \frac{T_k^2}{2} * A(k)^2) * X(k) \\ &\quad + (Id * T_k + \frac{T_k^2}{2} * A(k)) * U. \end{aligned} \tag{A.50}$$

Usando o raciocínio análogo ao aplicado no modelo MRU, os termos $F(k)$ e $W(k)$ da equação do processo para o modelo CT (A.50) são dados por

$$F(k) = \left(Id + T_k * A(k) + \frac{T_k^2}{2} * A(k)^2 \right) \tag{A.51}$$

$$W(k) = \left(Id * T_k + \frac{T_k^2}{2} * A(k) \right) * U. \tag{A.52}$$

Substituindo A.48 e A.39 em A.51 e A.52, temos:

$$\begin{aligned} F(k) &= \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} + \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\omega(k) & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & \omega(k) & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} * T_k \\ &\quad + \begin{bmatrix} 0 & 0 & 0 & -\omega(k) & 0 \\ 0 & -\omega(k)^2 & 0 & 0 & 0 \\ 0 & \omega(k) & 0 & 0 & 0 \\ 0 & 0 & 0 & -\omega(k)^2 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} * \frac{T_k^2}{2} \\ &= \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} + \begin{bmatrix} 0 & T & 0 & -\omega(k) * \frac{T_k^2}{2} & 0 \\ 0 & -\omega(k)^2 * \frac{T_k^2}{2} & 0 & -\omega(k) * T_k & 0 \\ 0 & \omega(k) * \frac{T_k^2}{2} & 0 & T_k & 0 \\ 0 & \omega(k) * T_k & 0 & -\omega(k)^2 * \frac{T_k^2}{2} & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} \\ &= \begin{bmatrix} 1 & T_k & 0 & -\omega(k) * \frac{T_k^2}{2} & 0 \\ 0 & 1 - \omega(k)^2 * \frac{T_k^2}{2} & 0 & -\omega(k) * T_k & 0 \\ 0 & \omega(k) * \frac{T_k^2}{2} & 1 & T_k & 0 \\ 0 & \omega(k) * T_k & 0 & 1 - \omega(k)^2 * \frac{T_k^2}{2} & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \end{aligned} \tag{A.53}$$

e

$$W(k) = \left(\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} * T_k + \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -\omega(k) & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & \omega(k) & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix} * \frac{T_k^2}{2} \right)$$

$$* \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} w_x \\ w_y \\ w_\omega \end{bmatrix}$$

$$= \begin{bmatrix} T_k & \frac{T_k^2}{2} & 0 & 0 & 0 \\ 0 & T_k & 0 & -\omega(k) * \frac{T_k^2}{2} & 0 \\ 0 & 0 & T_k & \frac{T_k^2}{2} & 0 \\ 0 & \omega(k) * \frac{T_k^2}{2} & 0 & T_k & 0 \\ 0 & 0 & 0 & 0 & T_k \end{bmatrix} * \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} w_x \\ w_y \\ w_\omega \end{bmatrix}$$

$$= \begin{bmatrix} \frac{T_k^2}{2} & 0 & 0 \\ T_k & 0 & 0 \\ 0 & \frac{T_k^2}{2} & 0 \\ 0 & T_k & 0 \\ 0 & 0 & T_k \end{bmatrix} * \begin{bmatrix} w_x \\ w_y \\ w_\omega \end{bmatrix} \tag{A.54}$$

$$= B(k) * w. \tag{A.55}$$

Assim, a equação do processo deste modelo é dada por

$$X(k+1) = \begin{bmatrix} 1 & T_k & 0 & -\omega(k) * \frac{T_k^2}{2} & 0 \\ 0 & 1 - \omega(k)^2 * \frac{T_k^2}{2} & 0 & -\omega(k) * T_k & 0 \\ 0 & \omega(k) * \frac{T_k^2}{2} & 1 & T_k & 0 \\ 0 & \omega(k) * T_k & 0 & 1 - \omega(k)^2 * \frac{T_k^2}{2} & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} * X(k)$$

$$+ \begin{bmatrix} \frac{T_k^2}{2} & 0 & 0 \\ T_k & 0 & 0 \\ 0 & \frac{T_k^2}{2} & 0 \\ 0 & T_k & 0 \\ 0 & 0 & T_k \end{bmatrix} * \begin{bmatrix} w_x \\ w_y \\ w_\omega \end{bmatrix}. \tag{A.56}$$

Como a equação do processo (A.56) não é linear, podemos reescreve-la na forma da equação 4.25, apresentada na seção 4.2. Assim, para o nosso caso, a função

vetorial de transição de estados, $f(k, X(k))$ (4.27), será definida como

$$\begin{aligned} f(k, X(k)) &= F(k) * X(k) \\ &= \begin{bmatrix} x(k) + V_x(k) * T_k - V_y(k) * \omega(k) * \frac{T_k^2}{2} \\ V_x(k) - V_x(k) * \omega(k)^2 * \frac{T_k^2}{2} - V_y(k) * \omega(k) * T_k \\ y(k) + V_y(k) * T_k + V_x(k) * \omega(k) * \frac{T_k^2}{2} \\ V_y(k) - V_y(k) * \omega(k)^2 * \frac{T_k^2}{2} + V_x(k) * \omega(k) * T_k \\ \omega(k) \end{bmatrix}. \end{aligned} \tag{A.57}$$

Por conseguinte, a equação do processo é dada por

$$\begin{aligned} X(k+1) = &\begin{bmatrix} x(k) + V_x(k) * T_k - V_y(k) * \omega(k) * \frac{T_k^2}{2} \\ V_x(k) - V_x(k) * \omega(k)^2 * \frac{T_k^2}{2} - V_y(k) * \omega(k) * T_k \\ y(k) + V_y(k) * T_k + V_x(k) * \omega(k) * \frac{T_k^2}{2} \\ V_y(k) - V_y(k) * \omega(k)^2 * \frac{T_k^2}{2} + V_x(k) * \omega(k) * T_k \\ \omega(k) \end{bmatrix} \\ &+ \begin{bmatrix} \frac{T_k^2}{2} & 0 & 0 \\ T_k & 0 & 0 \\ 0 & \frac{T_k^2}{2} & 0 \\ 0 & T_k & 0 \\ 0 & 0 & T_k \end{bmatrix} * \begin{bmatrix} w_x \\ w_y \\ w_\omega \end{bmatrix}, \end{aligned} \tag{A.58}$$

A matriz de covariância dos ruídos é calculada da seguinte forma:

$$\begin{aligned} Q(k) &= E[W(k) \cdot W(k)^T] \\ &= E[(B(k) \cdot w) \cdot (B(k) \cdot w)^T] \\ &= B(k) \cdot E[w \cdot w^T] \cdot B(k)^T \end{aligned} \tag{A.59}$$

onde w é o vetor de ruídos, assumidos como independentes, com media zero e variâncias σ_x^2, σ_y^2 e σ_ω^2. Desta forma

$$Q(k) = B(k) \cdot \begin{bmatrix} \sigma_x^2 & 0 & 0 \\ 0 & \sigma_y^2 & 0 \\ 0 & 0 & \sigma_\omega^2 \end{bmatrix} \cdot B(k)^T. \tag{A.60}$$

Substituindo $B(k)$ e assumindo que $\sigma_x^2 = \sigma_y^2 = \sigma_q^2$ temos que

$$
\begin{aligned}
Q(k) &= \begin{bmatrix} \frac{T_k^2}{2} & 0 & 0 \\ T_k & 0 & 0 \\ 0 & \frac{T_k^2}{2} & 0 \\ 0 & T_k & 0 \\ 0 & 0 & T_k \end{bmatrix} * \begin{bmatrix} \sigma_q^2 & 0 & 0 \\ 0 & \sigma_q^2 & 0 \\ 0 & 0 & \sigma_\omega^2 \end{bmatrix} * \begin{bmatrix} \frac{T_k^2}{2} & T_k & 0 & 0 & 0 \\ 0 & 0 & \frac{T_k^2}{2} & T_k & 0 \\ 0 & 0 & 0 & 0 & T_k \end{bmatrix} \\
&= \begin{bmatrix} \frac{T_k^4}{4} * \sigma_q^2 & \frac{T_k^3}{2} * \sigma_q^2 & 0 & 0 & 0 \\ \frac{T_k^3}{2} * \sigma_q^2 & \frac{T_k^2}{2} * \sigma_q^2 & 0 & 0 & 0 \\ 0 & 0 & \frac{T_k^4}{4} * \sigma_q^2 & \frac{T_k^3}{2} * \sigma_q^2 & 0 \\ 0 & 0 & \frac{T_k^3}{2} * \sigma_q^2 & \frac{T_k^2}{2} * \sigma_q^2 & 0 \\ 0 & 0 & 0 & 0 & \frac{T_k^2}{2} * \sigma_\omega^2 \end{bmatrix}
\end{aligned} \tag{A.61}
$$