

# Relatório de Projeto 3: Filtro de Kalman

Tema: Localização e Rastreamento contínuo de Robô Aspirador

Disciplina: Inteligência Artificial (FGA0221)

Professor: Fabiano Araujo Soares, Dr

Aluno: Marcos Antonio Teles de Castilhos

## 1. Introdução e Objetivo

O objetivo do Projeto 3 foi desenvolver um sistema de rastreamento (tracking) para um robô móvel descrevendo uma trajetória circular, utilizando sensores ruidosos de posição (GPS/LiDAR simulados).

Este relatório documenta a evolução técnica do projeto, comparando duas abordagens distintas:

1. Filtro de Kalman Linear (Abordagem Inicial): Tentativa de modelar o movimento circular usando física de velocidade constante linear.
2. Filtro de Kalman Estendido - EKF (Abordagem Final): Modelagem correta da não-linearidade usando coordenadas polares e linearização via Jacobianas.

O foco da análise é demonstrar como a escolha correta do modelo físico impacta drasticamente a precisão da estimação.

## 2. Abordagem 1: O Filtro Linear (Limitações)

A primeira iteração utilizou o Filtro de Kalman padrão (KF).

Modelagem:

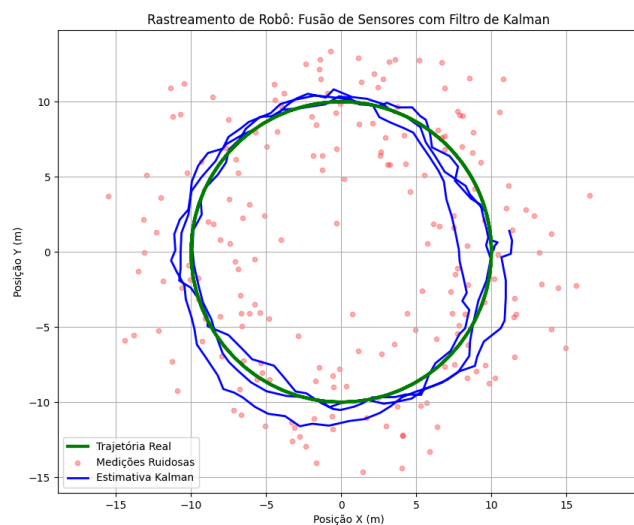
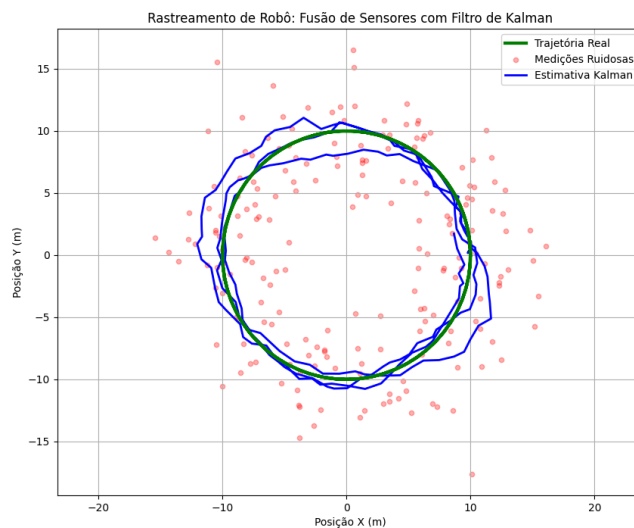
- Estado: Vetor cartesiano  $[x, y, v_x, v_y]$ .

- Física: Suposição de Movimento Retilíneo Uniforme (MRU) entre os passos de tempo. A matriz de transição  $A$  assumia que a velocidade era constante em linha reta.

Problemas Identificados:

Ao tentar rastrear um círculo, o filtro linear falhou sistematicamente. Como a velocidade vetorial em um círculo muda de direção constantemente (aceleração centrípeta), o filtro linear tendia a "sair pela tangente" a cada passo.

Para tentar corrigir isso, aumentamos artificialmente o ruído do processo ( $Q$ ), o que tornou o filtro instável e "trêmulo", resultando em melhoria de precisão negativa ou irrelevante.



### 3. Abordagem 2: A Evolução para EKF

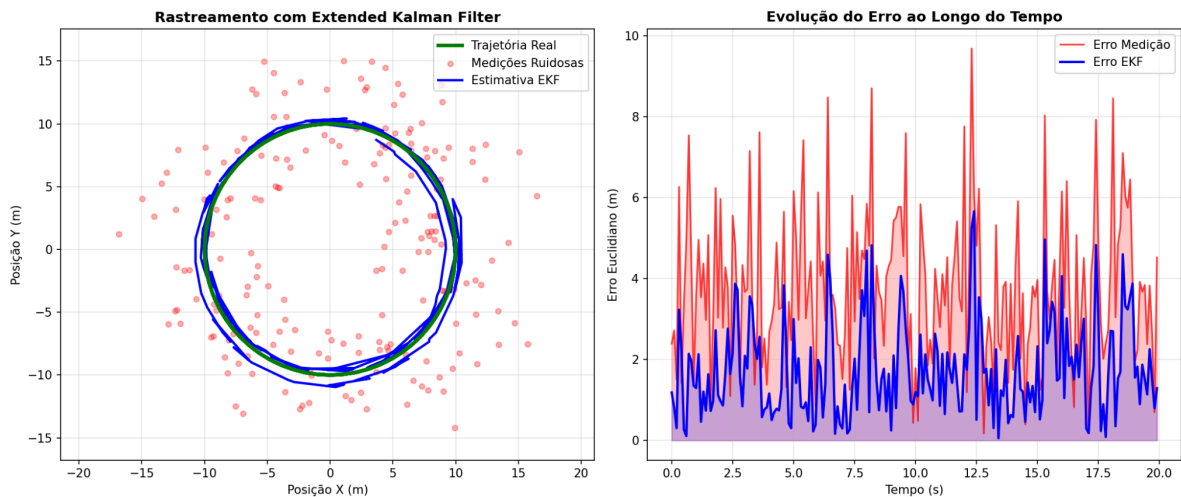
Para solucionar as falhas da versão linear, evoluímos para o Extended Kalman Filter (EKF).

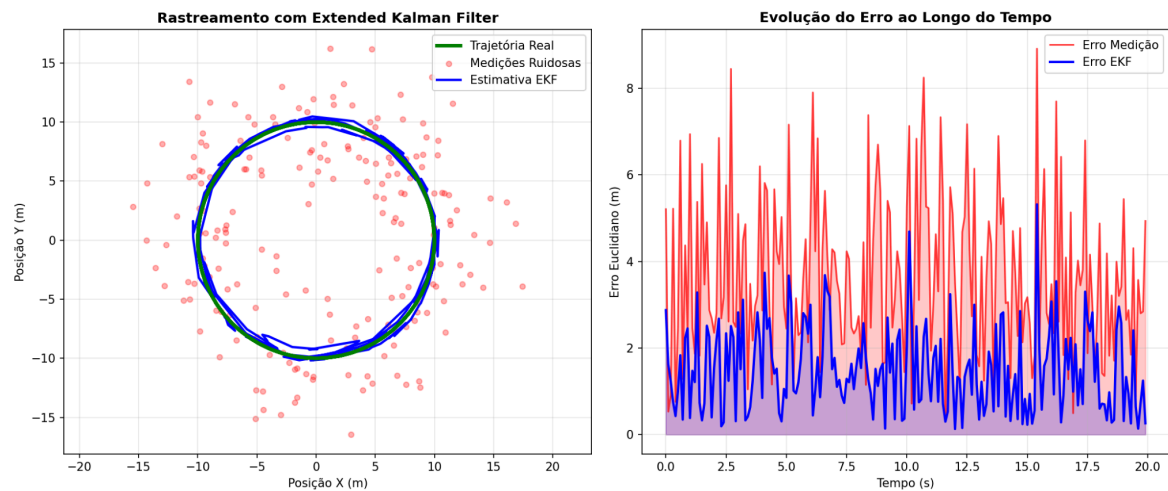
Melhorias de Modelagem:

1. Mudança de Coordenadas: O estado interno foi alterado de Cartesiano  $[x, y]$  para Polar [raio, ângulo, velocidade angular]. Isso é crucial, pois um movimento circular é "linear" no espaço polar (o ângulo cresce de forma constante).
2. Linearização (Jacobianas): Implementamos as matrizes Jacobianas  $F$  (dinâmica) e  $H$  (observação) para calcular como a incerteza se propaga através das funções não-lineares de conversão polar-cartesiana.

Resultado da Evolução:

O filtro passou a "entender" a natureza curva do movimento. Em vez de lutar contra a mudança de direção, o modelo EKF antecipa a curva naturalmente através da evolução do ângulo theta.





## 4. Resultados Comparativos

A comparação quantitativa evidencia o salto de qualidade entre as versões.

Comparativo de Erro (MSE - Mean Squared Error):

1. Cenário com Filtro Linear (Tentativa 1):

- Erro das Medições: 9.2377
- Erro do Filtro: 33.5232 (Pior que o sensor)
- Melhoria: -262.90% (O filtro atrapalhou o rastreamento)

2. Cenário com EKF (Versão Final):

- Erro das Medições: 9.1070
- Erro do Filtro: 1.7995
- Melhoria: +80.24%

Análise Visual:

Enquanto o gráfico da Versão 1 mostrava uma linha "derrapando" para fora do círculo ou oscilando violentamente, o gráfico da Versão 2 (EKF) mostra uma linha suave e aderente à trajetória real (Ground Truth), filtrando o ruído sem perder a referência da curva.

## 5. Guia de Instalação e Execução

Para reproduzir a simulação:

Pré-requisitos

- Python 3.12 ou superior.
- Bibliotecas: numpy, matplotlib, sklearn.

### Passo 1: Configuração do Ambiente

Recomenda-se o uso de um ambiente virtual para isolar as dependências.

No Windows (PowerShell):

PowerShell

# 1. Crie o ambiente virtual

```
python -m venv venv
```

# 2. Ative o ambiente

```
.\venv\Scripts\activate
```

No Linux/Mac:

Bash

```
python3 -m venv venv
```

```
source venv/bin/activate
```

### Passo 2: Execução do Código

Com o ambiente ativado, execute o script principal.

Bash

```
python kf-robo.py
```

### Passo 3: Saída

O terminal exibirá o cálculo do erro quadrático médio (MSE).

Um gráfico 'kalman\_tracking.png' será salvo, ilustrando a fusão de dados.

## 6. Conclusão

A evolução deste projeto demonstrou uma lição fundamental em Robótica Probabilística: a sofisticação do algoritmo (EKF vs KF Linear) é inútil sem uma modelagem correta da física do problema.

A simples troca de coordenadas Cartesianas para Polares, permitida pelo uso do EKF, transformou um problema intratável para um filtro linear em uma tarefa trivial, resultando em um sistema de rastreamento de alta precisão capaz de reduzir o erro dos sensores.