

## Relatório Final de Iniciação Científica

# Sistema de interceptação de bolas no Futebol de Robôs - SmallSize League.

Orientador: Prof. Dr. Flavio Tonidandel  
Departamento: Departamento de Ciência da Computação  
Bolsista: Fernando Rodrigues Junior  
N° FEI: 12.111.282-5



# Centro Universitário da FEI

## Projeto de pesquisa



### RESUMO

O projeto tem como premissa a evolução do sistema de interceptação de bolas dos robôs da categoria SmallSize de futebol de robôs da equipe RoboFEI. A aplicação desse estudo irá ampliar a dinâmica entre robô e bola, capacitando uma melhor interação e ampliação do domínio de bolas paradas e em movimento durante o jogo. A ideia central é utilizar métodos de iteração e recursividade para resolver problemas de atrasos durante a recepção e processamento das informações oriundas de sensores. Os atrasos acabam por dificultar a localização real dos objetos, dessa forma o uso de técnicas de *tracking* (e.g. Filtro de Kalman) serão úteis para estimar os estados futuros da bola e do robô, reduzindo o efeito negativo dos *delays*. Através de um modelo cinemático aproximado dos robôs e da bola, será possível estimar trajetórias e tempos para atingir determinados destinos e com isso avaliar um ponto de encontro entre o robô e a bola. O destino ou ponto de encontro pode pertencer à própria trajetória em que a bola está inserida ou outro caminho que tenha como objetivo, por exemplo, resgatar a bola evitando esbarrá-la. A interceptação depende de alguns fatores como a velocidade da bola e do robô, ponto de destino do robô e o tempo para atingir o destino, este pode ser determinado através de cálculos iterativos seguidos de comparações entre tempos e deslocamentos.

A interceptação de bola aumenta a competitividade nos jogos de futebol de robôs, ter a posse de bola mais vezes e mantê-la por mais tempo no domínio da equipe possibilita a recepção de passes e cobranças indiretas com a participação de um maior número de robôs em lances e jogadas inteligentes. O presente estudo apresenta uma solução de simples implementação com ótimo desempenho computacional e que possibilita ao robô aumentar as chances de interceptação de bolas com maior velocidade.

Palavras-chave:

1. Interceptação de bolas no Futebol de Robôs
2. Filtro de Kalman
3. Tracking de objetos
4. Processos Iterativos na avaliação de cenário futuro



# Centro Universitário da FEI

## Projeto de pesquisa



## Índice

RESUMO.....	2
I. INTRODUÇÃO.....	4
II. A PESQUISA.....	4
II.1. Objetivo .....	4
II.2. Justificativa .....	5
III. REVISÃO BIBLIOGRÁFICA .....	6
IV. ESTUDO DO PROJETO .....	7
IV.1. Método de captação dos dados .....	7
IV.2. Análise dos dados .....	7
IV.3. A bola e suas características.....	7
IV.3.1. Tratamentos dos dados .....	9
IV. 3.1.1. Filtro de média deslizando .....	10
Resultados Empíricos do Filtro de Média Deslizando.....	10
IV. 3.1.2. Filtro de Kalman.....	11
IV. 3.1.3. Implementação do Filtro de Kalman .....	12
Matriz de Estados .....	12
Modelo do sistema .....	12
Estrutura e Diagrama de Blocos do Filtro de Kalman .....	14
Resultados Empíricos do Filtro de Kalman .....	15
IV. 4. Robôs e suas características .....	17
Modelagem através do Método dos Mínimos Quadrados.....	18
IV. 5. Algoritmo de Interceptação.....	20
V. RESULTADOS E CONCLUSÕES FINAIS.....	24
VI. PLANO DE TRABALHO FUTURO .....	26



# Centro Universitário da FEI

## Projeto de pesquisa



## I. INTRODUÇÃO

O jogo de futebol de robôs [1] da categoria SmallSize - RoboCup [2] é caracterizado por sua dinâmica, possui lances rápidos e movimentos precisos que fazem dele uma disputa entre quem consegue manter o domínio da bola por mais tempo.

O termo interceptação no contexto de um jogo de futebol é definido como um movimento realizado com o objetivo de ganhar o controle da bola. Isso inclui interceptar passes e resgatar bolas que não necessariamente pertençam a alguma das equipes no momento, mas que esteja livre no campo.

Interceptar uma bola no mundo real consiste em saber o momento certo de agir, com qual velocidade chegar e onde se posicionar. No jogo dos robôs, o objetivo é o mesmo, contudo eles devem estar preparados para avaliar o melhor momento de se moverem, com qual velocidade e em qual ponto eles devem se posicionar para que consigam alcançar a bola.

Saber se é possível ou não chegar na bola em movimento depende de conhecer tanto o comportamento dos robôs quanto da bola, o conhecimento da cinemática do robô como velocidade e aceleração são imprescindíveis, já que através desses parâmetros é possível obter o tempo que o robô leva para se deslocar de uma posição a outra.

A bola também possui características fundamentais, seu modelo, assim como do robô, deve fornecer dados importantes como sua velocidade e aceleração que servirão de base para encontrar o ponto de interceptação.

## II. A PESQUISA

### II.1. Objetivo

O presente estudo tem como objetivo central o desenvolvimento de um algoritmo de interceptação de bola que seja aplicado ao time de futebol de robôs - RoboFEI, do Centro Universitário da FEI [3].

O principal desafio na interceptação de bolas é aplicar um método eficiente que seja capaz de determinar o ponto mais próximo possível em que o robô poderá encontrar a bola sem que ela saia da sua zona de atuação. Especialmente nos momentos em que não há a posse da bola, é muito importante para a dinâmica do jogo conseguir interceptá-la o maior número de vezes, aumentando as possibilidades do uso de jogadas inteligentes como passes e chutes diretos.

Primeiramente, a ideia é determinar o tipo de movimento a que os corpos estão submetidos, isto é, qual é a equação ou modelo que rege o comportamento cinemáticos dos dois objetos.

Diante das equações, determinar as grandezas como espaço, velocidade e tempo pode ser possível através da resolução de cálculos, possibilitando a resolução do problema de forma analítica.

O método de interceptação consiste em um processo de iteração entre bola e robô de forma que seja possível projetar o movimento da bola em uma série de pontos futuros segundo seu modelo

cinemático, a partir daí pode-se fazer uma comparação entre o tempo em que a bola levará para atingir o próximo ponto estimado e o tempo em que o robô levará para chegar a este mesmo ponto. Assim que o tempo de movimentação do robô for menor que o tempo de movimentação da bola, indica que a interceptação é possível e o ponto de encontro é estabelecido, conforme Figura 1.

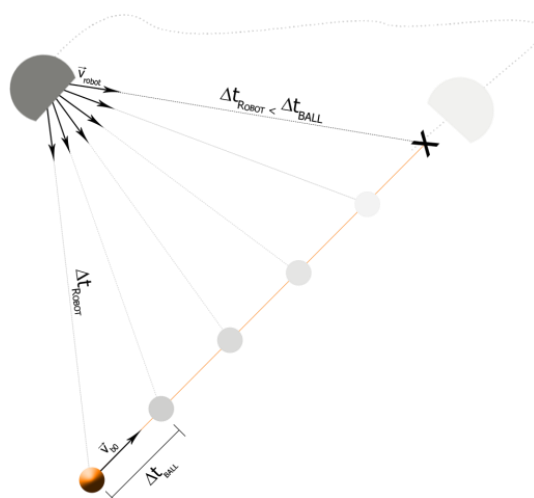


Figura 1. Processo iterativo para determinar o ponto de encontro entre bola e robô.

## II.2. Justificativa

Um sistema de interceptação eleva a capacidade de interação entre os robôs de forma a possibilitar jogadas inteligentes por meio de passes e dribles e abre perspectivas para a criação de novas técnicas e habilidades, como o chute ao gol direto, conhecido popularmente como chute de primeira. O sistema de interceptação abre caminhos para respostas mais eficientes como jogadas ensaiadas e um maior poder de domínio de bola.

A interceptação é um dos entraves do jogo, obter sucesso na recepção e domínio de bolas em movimento é uma tarefa difícil, isso faz com que a equipe que tenha tal habilidade se destaque e tenha um desempenho melhor nas partidas. Dentro da perspectiva do projeto RoboFEI, o objetivo é aumentar a complexidade dos movimentos dos robôs com a bola e abrir caminho para novas aplicações.

### III. REVISÃO BIBLIOGRÁFICA

De acordo com a equipe da Universidade Carnegie Mellon [4], a ideia é considerar uma linha entre o robô e o alvo a fim de determinar em qual ponto a bola irá interceptar essa linha, com isso o robô poderá ir para o ponto ao mesmo tempo em que a bola se aproxima.

Primeiro é necessário estimar esse ponto e, para isso há um método interessante que utiliza a comparação entre os tempos de deslocamento da bola e do robô. Quando há o conhecimento da aceleração e desaceleração de ambos, é possível, iterativamente, prever suas posições com o decorrer do tempo. Caso exista um ponto dentro da trajetória da bola em que o robô consiga chegar primeiro que ela, esse será o ponto em que a bola poderá ser interceptada [5].

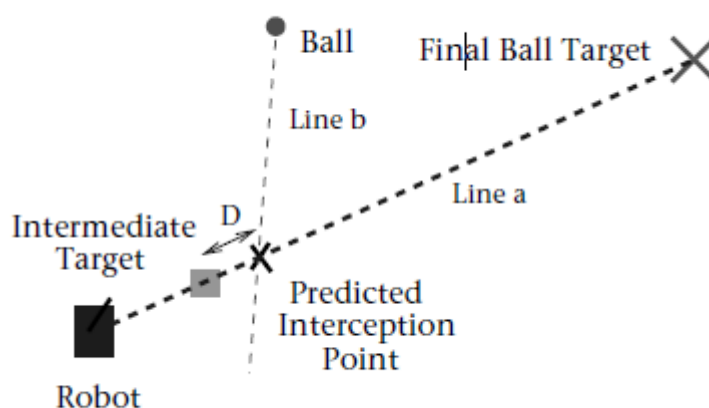


Figura 2 - Ball Interception (recepção de passe)

Alunos da Universität Bremen aplicaram esse conceito e utilizaram as equações cinemáticas da física clássica para estimarem os pontos futuros da bola e do robô [6]. Consideraram a relação linear entre a velocidade e o tempo, uma aceleração constante da bola e do robô [6] conforme as equações a seguir:

$$S_{ball} = v_{ball}\Delta t + \frac{1}{2}a_{ball}(\Delta t)^2 \quad (1)$$

$$S_{robot} = v_{robot}\Delta t + \frac{1}{2}a_{robot}(\Delta t)^2 \quad (2)$$

A condição de encontro pode ser feita também através da correspondência entre espaço final da bola e do robô, dessa forma é possível determinar em qual tempo  $t$  ocorrerá à interceptação.

Vale destacar que neste modelo são feitas algumas considerações:

1. O robô possui aceleração e desaceleração de mesmo módulo (isso não ocorre na prática com os robôs da equipe RoboFEI).
2. Não são considerados os momentos em que o robô possui velocidade constante (após atingida a velocidade máxima).



## IV. ESTUDO DO PROJETO

### IV.1. Método de captação dos dados

O futebol de robôs utiliza um campo oficial de 3025mm x 2025mm, cujos os robôs podem se mover livremente. Os robôs possuem 180mm de diâmetro e 150mm de altura e usam uma bola de golfe de 21mm de diâmetro para jogar. Para a captação dos objetos no campo são utilizadas câmeras AVT Stingray F-046C com uma taxa de atualização de 60fps (frames por segundo).

Cada robô possui uma sequência de 5 cores, sendo 4 delas para determinar a identidade e posição absoluta de cada robô no campo e uma cor para diferenciar as equipes. Um software de reconhecimento de imagem compara as cores obtidas pelas câmeras com um padrão e encontra as coordenadas x e y com a localização de cada objeto no campo. A partir daí os dados são empacotados e enviados para todas as equipes através da rede (e.g. broadcast).

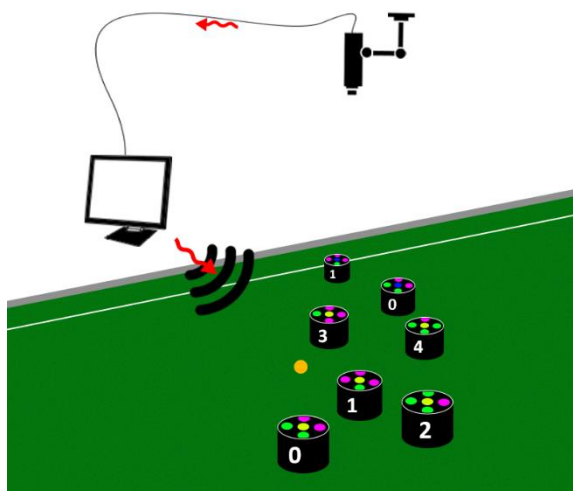


Figura 3. Esquema de funcionamento das câmeras e obtenção da localização dos objetos em campo.

### IV.2. Análise dos dados

Esta etapa tem como objetivo verificar qualitativamente os dados coletados pelas câmeras, verificar a quantidade de ruídos e determinar as linhas de tendência das funções geradas pela interpretação dos gráficos encontrados.

### IV.3. A bola e suas características

Foi aplicada uma força qualquer na bola e capturadas suas posições em coordenadas cartesianas até que a mesma atingisse velocidade nula ainda dentro do campo.

O seguinte experimento foi realizado em ambiente oficial, na competição brasileira de robótica (CBR2014), dentro do campo havia somente a presença de uma bola oficial para que não houvesse nenhum tipo de interferência na captação dos dados.

O gráfico 1 apresenta o resultado da captação dos dados do experimento demonstrando a variação da posição da bola em uma única dimensão e processada pelo computador a uma taxa



## Centro Universitário da FEI Projeto de pesquisa



de 60fps. Através dele é possível visualizar que a bola apresenta um movimento de característica quadrática, remetendo à física clássica newtoniana as equações que regem seu movimento deverão ser do tipo:

$$\text{Equação horária de segunda ordem: } S(t) = S_0 + V_0 * t + 0.5 * a * t^2$$

$$\text{Equação de velocidade de primeira ordem: } V(t) = V_0 + a * t$$

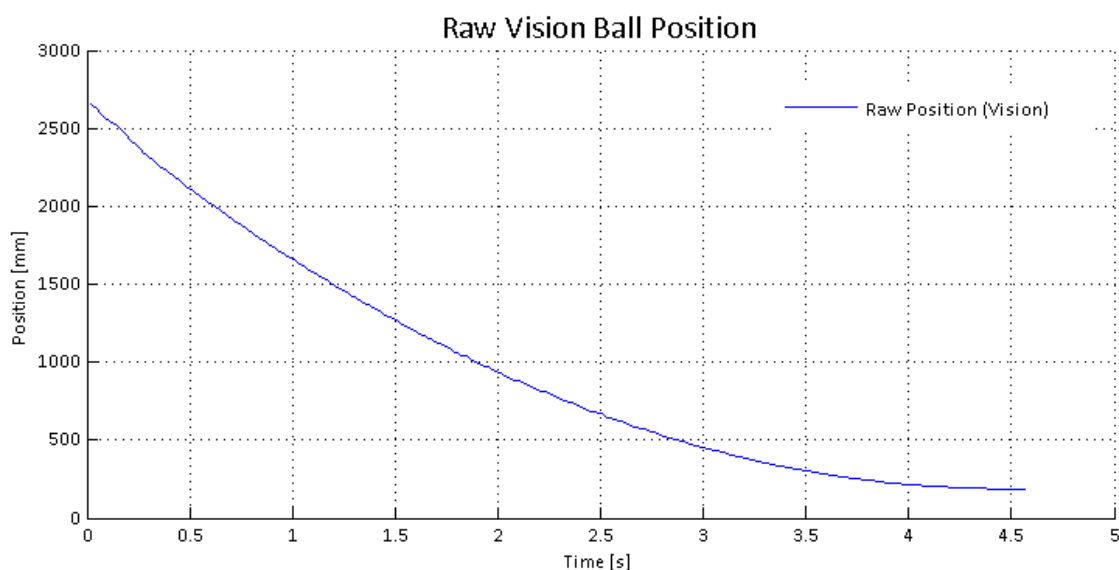


Gráfico 1. Posição da bola em função do tempo. Demonstra um movimento de característica quadrática.

A obtenção da velocidade da bola parte dos resultados do experimento anterior. Para encontrá-la, basta possuir as informações que relatem a posição da bola em intervalos consecutivos, isto é: em qual posição a bola está e em qual estava a um quadro atrás, conforme a seguir:

$$v(t) = S_{\text{Quadro Atual}} - S_{\text{Quadro Anterior}} \left[ \frac{\text{mm}}{\text{quadro}} \right] \quad (\text{I})$$

$$v(t) = v(t) * FPS \left[ \frac{\text{mm}}{\text{s}} \right] \quad (\text{II})$$

A velocidade calculada estará relacionada a um único quadro, dessa forma, multiplica-se o valor pela taxa de atualização da câmera (FPS) para obter a velocidade em segundos.

O resultado é um gráfico de velocidade em função do tempo (Gráfico 2), onde a velocidade negativa indica direção contrária da bola em relação ao eixo de posição, conforme abaixo:



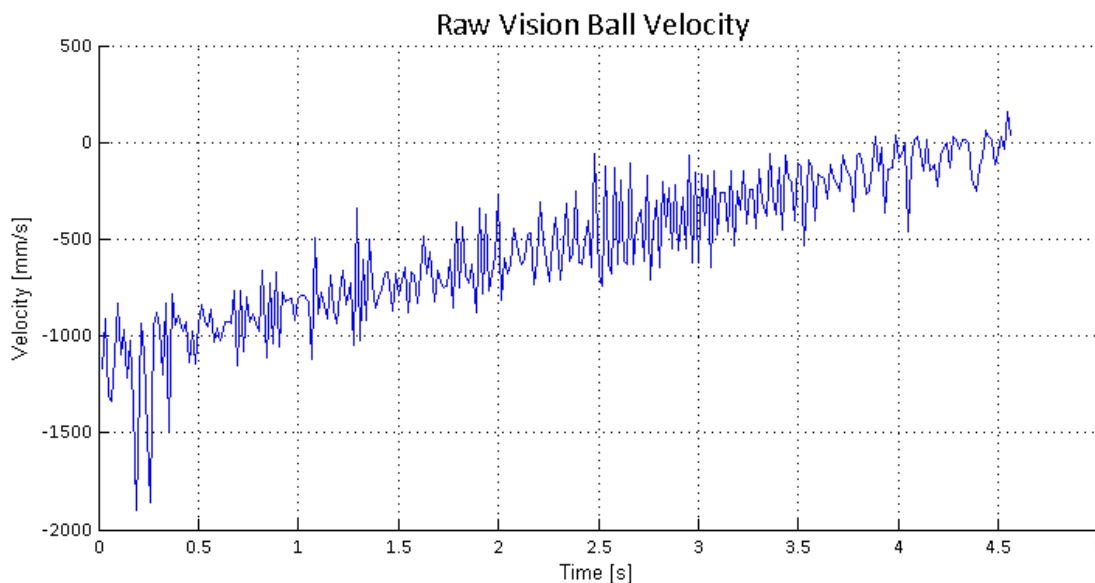


Gráfico 2. Velocidade em função do tempo utilizando a diferença de posições obtidas entre quadros consecutivos.

Analisando o gráfico, nota-se uma grande quantidade de ruídos. Esses ruídos devem-se às pequenas flutuações oriundas das informações de posição que ao serem acumuladas distorcem os dados causando incerteza e dificuldade na leitura das informações, além disso causam uma grande instabilidade no sistema de decisão dos robôs, já que muitas ações são determinadas a partir da velocidade da bola.

Uma das causas da presença desses ruídos está na própria captação dos dados. Algumas vezes a câmera acaba por capturar a mesma informação de posição da bola em quadros consecutivos pois a velocidade de captação é mais rápida que o deslocamento da bola. Utilizando o mesmo raciocínio usado para o cálculo da velocidade, teríamos uma velocidade zero em um dado instante, o que retrata uma informação inválida.

#### IV.3.1. Tratamentos dos dados

Tendo em vista os problemas enfrentados pelos ruídos, foi necessário encontrar uma solução que minimizasse a atuação desse fenômeno e aumentasse o nível de confiabilidade dos dados obtidos. A aplicação de filtros para a suavização e redução de falsos positivos é bem comum na área de estatística que por sua vez tem vasta aplicação na área da robótica.

Há uma gama muito grande de tipos de filtros (e.g. filtro de partícula, filtro de Kalman, filtro passa baixa/média móvel), dentre os quais cada um possui características diferentes e devem ser utilizados apropriadamente de acordo com cada aplicação [7].

Serão utilizados dois principais filtros: Moving Average filter ou filtro de média móvel e o Kalman Filter ou filtro de Kalman, sendo o segundo o foco do presente estudo.

#### IV. 3.1.1. Filtro de média deslizante

Uma primeira opção para melhorar os ruídos, seria a implementação de um filtro de média móvel simples, o filtro é um simples filtro passa baixa com resposta de impulso finito, utilizado com frequência para combater ruídos em dados obtidos por sensores.

Ele faz a amostra de uma quantidade de dados e faz a média desses dados produzindo como saída um único valor. Por exemplo, em uma amostra com elementos finitos, utilizando uma média deslizante com intervalo cinco, teremos a média dos cinco primeiros elementos gerando um único valor. A partir daí, para um novo conjunto de elementos, o primeiro da série anterior é descartado, fazendo com que a média se movimente ao longo do tempo.

Um filtro de média móvel reduz os ruídos substituindo cada informação de uma dada posição pela média das informações da vizinhança, conforme o intervalo do filtro [8]. Dado pela equação abaixo:

$$y_s(i) = \frac{1}{2N + 1} (y(i + N) + y(i + N - 1) + \dots + y(i - N))$$

Sendo  $y_s(i)$  o valor filtrado para o  $i$ ésimo ponto,  $N$  o número de pontos vizinhos a  $y_s(i)$  e  $2N+1$  representa o intervalo do filtro.

#### Resultados Empíricos do Filtro de Média Deslizante

Através do software MatLab [9] os dados de velocidade foram filtrados utilizando um filtro de média deslizante com intervalo de cinco. O resultado pode ser visto a seguir:

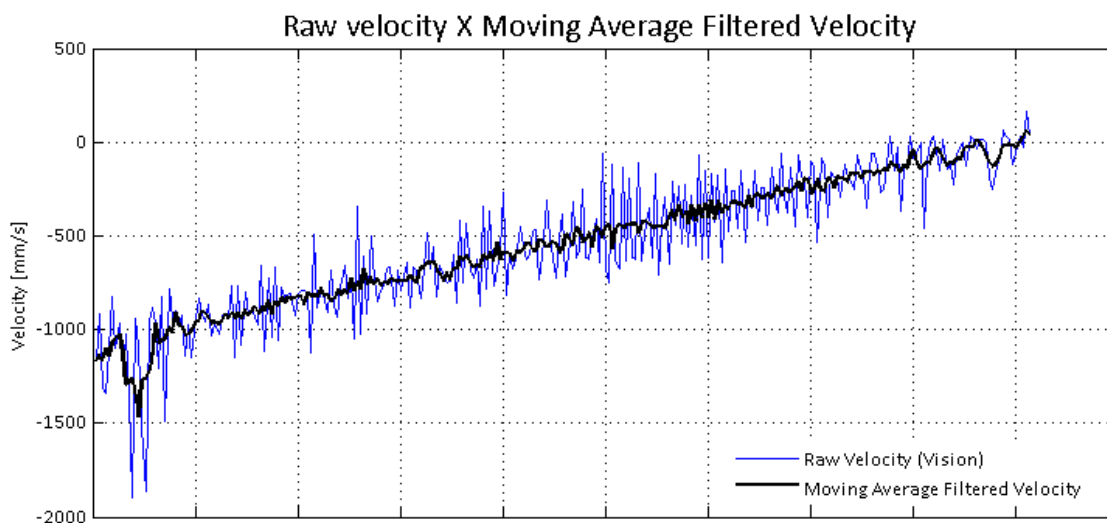


Gráfico 3. Velocidade em função do tempo após aplicar um filtro de média deslizante com intervalo de 5.



## Centro Universitário da FEI

### Projeto de pesquisa



Conforme o Gráfico 3, o filtro de média deslizando conseguiu reduzir grande parte dos ruídos que havia na velocidade da bola, entretanto ainda há variações indesejáveis na informação. É importante que a curva da velocidade seja o mais suave possível, já que para estimar o ponto de interceptação, principal objetivo desse estudo, deve-se ter informações confiáveis da posição e velocidade da bola.

Tendo em vista a necessidade de uma maior confiabilidade nos dados e a possibilidade de estimar futuros estados, o filtro de Kalman parece a ferramenta ideal para a aplicação e será descrito a seguir.

#### IV. 3.1.2. Filtro de Kalman

É importante destacar que o presente estudo tem como foco a aplicação do filtro de Kalman nos dados obtidos das câmeras, sendo assim, não há a pretensão de um aprofundamento teórico no assunto.

O filtro foi inventado por R.E. Kalman e publicado pela primeira vez em 1960 [10]. É um método utilizado para estimar um estado de um sistema baseado em leituras recursivas de informações ruidosas. O filtro processa todas as informações fornecidas à ele (e.g. sensores) para obter uma estimativa em relação à uma variável de interesse. Ele pode ser encontrado em diversas formas incluindo variantes contínuas e discretas no tempo, além disso pode ser aplicado para a resolução de problemas lineares e não lineares (Filtro de Kalman Extendido).

A ideia central do filtro de Kalman é modelar um sistema de interesse em um sistema linear dinâmico dotado de ruído de processo e também ruído oriundo dos sensores. Ao determinar a natureza desses ruídos, é possível construir uma estimativa do sistema mesmo que os dados obtidos sejam incertos [11].

É um filtro recursivo, ele não necessita de todo um histórico de dados anteriores para funcionar, utiliza a *priori* o estado atual do sistema, resultando em praticidade em sua implementação. [12]

O seu funcionamento é baseado nas seguintes principais concepções:

1. Conhecimento do sistema e o uso do modelo dinâmico característico do objeto/fenômeno, isto é, como o objeto se comporta na natureza.
2. A descrição estatística dos ruídos do sistema levando em conta a incerteza dos sistemas dinâmicos, quão deformadas estão as informações que irão servir de alimentação para o filtro.
3. Informações sobre condições de início da variável de interesse.

O filtro consiste em duas principais etapas:

1. Predição
2. Atualização

A primeira etapa representa a estimativa do estado futuro baseado no modelo dinâmico do objeto estudado, para isso há a necessidade de conhecer as características físicas do objeto, conforme foi analisado em IV. 4.3.1.

A segunda etapa consiste na adição da informação proveniente da observação (e.g. câmeras) e atualiza o estado estimado na primeira etapa. Ambas etapas possuem uma variável de covariância que servirá de “balança” para medir o grau de confiança entre as fases e determinar com sucesso o estado estimado.

O processo é recursivo, portanto as duas etapas se repetem representando estados atuais e futuros do sistema conforme pode ser visto na Figura4.

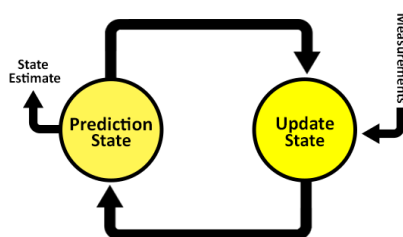


Figura 4. Recursividade do Filtro de Kalman apresentando as duas principais fases, predição e atualização.

O filtro de Kalman Estendido (EKF) é uma variação do filtro de Kalman e é utilizado quando há a necessidade de tratar problemas que possuem características não lineares, por exemplo a velocidade da bola. Sendo assim, a pesquisa tem como finalidade a aplicação do EKF.

#### IV. 3.1.3. Implementação do Filtro de Kalman

##### Matriz de Estados

É na matriz de estados que estarão localizadas as variáveis de interesse do filtro, ela descreve a dinâmica dos estados do sistema. Podem representar grandezas como velocidade, posição, ângulos, etc. Nesse contexto, a matriz de estados para a bola irá conter posição e velocidade, ambas para duas dimensões no espaço. Assim:

$$x_k = \begin{bmatrix} x \\ y \\ \dot{x} \\ \dot{y} \end{bmatrix}, \text{ sendo } x, y \text{ coordenadas da posição da bola e } \dot{x}, \dot{y} \text{ das velocidades.}$$

A matriz de estados possui dois valores ao mesmo tempo, isto é, um valor *a priori*, valor estimado antes da atualização e outro *a posteriori*, o valor corrigido após o estado de atualização. Por convenção, será utilizado  $\hat{x}_k^-$  para valores *a priori* e  $\hat{x}_k$  para valores *a posteriori*.

##### Modelo do sistema

O modelo dinâmico descreve a transformação da matriz de estados ao longo do tempo. Geralmente é representada por uma equação diferencial do sistema.

Quando a bola rola livremente, há a presença de atrito como uma força de retardo constante, sendo assim o modelo fica:



## Centro Universitário da FEI

### Projeto de pesquisa



$$\begin{aligned}\hat{x}_k^- &= \hat{x}_{k-1} + \dot{x}_{k-1}\Delta t + 0.5 a_x \Delta t^2 \\ \hat{y}_k^- &= \hat{y}_{k-1} + \dot{y}_{k-1}\Delta t + 0.5 a_y \Delta t^2 \\ \dot{x}_{k-1}^- &= \dot{x}_{k-1} + a_x \Delta t \\ \dot{y}_{k-1}^- &= \dot{y}_{k-1} + a_y \Delta t\end{aligned}$$

Matricialmente, tem-se:

$$\hat{x}_k^- = A\hat{x}_{k-1} + ac_k = \begin{pmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \hat{x}_{k-1} + \begin{pmatrix} 0.5 a_x \Delta t^2 \\ 0.5 a_y \Delta t^2 \\ a_x \Delta t \\ a_y \Delta t \end{pmatrix}$$

Tem-se:

$$a = \begin{cases} -a_{fat} \cos \delta, & |v| > a_{fat} \Delta t \\ \frac{-v}{\Delta t}, & \text{caso contrário} \end{cases}$$

Sendo  $\delta$  o ângulo de movimentação da bola,  $v$  a velocidade da bola e  $a_{fat}$  uma aceleração constante determinada empiricamente que representa o atrito entre a bola e o carpete do campo [13].

Serão consideradas as equações abaixo para estimativa de estado futuro considerando as covariâncias e os devidos ganhos.

#### Time Update (Prediction State)

Estima o estado futuro

$$\hat{x}_k^- = f(\hat{x}_{k-1}, u_k)$$

Estima o erro de covariância futuro

$$P_k^- = A_k P_k A_k^T + W_k Q_k W_k^T$$

#### Measurement Update (Update State)

Calcula o ganho de Kalman

$$K_k = P_k^- H_k^T (H_k P_k^- H_k^T + V_k R_k V_k^T)^{-1}$$

Atualiza a estimativa com as medidas do sensor

$$\hat{x}_k = \hat{x}_k^- + K(z_k - h(\hat{x}_k^-))$$

Atualiza o erro de covariância

$$P_k = (I - K_k H_k) P_k^-$$

Sendo

$\hat{x}_k^-$  = Previsão *a priori* do estado futuro

$\hat{x}_{k-1}$  = Estado anterior do sistema

$f$  = Matriz de transição ou modelo dinâmico do sistema

$A_k$  = Matriz jacobiana de transição ou modelo dinâmico do sistema

$H_k$  = Matriz jacobiana do modelo dinâmico do sensor/observação

$P_k$  = Matriz de covariância do erro do estado atual

$P_k^-$  = Matriz de covariância do erro do estado passado

$K_k$  = Kalman Gain (Ganho de Kalman) no estado futuro

$K$  = Kalman Gain (Ganho de Kalman) no estado atual

$z_k$  = Matriz de Dados medidos/obtidos pelos sensores

$I$  = Matriz identidade

$W_k$  = Ruído de processo com média zero e distribuição normal

$R_k$  = Matriz de covariância de observação (ruído)

$V_k$  = Ruído de observação com média zero e distribuição normal

$\hat{x}_k$  = Previsão *a posteriori* do estado futuro

$u_k$  = Matriz de comando

$h$  = Matriz do modelo dinâmico do sensor/observação

$Q_k$  = Matriz de covariância do processo (ruído)

### Estrutura e Diagrama de Blocos do Filtro de Kalman

O filtro de Kalman foi implementado no software do RoboFEI utilizando linguagem de programação C++. Para os cálculos matriciais, foi implementada e utilizada uma library contendo métodos otimizados para a resolução de problemas envolvendo álgebra linear. Com o Eigen [14], foi possível implementar o filtro de Kalman de maneira eficaz e com um bom desempenho na etapa de resolução de cálculos. O diagrama funcional de blocos que demonstra a implementação do Filtro de Kalman Estendido pode ser visto a seguir na Figura 5.

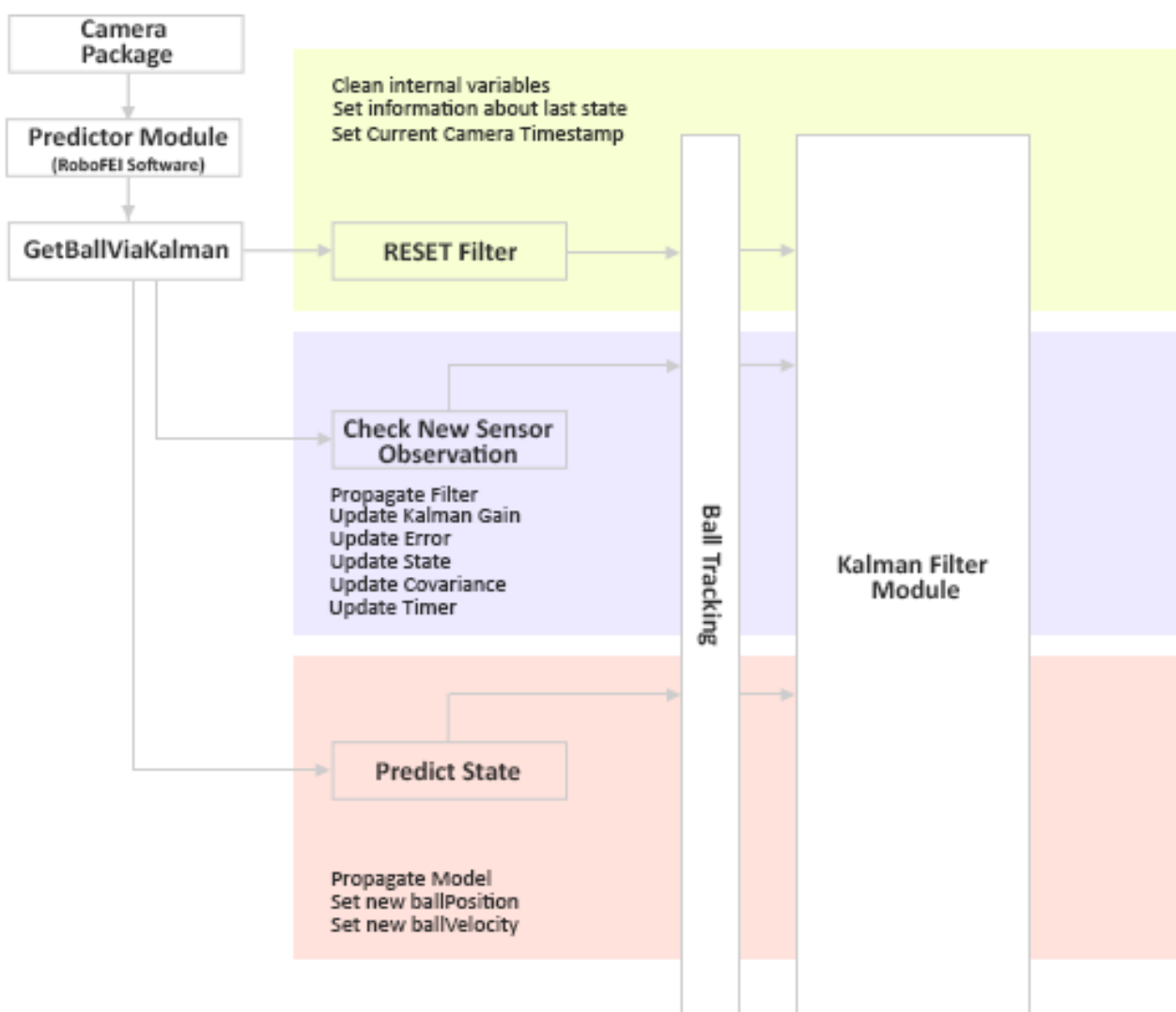


Figura 5. Diagrama funcional de blocos com a implementação do Filtro de Kalman no software do RoboFEI

### Resultados Empíricos do Filtro de Kalman

O filtro de Kalman resolveu o problema dos ruídos de forma satisfatória, conseguiu filtrar os dados com eficiência além de determinar a velocidade da bola com maior precisão, o resultado pode ser visualizado através do gráfico abaixo:

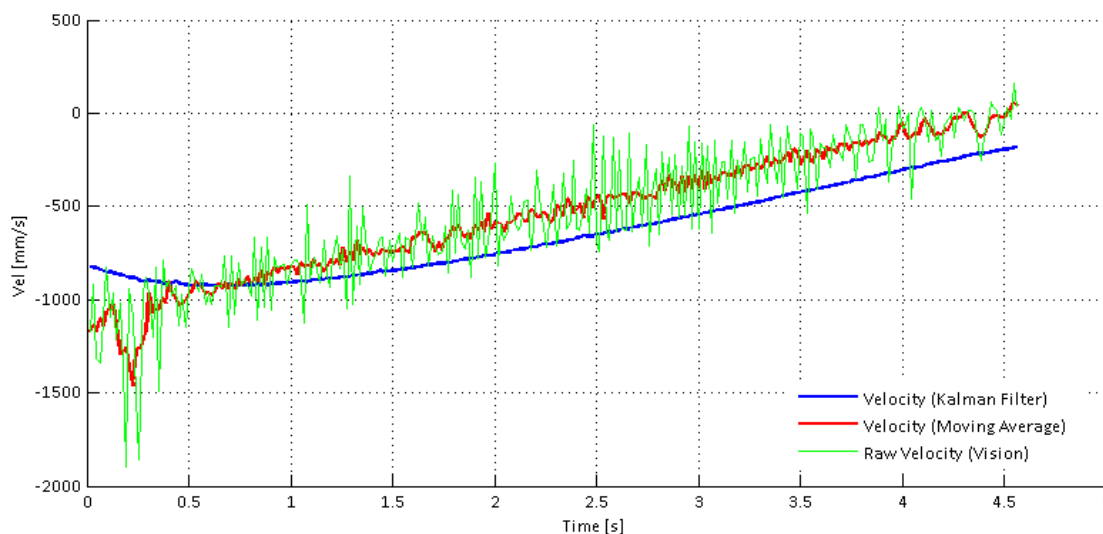


Gráfico 4. Resultado final do Filtro de Kalman na determinação da velocidade da bola. A partir do gráfico é possível ver a evolução dos filtros com relação ao tratamento dos dados.

O gráfico em verde demonstra a obtenção da velocidade através da diferença entre posições captadas pela câmera. Esse método apresenta muito ruído e impossibilita a obtenção de uma medida de velocidade com o mínimo de confiabilidade. A resposta em vermelho demonstra a velocidade da bola após passar por um filtro de média deslizante. O resultado é mais satisfatório que o anterior, entretanto os ruídos ainda podem atrapalhar quando se faz necessária uma resposta sem grandes oscilações. Finalmente, a resposta em azul demonstra a saída do filtro de Kalman, com oscilações praticamente inexistentes, entretanto com uma pequena variação no módulo da velocidade com relação aos dois métodos anteriores. Isto deve-se a calibração das matrizes de covariância que têm como papel equilibrar os dados dos sensores com a posição estimada através da matriz dinâmica do sistema.

Além do Filtro de Kalman solucionar o problema dos ruídos, ele ainda pode servir para estimar a posição da bola no futuro. Para que isso seja possível é necessário que a fase de predição seja executada proporcionalmente ao número de quadros que se deseja estimar a bola. Por exemplo, caso a frequência de processamento do software seja de 60Hz, a cada  $1/60s$  ( $\sim 16ms$ ), o filtro será atualizado com a posição proveniente da câmera.

Sendo assim, caso haja a necessidade de estimar a bola algumas posições à frente no tempo, basta processar a fase de predição na proporção de  $n/60$ , isto é: para estimar a bola em 30 quadros à frente deve-se processar a predição 30 vezes.

Ao deslocar a bola na direção do eixo das ordenadas, sua posição foi estimada para quinze e trinta frames a frente do seu movimento, conforme a seguir:

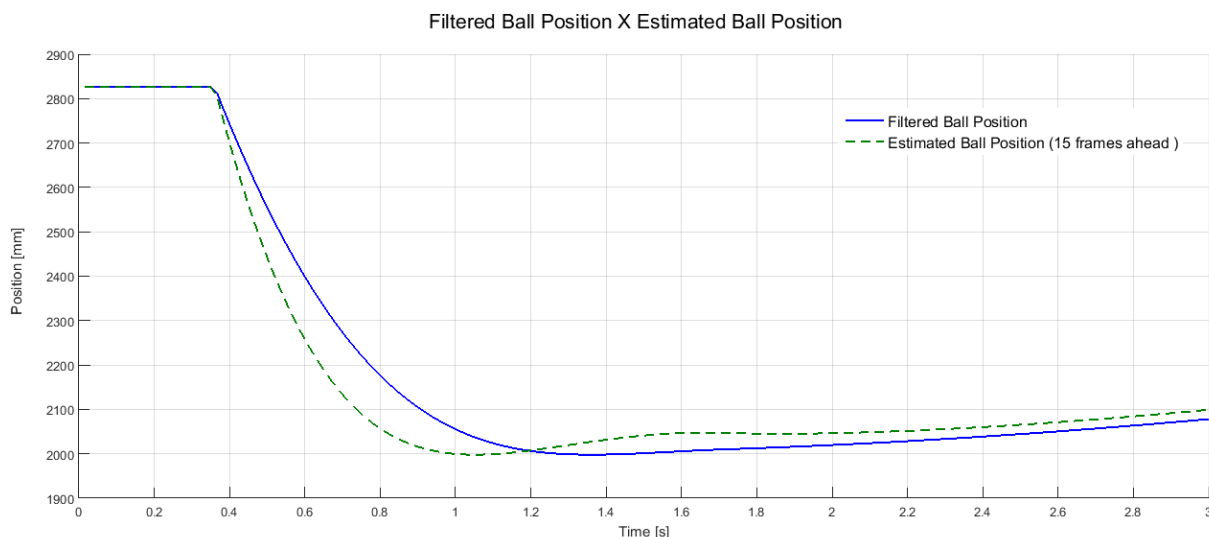


Gráfico 5. Posição estimada da bola 15 quadros à frente.

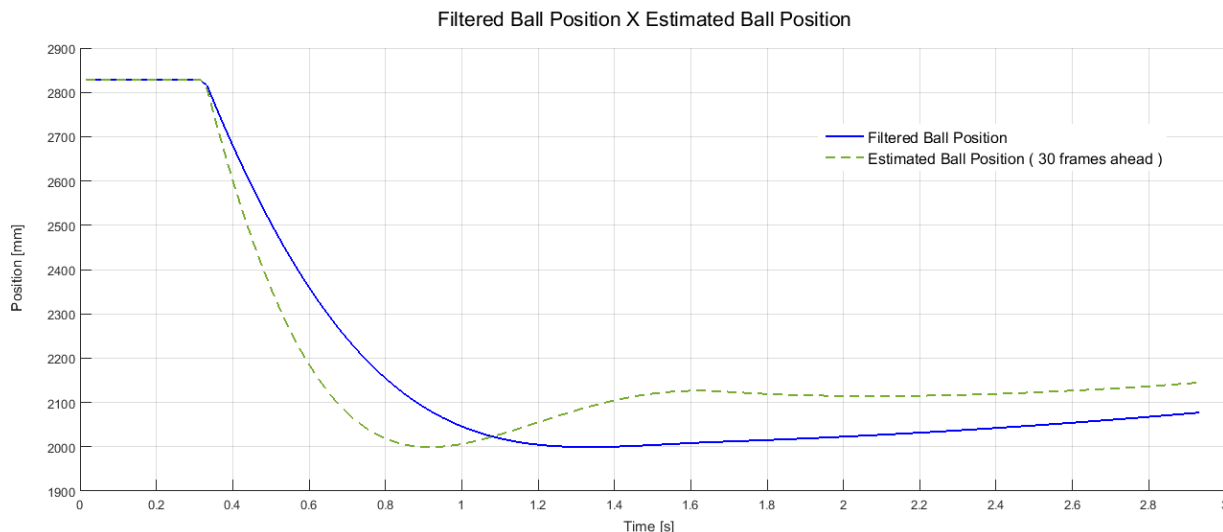


Gráfico 6. Posição estimada da bola 30 quadros à frente.

Analisando os gráficos é possível observar que quanto mais próximo do tempo de um quadro, mais confiável é a saída do filtro. Quanto maior o tempo de estimativa, mais tempo o sistema gastará rodando a fase de predição do filtro, nesse tempo a bola pode ter encontrado um obstáculo, entretanto não haverá correção por meio da fase de observação (onde há o recebimento dos sensores e atualização da posição real da bola), então a predição pode apresentar um erro significativo. O filtro como estimador reduz os efeitos de latência, pois é possível inserir a perda causada pelo *delay* na fase de predição do filtro.





#### IV. 4. Robôs e suas características

Assim como a bola, é necessário conhecer a dinâmica de movimento dos robôs e para isso foi feito um experimento que consistiu no deslocamento de um robô por um percurso linear de um lado a outro do campo (aproximadamente quatro metros). O experimento foi realizado em ambiente oficial, na competição brasileira de robótica (CBR2014), dentro do campo havia somente a presença de um robô para que não houvesse nenhum tipo de interferência na captação dos dados.

O Gráfico 7 apresenta o resultado da captação dos dados demonstrando a variação da velocidade do robô. Os dados foram processados pelo computador a uma taxa de 60fps e passaram por um filtro de média deslizante.

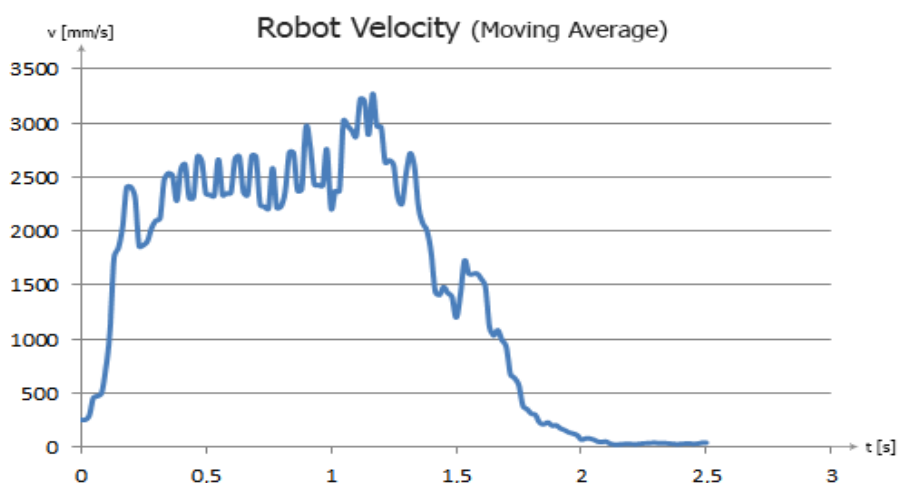


Gráfico 7. Velocidade do robô após a aplicação de um filtro de média deslizante de intervalo cinco.

A flutuação presente no Gráfico 7 deve-se, principalmente ao fato de a velocidade do robô ser calculada através da subtração entre dois quadros consecutivos. Uma pequena oscilação na posição do robô faz com que o valor calculado da velocidade se altere significativamente e perpetue o erro na composição dos próximos valores. O tratamento dos dados pode ser feito através de um filtro de Kalman, como discutido no tópico IV. 3.1.2.

Com os dados é possível determinar o perfil aproximado da velocidade do robô conforme pode ser visto no Gráfico 8. Ao visualizar o perfil nota-se que o robô apresenta dois diferentes comportamentos ao longo do tempo.

1. Aceleração constante (linear)
2. Aceleração nula
3. Desaceleração constante (linear)

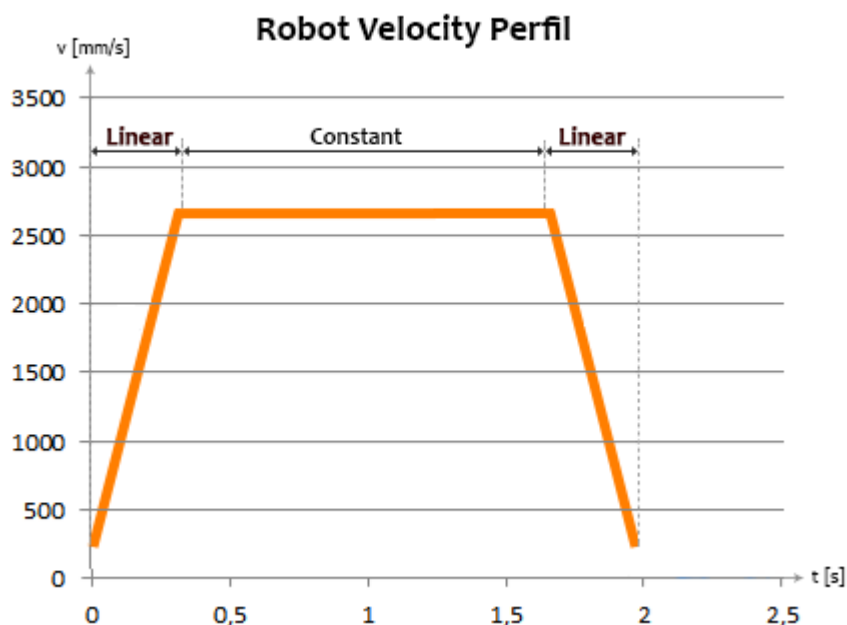


Gráfico 8. Perfil demonstrando a dinâmica teórica do robô ao longo do tempo

Conhecendo o perfil de aceleração, o próximo passo é determinar quais são as equações cinemáticas que regem esse movimento.

### Modelagem através do Método dos Mínimos Quadrados

O objetivo é utilizar o método dos mínimos quadrados para obter uma função/modelo aproximado que seja capaz de representar a dinâmica do robô e possibilite a extração de seus parâmetros de aceleração.

O método numérico foi escolhido devido à dificuldade em modelar o robô algebricamente, há muitas variáveis como o atrito entre as engrenagens, atrito entre o feltro e as rodas, torque dos motores e deslizamentos que são difíceis de serem mensurados.

A expressão utilizada para obter a função linear através do método dos mínimos quadrados está representada a seguir [15]:

$$y = a + bx_i \quad (1)$$

$$\begin{cases} na + \left(\sum_{i=1}^n x_i\right)b = \sum_{i=1}^n y_i \\ \left(\sum_{i=1}^n x_i\right)a + \left(\sum_{i=1}^n x_i^2\right)b = \sum_{i=1}^n x_i y_i \end{cases} \quad (2)$$



## Centro Universitário da FEI Projeto de pesquisa



Nota-se que ao determinar as expressões de velocidade o parâmetro de aceleração e desaceleração serão conhecidos e, portanto, será possível utilizar as leis de Newton para compor equações que poderão fornecer informações de posição e velocidade do robô em qualquer instante de tempo.

Para simplificar a análise, foi obtido um perfil de velocidade do robô através de um ambiente de simulação, conforme gráfico abaixo:

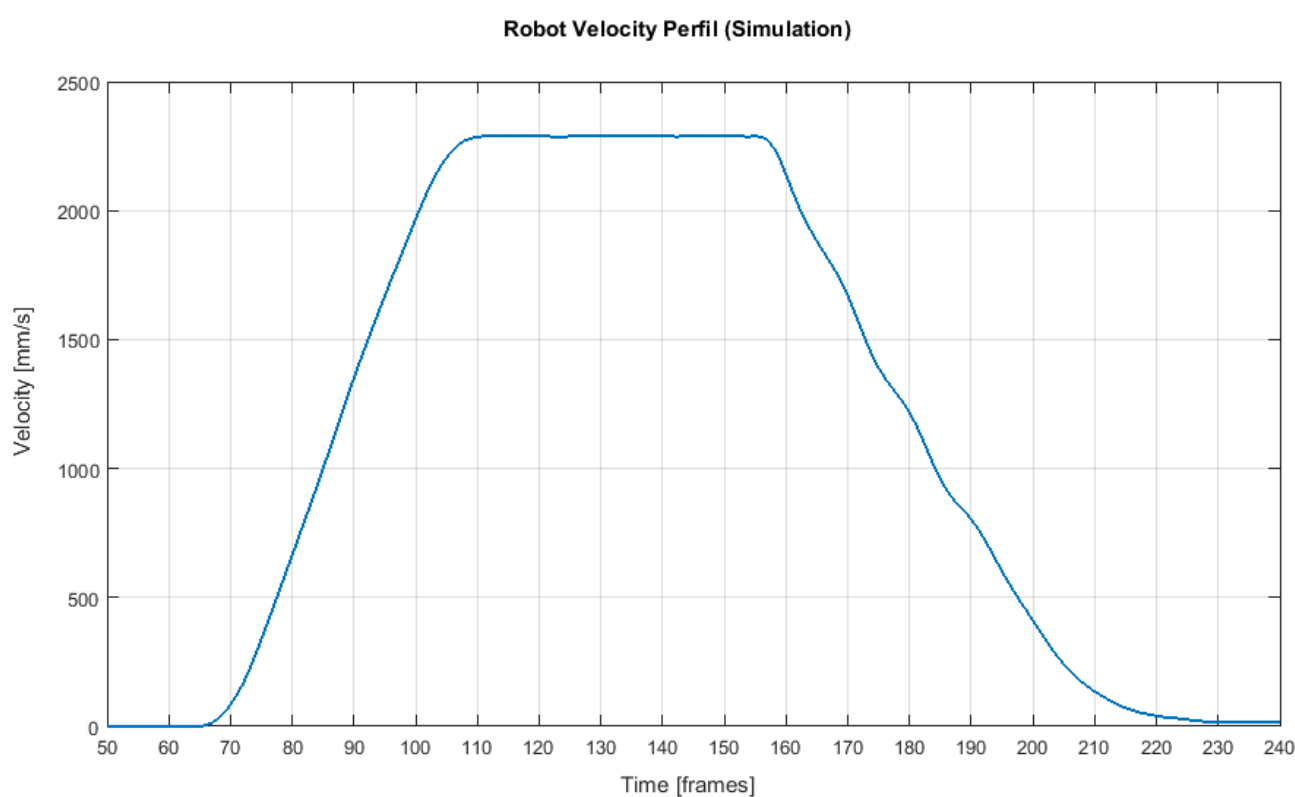


Gráfico 9. Perfil de velocidade do robô em ambiente de simulação.

Os dados foram separados de acordo com cada comportamento e inseridos no Excel, já que no software há funções que aplicam o método dos mínimos quadrados de maneira simples e rápida. O resultado pode ser visto na Figura 6.

Foi possível obter as seguintes equações:

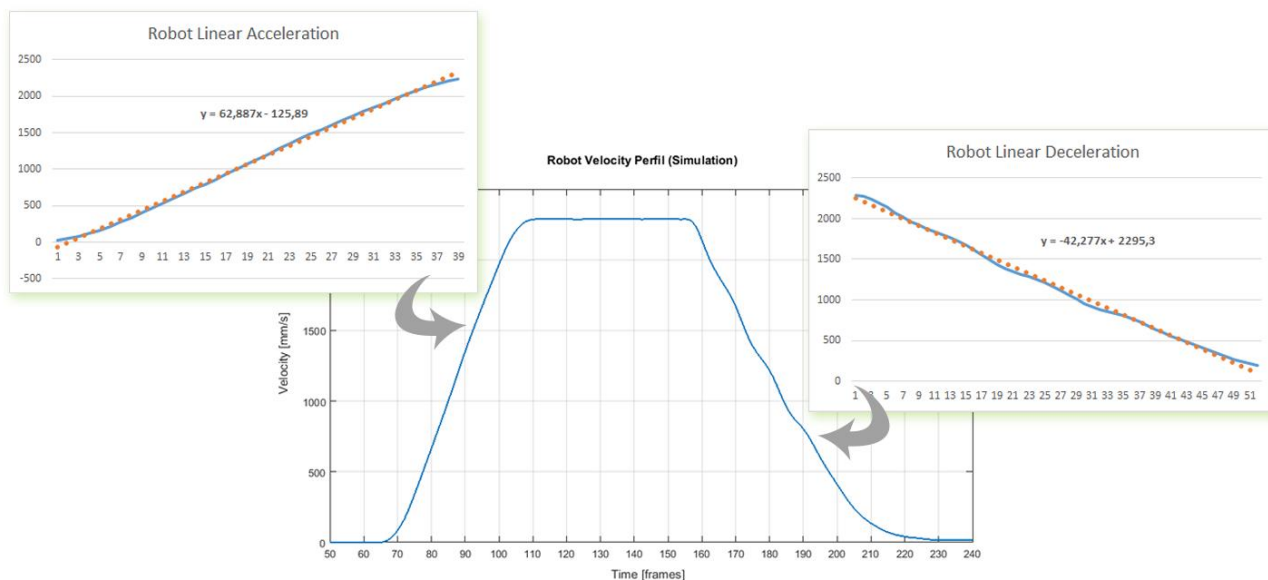


Figura 6. Perfil de Velocidade do robô e resultado das equações através do método dos mínimos quadrados.

$$v(t) = v_0 + 62,87\Delta t \left[ \frac{mm}{quadro} \right] - \text{Aceleração linear}$$

$$v(t) = v_0 - 42,227\Delta t \left[ \frac{mm}{quadro} \right] - \text{Desaceleração Linear}$$

De acordo com as equações, a aceleração do robô tem módulo aproximadamente igual a 62,87 mm/quadro, isto significa 4000 mm/s<sup>2</sup> ou 4 m/s<sup>2</sup> (um quadro tem período aproximado de 16ms). Já a desaceleração tem módulo aproximadamente igual a 42,3 mm/quadro, ou seja, 2640 mm/s<sup>2</sup> ou 2,64 m/s<sup>2</sup>.

Outra informação importante retirada dos gráficos é a velocidade máxima do robô que chega em torno de 2300 mm/s ou 2,3 m/s. A velocidade foi limitada em 80% da velocidade máxima para este experimento.

#### IV. 5. Algoritmo de Interceptação

Finalmente em posse das equações é possível colocar em prática o processo de iteração entre bola e robô para estabelecer o ponto ideal de encontro entre os dois objetos. Em um primeiro momento, não há preocupação em como o robô irá resgatar a bola, mas sim para qual ponto ele deverá se movimentar para que consiga interceptar a bola antes que ela saia do campo.



## Centro Universitário da FEI Projeto de pesquisa



O algoritmo compara os tempos de deslocamento entre o robô e a bola ao se deslocar de uma posição inicial até uma posição estimada (através do filtro de Kalman). Quando o robô conseguir se deslocar em um tempo menor que a bola, encontramos o ponto ideal de interceptação.

Primeiro é necessário verificar se a distância entre o robô e o ponto estimado da bola é maior que a distância necessária para o robô atingir velocidade máxima e em seguida parar. Sendo assim, sabe-se que neste caso, tem-se:

$$\text{Distância em vel. const} = \text{Distância Total} - (\text{Distância}_{\text{Máx. vel.}} + \text{Distância}_{\text{vel. Nula}})$$

Com isso, é possível calcular o tempo total de deslocamento do robô, como:

$$\text{Tempo}_{\text{Total}} = \text{Tempo}_{\text{Aceleração}} + \text{Tempo}_{\text{vel. Const}} + \text{Tempo}_{\text{Desaceleração}}$$

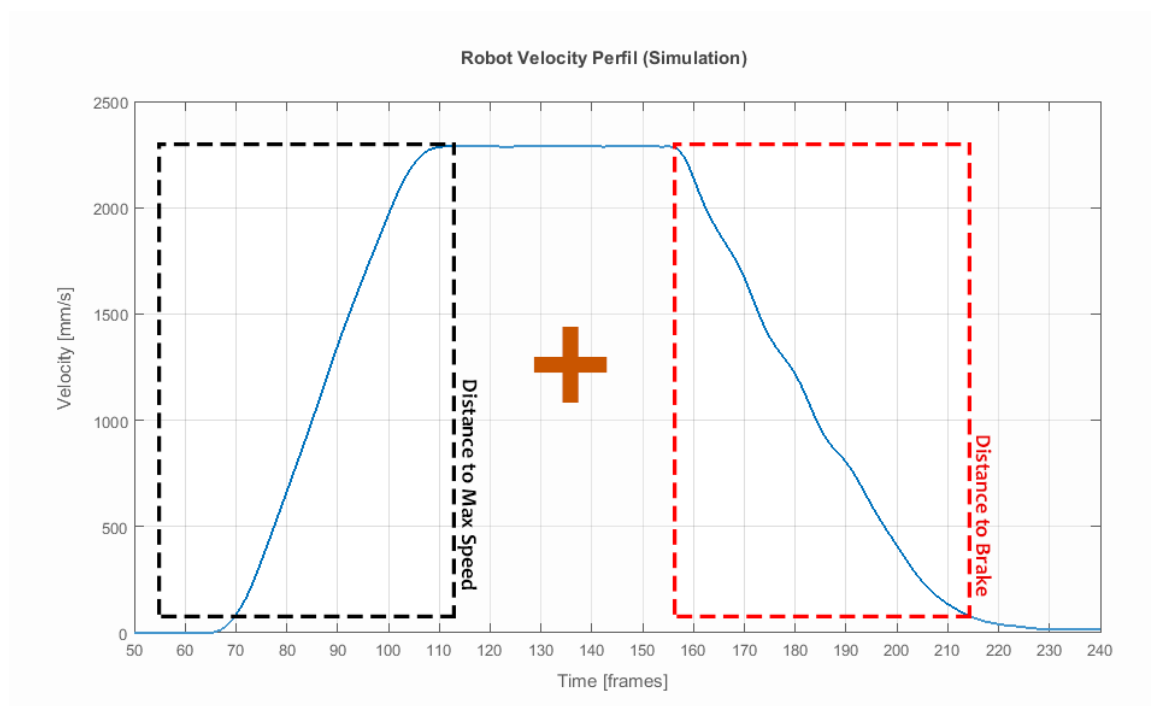


Figura 7. Distância percorrida até atingir velocidade máxima somada com a distância percorrida até parar pode ser utilizada para encontrar a distância a ser percorrida em velocidade constante.

Outra situação é quando a distância até o ponto de destino é menor que a soma das distâncias necessárias para atingir velocidade máxima e parar. Neste caso, não se sabe com qual velocidade o robô deverá atingir até começar a frear, sendo assim, através de iteração é calculada a distância de aceleração e desaceleração em cada quadro e somadas até que o valor seja inferior ao da distância de destino. O esquema pode ser visto a seguir:

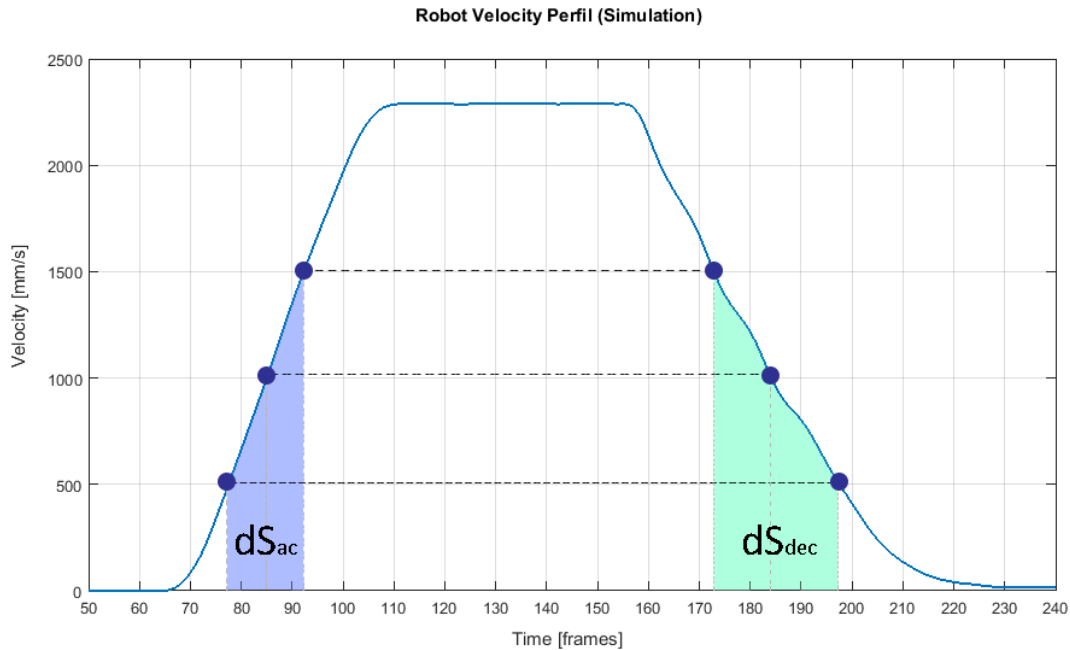


Figura 8. Iteração simultânea entre os momentos de aceleração e desaceleração para a composição do deslocamento total do robô.

O algoritmo foi implementado em linguagem C++ utilizando como ambiente de desenvolvimento o software Visual Studio 2012 da Microsoft. O pseudocódigo pode ser visto a seguir:

**Process Intercept**

1. *GetEnvironment()*
2. *StartKalmanFilter()*
3. *initialRbtVel* ← *GetRbtVel()*
4. *initialRbtPos* ← *GetRbtPos()*
5. **for** *step* ← 1 to *LookAheadTime*
6.     *predictedBallPos* ← *PredictBallPosByKalman(step)*
7.     *target* ← *predictedBallPos*
8.     *timeRbt2PredictedBall* ← *TimeRobot2Target(initialRbtPos, initialRbtVel, target, 0)*
9.     **if** *timeRbt2PredictedBall* < *step* \* (1/*FPS*)
10.         *robotDestination* ← *target*
11.     **break**

**time TimeRobot2Target**(*rbtPosition, rbtVelocity, goalPosition, goalVelocity*)

1.     *rbtAcceleration* ← 4000
2.     *rbtDeceleration* ← 2640
3.     *rbtMaxVel* ← 2300
4.     *minDistance* ← (*goalVelocity*<sup>2</sup> - *rbtMaxVel*<sup>2</sup>)/-2 \* *rbtDeceleration*
5.     *maxDistance* ← (*rbtMaxVel*<sup>2</sup> - *goalVelocity*<sup>2</sup>)/-2 \* *rbtAcceleration*



## Centro Universitário da FEI

### Projeto de pesquisa



```
6. distance2Goal ← (rbtPosition – goalPosition)
7. if distance2Goal > minDistance + maxDistance then
8.     distance2Goal ← distance2Goal – (minDistance + maxDistance)
9.     constVelTime ← (distance2Goal/robotMaxVel)
10.    decVelTime ← (goalVelocity – rbtMaxVel) / –rbtDeceleration
11.    accVelTime ← (rbtMaxVel – goalVelocity)/rbtAcceleration
12.    time ← accVelTime + constVelTime + decVelTime
13. else
14.    for step ← 1 to LookAheadTime
15.        if accDist + decDist < distance2Goal then
16.            predictedVel ← rbtVelocity + rbtAcceleration * step * (1/FPS)
17.            accDist ← (predictedVel2 – rbtVelocity2)/2 * rbtAcceleration * step
18.            decDist ← (predictedVel2 – rbtVelocity2)/2 * rbtAcceleration * step
19.        else
20.            decVelTime ← (goalVelocity – predictedVel) / –rbtDeceleration
21.            accVelTime ← (predictedVel – rbtVelocity)/rbtAcceleration
22.            time ← accVelTime + decVelTime
23.        break
24.    return time
```

---

## V. RESULTADOS E CONCLUSÕES FINAIS

O algoritmo de interceptação foi testado em ambiente de simulação utilizando os parâmetros de aceleração e desaceleração do robô conforme foram apresentados nas seções anteriores. A Figura 9 mostra a evolução do algoritmo na tentativa de o robô interceptar a bola após a intervenção de um chute.

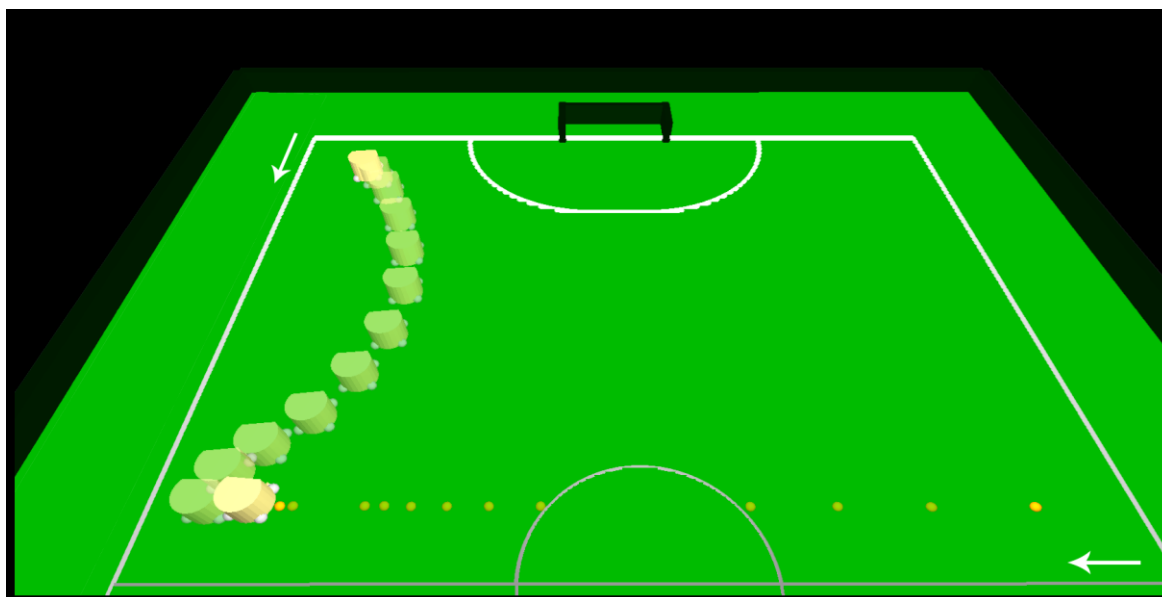


Figura 9. Interceptação de bola utilizando o filtro de Kalman e modelagem dos parâmetros cinéticos do robô.

A bola recebeu aceleração em uma única dimensão (eixo das ordenadas) e, portanto, só teve deslocamento em um único eixo como pode ser observado no Gráfico 10. Já o robô teve

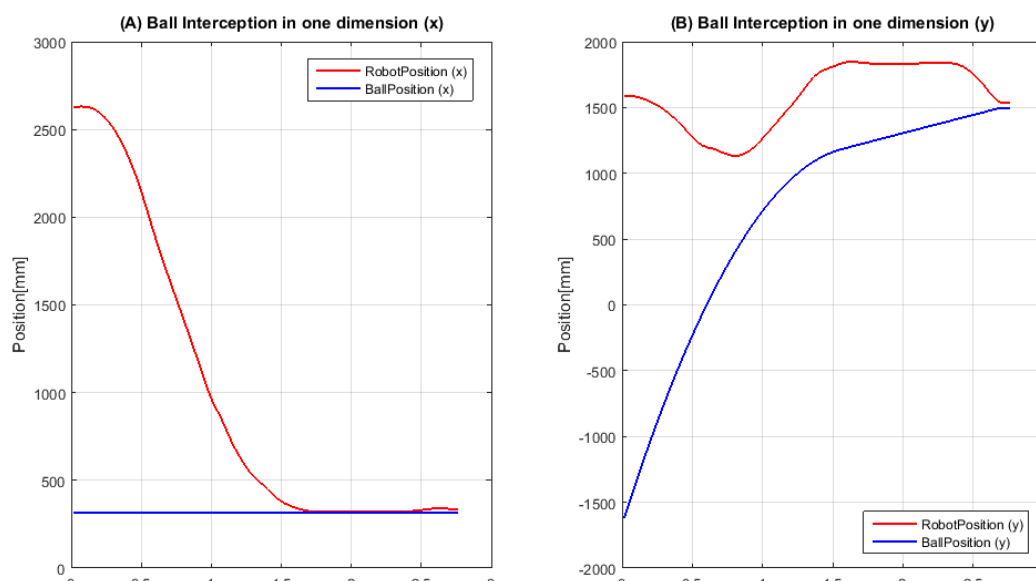


Gráfico 10. Posição e tempo da bola e robô de acordo com cada dimensão.





aceleração em ambas as dimensões tendo como destino o ponto de encontro calculado pelo algoritmo.

Através do Gráfico 11 é possível verificar que o ponto de encontro se deu em aproximadamente 2.75 segundos.

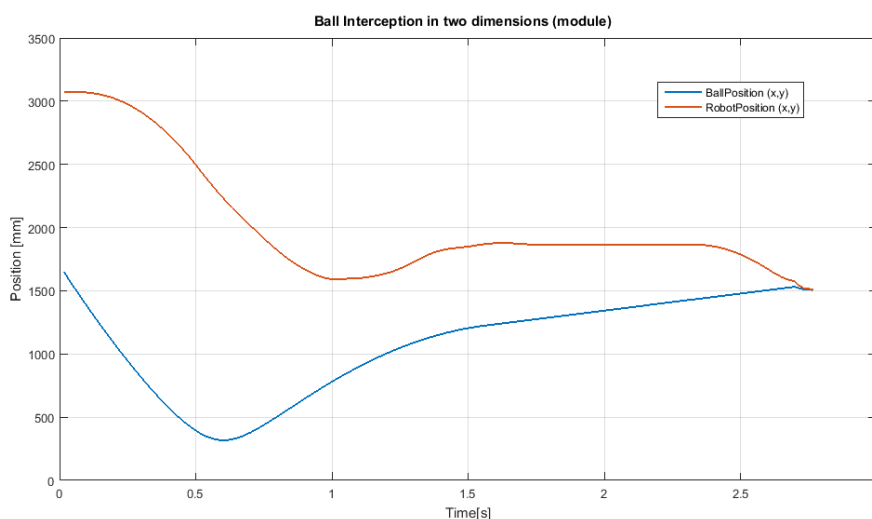


Gráfico 11. Módulo do deslocamento do robô e da bola até atingir o ponto de encontro otimizado entre eles.

O filtro de Kalman demonstrou robustez na estimativa dos estados futuros da bola, o que foi essencial no algoritmo de interceptação e apresentou desempenho excelente quanto a eliminação de falsos positivos, fornecendo informações mais confiáveis sobre o contexto da bola. Sua implementação é simples, embora o modelo utilizado na matriz de transformação do filtro deva ser aproximado ao modelo real do objeto, caso contrário o filtro terá sua fase de predição comprometida.

Em relação ao algoritmo de interceptação, os testes indicaram boa performance com um número máximo de 60 a 65 iterações, para grandes distâncias entre robô e bola (aproximadamente 6000mm), até encontrar o ponto. A simplicidade dos cálculos utilizando as leis de Newton favorece o desempenho no processamento, apesar de a dinâmica do robô não poder ser definida através de uma única equação, o que leva o algoritmo a ficar mais complexo. Não considerar a dinâmica e as fases do movimento do robô causaria maiores erros no cálculo do ponto de encontro e, muitas vezes não capacitaria aos robôs capturarem bolas com velocidades mais elevadas.



## VI. PLANO DE TRABALHO FUTURO

Como plano de um trabalho de continuação no futuro, uma ferramenta que adquira as características cinéticas do robô de forma automática facilitaria a utilização do algoritmo após mudanças mecânicas nos robôs ou no ambiente de jogo.

Considerar obstáculos na trajetória do robô e incorporar os caminhos traçados pelo algoritmo de *Path Planning* do robô e constituir um novo cálculo de tempo no deslocamento até o ponto de destino também é algo que precisaria ser explorado em trabalhos futuros.

## VII. BIBLIOGRAFIA

- [1] J. A. Gurzoni Jr., M. F. Martins, F. Tonidandel e R. A. C. Bianchi, "On the construction of a RoboCup small size league team," *Journal of the Brazilian Computer Society*, vol. 17, pp. 69-82, 2011.
- [2] *SmallSize League WebSite* - <http://robocupssl.cpe.ku.ac.th/>.
- [3] E. RoboFEI, *RoboFEI WebSite* [http://portal.fei.edu.br/pt-BR/pesquisas\\_projetos/projetos\\_institucionais/Robo\\_FEI/](http://portal.fei.edu.br/pt-BR/pesquisas_projetos/projetos_institucionais/Robo_FEI/).
- [4] M. Veloso, M. Bowling e S. Achim, "The CMUnited-99 Small-Size Robot Team," em *RoboCup-99: Robot Soccer World Cup III*, vol. 1856, M. Veloso, E. Pagello e H. Kitano, Eds., Springer Berlin Heidelberg, 2000, pp. 661-662.
- [5] F. Stolzenburg, O. Obst e J. Murray, "Qualitative Velocity and Ball Interception," em *KI 2002: Advances in Artificial Intelligence*, vol. 2479, M. Jarke, G. Lakemeyer e J. Koehler, Eds., Springer Berlin Heidelberg, 2002, pp. 283-298.
- [6] T. Laue, A. Burchardt, K. Cierpka, S. Fritsch, N. Göde, K. Huhn, T. Kirilov, B. Lassen, M. Miezal, E. Lyatif, M. Schwarting, A. Seekircher e R. Stein, "B-Smart (Bremen Small Multi-Agent Robot Team) Team Description for RoboCup 2007," em *In RoboCup 2007: Robot Soccer World Cup XI Preproceedings*.
- [7] J. Simonoff, *Smoothing Methods in Statistics*, Springer, 1996.
- [8] The MathWorks, Inc., "MathWorks," 2014. [Online]. Available: [http://www.mathworks.com/help/curvefit/smoothing-data.html#bq\\_6ys3-1](http://www.mathworks.com/help/curvefit/smoothing-data.html#bq_6ys3-1).
- [9] MATLAB, version 7.10.0 (R2010a), Natick, Massachusetts: The MathWorks Inc., 2010.
- [10] K. G. Gauss, *Theory of Motion of the Heavenly Bodies*, New York, Dover, 1963.
- [11] A. Kelly e C.-M. U. P. P. R. INST., *A 3D State Space Formulation of a Navigation Kalman Filter for Autonomous Vehicles*, Defense Technical Information Center, 1994.
- [12] P. S. Maybeck, "Stochastic models, estimation, and control," em *Mathematics in Science and Engineering*, vol. 141, 1979.
- [13] M. B. M. V. Brett Browning, "Improbability Filtering for Rejecting False Positives".
- [14] G. Guennebaud, B. Jacob e others, *Eigen v3*, 2010.
- [15] E. W. Weisstein, "'Least Squares Fitting." From MathWorld--A Wolfram Web Resource," [Online]. Available: <http://mathworld.wolfram.com/LeastSquaresFitting.html>.
- [16] D. J. Hand, "Empirical Model Building: Data, Models, and Reality, Second Edition by James R. Thompson," *International Statistical Review*, vol. 80, n° 1, pp. 192-192, 2012.
- [17] M. V. Studio, version 11.0 (Update1), King County, Washington: Microsoft Corporation, 2012.
- [18] A. Oppenheim, "RES.6-007 Signals and Systems, Spring 2011.," (Massachusetts Institute of Technology: MIT OpenCourseWare), [Online]. Available: <http://ocw.mit.edu>. [Acesso em 10 Nov 2014].



Centro Universitário da FEI  
Projeto de pesquisa

