Modelo de controlador preditivo para rastreamento de trajetória de robôs omnidirecionais com tratamento de restrições

Francisco Arthur Bonfim Azevedo [*]

[*] *Instituto Tecnológico de Aeronáutica,*
Praça Marechal Eduardo Gomes, 50, Vila das Acácias, 12228-900,
São José dos Campos, SP, Brasil,
(e-mail: arthurazevedo41@gmail.com).

Abstract: This work study and develops a MPC to a omnidirecional robot, turned to the trajectory tracking with actuation constraints. The dynamic of the sistem is nonlinear, but linearization around a reference results in a linear time-varying system. Finally, simulations are show to validate and analyze the proposed method.

Resumo: O presente trabalho estuda e desenvolve um controlador do tipo preditivo para um robô omnidirecional, voltado para o rastreio de trajetórias com respeito a restrições. A dinâmica do sistema é não-linear, mas a linearização em torno de uma referência resulta em um sistema linear variante no tempo. Por fim, são mostradas simulações para validar e analisar o método proposto.

Keywords: Model Preditive Control; Robotic; Robot Control; Omnidirecional; Mobile Robot; Constraints.

Palavras-chaves: Controle Preditivo; Robótica; Controle de Robôs;Omnidirecional; Robótica móvel; Restrições.

1. INTRODUÇÃO

Robô omnidirecional é uma classe da robótica móvel com robôs que podem se mover facilmente para qualquer direção no plano. Para ser capaz de realizar esse movimento, pode-se utilizar de diversas técnicas, variando geometria, direção, disposição e velocidade das rodas. Um exemplo de robô omnidirecional, o qual é tratado no presente trabalho, é o Small Size.

A *Small Size League* (SSL) é uma competição de futebol de robôs omnidirecionais com partidas de dois times de 6 robôs no caso da divisão B ou 8 robôs no caso da divisão A(RoboCup, 2019). As regras estabelecem que os robôs caibam em um cilindro de 180 mm de diâmetro e 150 mm de altura. Em geral, os robôs possuem 4 rodas, com vários pequenos roletes nelas, com o número variando de equipe para equipe, que são responsáveis pelo movimento omnidirecional do robô. O sistema de visão é composto por câmeras posicionadas no teto do campo e algoritmos de visão capazes de identificar cada robô graças a um padrão de marcações coloridas posicionado na parte superior de cada um, medindo sua posição e sua orientação. A maior parte do processamento, como algoritmos de estratégia e de cálculo de trajetórias, ocorrem em um computador central que se comunica com os robôs por rádio.

Uma habilidade importante de um robô Small Size é a de se movimentar rapidamente pelo campo enquanto desvia de outros jogadores, tanto para alcançar a bola, quando para armar jogadas e preparar para passes. O presente trabalho documenta um controlador para rastreamento de trajetórias para a equipe Small Size da ITAndroids. ITAndroids é um grupo do Instituto Tecnológico de Aeronáutica (ITA) que participa de competições de robótica. O grupo também desenvolve outros robôs, em categorias simuladas como é o caso do *Soccer 2D*, *Soccer 3D* e categorias físicas como é o caso da *Humanoid Kid-Size* e *VSS*, além de competir em competições nacionais como a *LARC (Latin American Robotics Competition)* e internacionais como a *RoboCup*. O robô Small Size da equipe e sua roda pode ser visto na Figura 1.

(a) Vista geral do robô. (b) Detalhe da roda com roletes.

Figura 1. Robô Small Size da equipe ITAndroids.

É comum na robótica móvel dividir o movimento do robô em duas camada: planejamento e controle (Maximo, 2012). Primeiramente, o robô planeja qual seria a melhor trajetória que ele deveria seguir para atingir seu objetivo, desviando de obstáculos, e, em seguida, o controle age obtendo os comandos que o robô deve obedecer para seguir fisicamente possível seguir a dada trajetória. O problema do planejamento é geralmente resolvido através do emprego de algoritmos como A* ou *Rapid Exploring Random Tree* (RRT)(Siegwart et al., 2011). Já o controle é resolvido através de técnicas da teoria de controle (Maximo, 2012; Camacho and Bordons, 2007; Klančar and Škrjanc, 2007). No presente trabalho, a atenção será voltada para o controle do robô sobre uma trajetória pré-definida, por tanto, será assumido que a trajetória já foi planejada.

Uma dessas técnicas de controle é o Controle Preditivo, ou MPC (*Model Preditive Control*), a qual será aqui tratada e aplicada. Controle preditivo é uma técnica de controle ótimo que se baseia em levar em conta o comportamento futuro do sistema para corrigir o sistema no presente(Maximo, 2012). Ele age em tempo discreto e requer resolver problemas de otimização em cada *time step* considerando um número fixo de passos futuros, conhecidos como horizonte de controle. Uma das grandes vantagens do MPC é o fato de ele lidar muito bem com restrições, as quais também são incluídas em um problema de otimização que deve ser resolvido em cada *time step*, obtendo assim não apenas um comando que obedece as restrições, mas um valor ótimo de comando para o dado caso. A solução desses problemas de otimização com restrições requerem soluções numéricas através de algoritmos iterativos, o que faz com que o MPC seja naturalmente um controlador pesado computacionalmente.

No caso do robô Small Size, existem restrições quanto a velocidade de rotação das rodas e tensão máxima que o motor é capaz de suportar. Utilizando controladores como o PID esses problemas de restrição são resolvidos através de saturação do comando no robô, o que não é tão efetivo quanto a abordagem do MPC, que consegue calcular o controle ótimo até nesses casos, e, portanto, por esse ponto de vista o controle preditivo se sobressai em relação ao PID. Por isso, decidiu-se utilizar o MPC para um controlador para seguir trajetória no Small Size da ITAndroids.

2. MODELO DO ROBÔ

O modelo do robô é feito com base no ambiente da competição da SSL e nos robôs Small Size utilizados pela ITAndroids (Maximo, 2018). Seja $q = [x, y, \phi]^T$ o vetor de estados do robô sendo definido em relação a um sistema global de coordenadas, que no caso da SSL fica no centro do campo. Os pontos x e y são tomados em relação ao centro do robô e o ângulo ϕ toma-se como referência a frente do robô. Além disso, considera-se que cada roda tem sua velocidade controlada, o que no nosso caso é feito através de um controlador PID com realimentação feita pela leitura do encoder.

2.1 Modelo cinemático

Agora, toma-se um sistema de coordenadas x_r, y_r solidário ao robô com origem em seu centro e com o eixo x_r alinhado com a frente do robô. Denota-se por v, v_n e w as velocidades frontal, lateral e angular, respectivamente. Os sistemas de referência podem ser vistos na Figura 2.

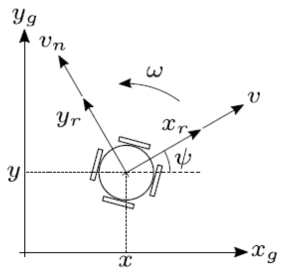
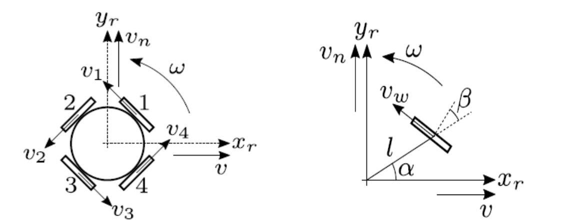


Figura 2. Convenções relativas a sistemas de coordenadas cartesianos.

A numeração de cada roda é feita como na Figura 3a seguindo o sentido anti-horário, a partir do eixo x_r. A convenção de velocidade angular da roda é feita de modo a manter a consistência com a sua velocidade linear, de forma que, para a i-ésima roda vale $v_i = \omega_i r_i$, sendo v_i, ω_i e r_i respectivamente a velocidade linear, a velocidade angular e o raio da i-ésima roda, respectivamente.

Quanto a roda, considera-se uma roda omnidirecional genérica a uma distância l do ponto de referência e posicionada conforme os ângulos α e β, conforme a Figura 3b. Por fim, a condição de não deslizamento para cada roda está expressa na Equação (1).



(a) Convenção considerada em cada roda.

(b) Detalhe de cada roda no referencial do robô.

Figura 3. Modelagem das rodas do robô.

$$v_w = \omega_w r = -sin(\alpha + \beta)v + cos(\alpha + \beta)v_n + lcos(\beta)\omega \quad (1)$$

Sendo: v_w e ω_w a velocidade linear e angular da roda, respectivamente. Para expressar simultaneamente as 4 rodas, coloca-se na forma matricial conforme a Equação 2. A matriz M mapeia a velocidade do robô nas velocidades das rodas.

$$\underbrace{\begin{bmatrix} v_1 \\ v_2 \\ v_3 \\ v_4 \end{bmatrix}}_{v_w} = \underbrace{\begin{bmatrix} -sin(\alpha_1 + \beta_1) & cos(\alpha_1 + \beta_1) & l_1 cos(\beta_1) \\ -sin(\alpha_2 + \beta_2) & cos(\alpha_2 + \beta_2) & l_1 cos(\beta_2) \\ -sin(\alpha_3 + \beta_3) & cos(\alpha_3 + \beta_3) & l_1 cos(\beta_3) \\ -sin(\alpha_4 + \beta_4) & cos(\alpha_4 + \beta_4) & l_1 cos(\beta_4) \end{bmatrix}}_{M} \underbrace{\begin{bmatrix} v \\ v_n \\ \omega \end{bmatrix}}_{v_r} \quad (2)$$

Como a matriz M não é quadrada, para isolar o vetor v_r, utiliza-se a pseudoinversa M^+ da matriz M, conforme representado na Equação (3).

$$v_r = M^+ v_w = (M^T M)^{-1} M^T v_w \quad (3)$$

2.2 Modelo dinâmico

Para o modelo dinâmico do robô será utilizada a formulação de mecânica Lagrangeana. As coordenadas generalizadas escolhidas são os ângulos de rotação das rodas $q = [\theta_1 \theta_2 \theta_3 \theta_4]^T$. Da formulação da mecânica Lagrangeana, tem-se a Equação (4).

$$\frac{d}{dt}\left(\frac{\partial L}{\partial \dot{q}}\right) - \frac{\partial L}{\partial q} = \tau \tag{4}$$

Sendo L a lagrangeana do sistema e τ o vetor de forças generalizadas, que no caso trata-se do torque aplicado pelas rodas. A lagrangeana é dada pela Equação (5).

$$L = T - V \tag{5}$$

Em que T e V são, respectivamente, as energias cinética e potencial do sistema. No caso tratado, a energia potencial é devida a gravidade, mas como o robô se move no plano, ter-se-á V = 0. Já a energia cinética será dada pela Equação (6).

$$T = \frac{1}{2} m v_{cm}^T v_{cm} + \frac{1}{2} I_{cm} \omega^2 \tag{6}$$

Sendo m a massa do robô, I_{cm} o momento de inércia em relação ao centro de massa (CM) e v_{cm} o vetor velocidade linear do centro de massa. Preferiu-se não adicionar as energias cinéticas de rotação do motor e da roda pois pode-se adicionar posteriormente os efeitos dos torques para acelerar o motor e a roda(Maximo, 2018). Porém, incluir ou não essas energias altera a interpretação de τ. Se essas energias não são incluídas, τ representa torques aplicados pela roda no chão, depois de já ter sido descontados os torques para acelerar o motor e a roda.

Escrevendo v_{cm} projetado no sistema de referência do robô x_r, y_r, sendo um deslocamento de $r_{cm} = [x_{cm}\ y_{cm}\ z_{cm}]^T$ do CM em relação ao ponto de referência, tem-se a Equação (7).

$$v_{cm} = \begin{bmatrix} v \\ v_n \\ 0 \end{bmatrix} + \omega \hat{k} \times \begin{bmatrix} x_{cm} \\ y_{cm} \\ z_{cm} \end{bmatrix} \Rightarrow v_{cm} = \begin{bmatrix} v - \omega y_{cm} \\ v_n + \omega x_{cm} \\ 0 \end{bmatrix} \tag{7}$$

Por fim, falta colocar colocar v,v_n e ω nas coordenadas generalizadas q. Para isso, basta se utilizar da Equação (3).

Substituindo as Equações (3) e (7) em (6) obtêm-se a expressão da lagrangeana, que por fim deve ser substituída na Equação (4), obtendo-se as equações dinâmicas do robô. A solução será da forma da Equação (8).

$$H_r \underbrace{\begin{bmatrix} \dot{\omega}_1 \\ \dot{\omega}_2 \\ \dot{\omega}_3 \\ \dot{\omega}_4 \end{bmatrix}}_{\dot{v_w}} = \underbrace{\begin{bmatrix} \tau_1 \\ \tau_2 \\ \tau_3 \\ \tau_4 \end{bmatrix}}_{\tau} \tag{8}$$

Com H_r sendo a matriz que descreve a dinâmica do robô.

Agora, desconta-se os torques necessários para acelerar o motor e a roda. O torque gerado no i-ésimo motor é usado para acelerar o próprio motor, a respectiva roda e fornecer τ, logo, utilizando-se as equações de modelamento de motor elétrico, tem-se as Equações (9), (10) e (11).

$$\tau_g = K j_i - J_m n \dot{\omega}_i - B_m N \dot{\omega}_i \tag{9}$$

$$\tau_e = N \eta \tau_g \tag{10}$$

$$\tau_e = J_w \dot{\omega}_i + B_m \dot{\omega}_i + \tau_i \tag{11}$$

Sendo τ_g e τ_e os torques antes e depois da redução, respectivamente; J_m e J_w são as inércias do motor e da roda, respectivamente; B_m e B_w são os coeficientes de atrito viscoso do motor e da roda, respectivamente; K é a constante de torque do motor; j_i é a corrente no i-ésimo motor; N é o fator de redução e η é a eficiência do sistema de redução. Isolando τ_i em função dos demais termos, tem-se a Equação (12).

$$\tau_i = N \eta K j_i - \underbrace{(N^2 \eta J_m)}_{J_{eq}} \omega_i - \underbrace{(N^2 \eta B_m + B_w)}_{B_{eq}} \omega_i \tag{12}$$

Substituindo a Equação (12) na Equação (8), obtêm-se a Equação (13), sendo I_4 a matriz identidade de ordem 4.

$$\underbrace{(H_r + J_e q I_4)}_{H_j} \underbrace{\begin{bmatrix} \dot{\omega}_1 \\ \dot{\omega}_2 \\ \dot{\omega}_3 \\ \dot{\omega}_4 \end{bmatrix}}_{\dot{v}_w} + \underbrace{B_{eq} I_4}_{C_j} \underbrace{\begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \\ \omega_4 \end{bmatrix}}_{v_w} = \underbrace{N \eta K}_{k_j} \underbrace{\begin{bmatrix} j_1 \\ j_2 \\ j_3 \\ j_4 \end{bmatrix}}_{j} \tag{13}$$

A Equação (13) trata a dinâmica em função da corrente j enviada para o motor, e é muito útil para o caso da implementação de uma malha de corrente. Pode-se ainda representar essa equação em função da tensão no motor, bastando para isso utilizar a Equação (14).

$$j_i = \frac{V_i - V_{backemf}}{R} = \frac{V_i - K \omega_i}{R} \tag{14}$$

Sendo V_i a tensão no i-ésimo motor, $V_{backemf}$ a força contra-eletromotriz induzida e R a resistência do motor. Substituindo a Equação (14) na Equação (13) obtêm-se a Equação (15).

$$\underbrace{(H_r + J_e q I_4)}_{H_v} \underbrace{\begin{bmatrix} \dot{\omega}_1 \\ \dot{\omega}_2 \\ \dot{\omega}_3 \\ \dot{\omega}_4 \end{bmatrix}}_{\dot{v}_w} + \underbrace{\left(B_{eq} + \frac{N \eta K^2}{R}\right) I_4}_{C_v} \underbrace{\begin{bmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \\ \omega_4 \end{bmatrix}}_{v_w} = \underbrace{\frac{N \eta K}{R}}_{k_v} \underbrace{\begin{bmatrix} V_1 \\ V_2 \\ V_3 \\ V_4 \end{bmatrix}}_{V} \tag{15}$$

Por fim, o sistema pode ainda ser representado na forma de espaço de estados, conforme Equação (16).

$$\dot{v_w} = \underbrace{-H_v^{-1} C_v}_{A} v_w + \underbrace{k H_v^{-1}}_{B} \underbrace{V}_{u} = A v_w + B u \tag{16}$$

2.3 Modelo de rastreamento de trajetória

Agora falta resolver o problema de rastreamento de trajetória. Primeiramente, define-se as variáveis de controle que serão usadas no nosso modelo, a saber $u = [v, v_n, \omega]^T$. Além disso, a dinâmica de movimento do robô não linear segue a Equação (17) retirada diretamente da análise da Figura 2.

$$\dot{q}(t) = \begin{bmatrix} \dot{x}(t) \\ \dot{y}(t) \\ \dot{\theta}(t) \end{bmatrix} = \begin{bmatrix} cos\theta & -sin\theta & 0 \\ sin\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ v_n \\ \omega \end{bmatrix} \tag{17}$$

Sendo $q = [x, y, \theta]^T$ o vetor de estados do robô. O sistema de visão diretamente mede o estado do robô q.

Os motores das rodas são motores DC, os quais possuem um limite de velocidade devido ao limite de voltagem da bateria. Além disso, o campo possui dimensões fixas, e portanto o caminho pelo qual o robô pode caminhar durante o jogo é limitado, no entanto essa restrição só realmente precisa ser aplicada durante uma partida. Dessa forma, as Inequações (18), (19) e (20).

$$-\omega_{MAX} \leq \omega_i \leq \omega_{MAX} \tag{18}$$

$$-X_{MAX} \leq x_i \leq X_{MAX} \tag{19}$$

$$-Y_{MAX} \leq y_i \leq Y_{MAX} \tag{20}$$

Apesar de certas diferenças de notação e, principalmente, as modificações nas equações para o robô omnidirecional, o equacionamento a seguir segue o mostrado em (Klančar and Škrjanc, 2007) e (Maximo, 2012). Considere um robô virtual que segue perfeitamente a trajetória de referência. O robô real terá um erro estacionário $e(t) - [e_1(t), e_2(t), e_3(t)]^T$ como mostrado na Figura 4. Isso pode ser escrito no sistema de coordenadas reais do robô como na Equação (21).

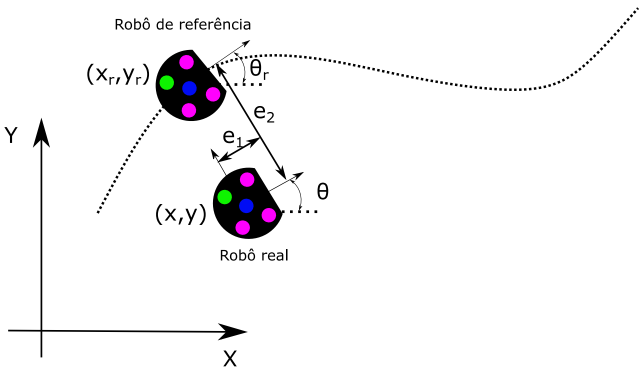


Figura 4. Erro entre o robô seguindo perfeitamente a trajetória e o robô real.

$$e = \begin{bmatrix} cos\theta & sin\theta & 0 \\ -sin\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} (q_r - q) \tag{21}$$

Derivando a equação (21) e substituindo nela a Equação (17), chega-se a Equação (22).

$$\dot{e} = \begin{bmatrix} v \\ v_n \\ \omega \end{bmatrix} + \begin{bmatrix} -1 & 0 & e_2 \\ 0 & -1 & e_1 \\ 0 & 0 & -1 \end{bmatrix} u \tag{22}$$

Pode-se pensar na variável de controle u como composta por componentes, sendo um *feedforward* u_Fe um *feedback* u_B. Levando em conta o erro de orientação, tem-se a Equação (23).

$$u_F = \begin{bmatrix} vcos(e_3) \\ v_n cos(e_3) \\ \omega \end{bmatrix} \tag{23}$$

Substituindo $u = u_F + u_B$ na Equação (22), obtêm-se a Equação (24).

$$\dot{e} = \begin{bmatrix} v(1 - cos(e_3)) + \omega e_2 \\ v_n(1 - cos(e_3)) + \omega e_1 \\ \omega \end{bmatrix} + \begin{bmatrix} -1 & 0 & e_2 \\ 0 & -1 & e_1 \\ 0 & 0 & -1 \end{bmatrix} u \tag{24}$$

Tal sistema de equações é não linear, mas nota-se que o ponto de referencia da trajetória do sistema, ou seja, $e = 0_{3\times1}$ e $u_B = 0_{3\times1}$, é um ponto de equilíbrio, já que $\dot{e}(e = 0_{3\times1}, u_B = 0_{3\times1}) = 0_{3\times1}$. Com isso, pode-se linearizar o sistema em torno da referência da trajetória, o que resulta na Equação (25).

$$\dot{\bar{e}} = \underbrace{\begin{bmatrix} 0 & \omega & 0 \\ -\omega & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}}_{A} \bar{e} + \underbrace{\begin{bmatrix} -1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & -1 \end{bmatrix}}_{B} \bar{u}_B \tag{25}$$

2.4 Modelo em tempo contínuo no espaço de estados

Sendo $\bar{e}$ e $\bar{u}_B$ respectivamente o erro linearizado e a variável de controle linearizada. O sistema pode ser então expresso na forma de espaço de estados, conforme Equação (26). y é a saída do sistema e C é a matriz que transforma a variável $\bar{e}$ na saída do sistema, e trata-se da identidade de ordem 3.

$$\begin{aligned} \dot{\bar{e}} &= A\bar{e} + B\bar{u}_B \\ y &= C\bar{e} \end{aligned} \tag{26}$$

Vale ainda ressaltar que a matriz A varia no tempo, por ser função da velocidade angular do robô, e por isso deve ser recalculada em cada *timestep*. A notação de barra sobre as variáveis linearizadas será deixada implícita nas equações a seguir por questões de simplicidade, no entanto, a seguir sempre será usado o sistema já linearizado.

3. CONTROLE PREDITIVO (MPC)

A técnica de controle preditivo resolve um problema de otimização considerando o comportamento futuro do sistema e culmina em uma sequencia ótima de ações de controle (Camacho and Bordons, 2007). O sistema de controle do presente trabalho possui as seguintes características: modelo que varia linearmente no tempo, do tipo MIMO, implementação em tempo discreto, função de custo quadrática, uso de horizonte retrocedente e com tratamento de restrições.

3.1 Discretização no tempo

O sistema foi discretizado utilizando a Equação (27), conhecida como discretização exata ou *Zero-Order-Hold* (ZOH)(Camacho and Bordons, 2007). Dessa forma, obteve-se as matrizes A_d e B_d do espaço de estados para tempo discreto, conforme Equação (28), com um período de amostragem de $T = 0.005\ s$. A e B são as matrizes do espaço de estados em tempo contínuo na Equação (26).

$$\begin{cases} A_d = -e^{-AT} \\ B_d = \displaystyle\int_0^T -e^{-A\xi} B d\xi \end{cases} \tag{27}$$

$$\begin{array}{c} e(k+1) = A_d(k)e(k) + B_d(k)u_b(k) \\ y(k) = Ce(k) \end{array} \tag{28}$$

3.2 Função de custo

A função de custo J utilizada foi do tipo quadrada com horizonte de controle e de predição finitos (Camacho and Bordons, 2007; Klančar and Škrjanc, 2007; Maximo, 2012) como pode ser visto na Equação (29).

$$\begin{aligned} J(\hat{e}(k+i|k), \Delta\hat{u}_B(k+i-1|k)) = \\ \sum_{i=1}^{N} (\hat{e}(k+i|k) - e_r)^T Q(\hat{e}(k+i|k) - e_r) \\ + \sum_{i=1}^{M} \Delta\hat{u}_B(k+i-1|k)^T R \Delta\hat{u}_B(k+i-1|k) \end{aligned} \tag{29}$$

Sendo N o horizonte de predição e M o horizonte de controle, sendo $M \geq N$. $\Delta u_B(k)$ tata-se dos incrementos de controle aplicados e possui a seguinte forma: $\Delta u_B(k) = u_B(k) - u_B(k-1)$. Os acentos circunflexos por sobre as variáveis indica valores preditos, ou seja, são as ações futuras do sistema. Q e R são as matrizes peso do erro de estado e do esforço de controle, respectivamente. $Q \in R^3 \times R^3$ e $R \in R^3 \times R^3$ são matrizes diagonais, com $Q > 0$ e $R > 0$. Por fim, e_r é a referência para o controle, no caso, tratando-se do erro entre a trajetória atual e a trajetória planejada pelo robô. Utiliza-se sempre $e_r = 0_{3\times1}$.

3.3 Predição

Para o processo de predição, definem-se os vetores apresentados na Equação (30).

$$\hat{y} = \begin{bmatrix} \hat{y}(k+1|k) \\ \hat{y}(k+2|k) \\ \vdots \\ \hat{y}(k+N|k) \end{bmatrix}_{3\cdot N\times1} ; \quad \hat{u} = \begin{bmatrix} \hat{u}_B(k|k) \\ \hat{u}_B(k+1|k) \\ \vdots \\ \hat{u}_B(k+M-1|k) \end{bmatrix}_{3\cdot M\times1} ;$$
$$ref = \begin{bmatrix} e_r \\ e_r \\ \vdots \\ e_r \end{bmatrix}_{3\cdot N\times1} ; \quad \Delta\hat{u} = \begin{bmatrix} \Delta\hat{u}_B(k|k) \\ \Delta\hat{u}_B(k+1|k) \\ \vdots \\ \Delta\hat{u}_B(k+M-1|k) \end{bmatrix}_{3\cdot M\times1} \tag{30}$$

Com isso, a função custo pode agora ser representada como na Equação (31).

$$J(\hat{y}, \Delta\hat{u}) = (\hat{y} - ref)^T Q(\hat{y} - ref) + \Delta\hat{u}^T R \Delta\hat{u} \tag{31}$$

Tomando a solução da Equação (31), que será resolvida em 3.5, chega-se na Equação (32) (Camacho and Bordons, 2007) que trata-se da equação de predição em termos de $\hat{u}$.

$$\hat{y} = H\hat{u} + f_u \tag{32}$$

Com as matrizes H e f_u apresentadas na Equação (33).

$$H = \begin{bmatrix} CB_d & 0 & \cdots & 0 \\ CA_dB_d & CB_d & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ CA_d^{N-1}B & CA_d^{N-2}B & \cdots & CB_d \end{bmatrix}_{3\cdot N\times3\cdot N}$$
$$f_u = \Phi_u e(k), \ \Phi_u = \begin{bmatrix} CA_d \\ CA_d^2 \\ \vdots \\ CA^N \end{bmatrix}_{3\cdot N\times3} \tag{33}$$

No entanto, para ficar na forma mais clássica do MPC, deseja-se exprimir a equação de predição em termos do incremento de controle $\Delta\hat{u}$, e, para isso, utiliza-se a estratégia do estado aumentado, criando-se o estado aumentado $\xi(k)$ conforme a Equação (34). Assim, a equação em espaço de estados se torna a Equação (35), que fica em termos de $\Delta\hat{u}$. Ficam também definidas as matrizes do espaço de estados aumentado $\tilde{A}$,$\tilde{B}$ e $\tilde{C}$

$$\xi(k) = \begin{bmatrix} e(k) \\ u_B(k-1) \end{bmatrix} \tag{34}$$

$$\xi(k+1) = \underbrace{\begin{bmatrix} A_d & B_d \\ 0_{3\times3} & I_3 \end{bmatrix}}_{\tilde{A}} \xi(k) + \underbrace{\begin{bmatrix} B_3 \\ I_3 \end{bmatrix}}_{\tilde{B}} \Delta u_B(k)$$
$$y(k) = \underbrace{[C \ 0_{3\times3}]}_{\tilde{C}} \xi(k) \tag{35}$$

A equação de predição fica sendo então a Equação (36).

$$\hat{y} = G\Delta\hat{u} + f \tag{36}$$

Com as matrizes G e f apresentadas na Equação (37).

$$G = \begin{bmatrix} \tilde{C}\tilde{B} & 0 & \cdots & 0 \\ \tilde{C}\tilde{A}\tilde{B} & \tilde{C}\tilde{B} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ \tilde{C}\tilde{A}^{N-1}\tilde{B} & \tilde{C}\tilde{A}^{N-2}\tilde{B} & \cdots & C\tilde{B} \end{bmatrix}_{3\cdot N\times3\cdot N}$$
$$f = \Phi\xi(k), \ \Phi = \begin{bmatrix} \tilde{C}\tilde{A} \\ \tilde{C}\tilde{A}^2 \\ \vdots \\ \tilde{C}\tilde{A}^N \end{bmatrix}_{3\cdot N\times3} \tag{37}$$

3.4 Tratamento de restrições

Da Inequação (18), vem a Inequação (38).

$$\begin{bmatrix}-\omega_{max}\\-\omega_{max}\\-\omega_{max}\\-\omega_{max}\end{bmatrix} \leq \underbrace{\begin{bmatrix}\omega_1\\\omega_2\\\omega_3\\\omega_4\end{bmatrix}}_{\omega_w} \leq \begin{bmatrix}\omega_{max}\\\omega_{max}\\\omega_{max}\\\omega_{max}\end{bmatrix} \tag{38}$$

No entanto, da condição de não deslizamento, $\omega_w = \frac{v_w}{r}$. Combinando esse fato, a Inequação (38), a Equação (3) e o fato de que $u = [v, v_n, \omega]^T$, ter-se-á a Equação (39), que impõe as restrições na variável de controle.

$$\underbrace{\frac{1}{r}M^+\begin{bmatrix}-\omega_{max}\\-\omega_{max}\\-\omega_{max}\\-\omega_{max}\end{bmatrix}}_{u_{min}} \leq u_F + u_B \leq \underbrace{\frac{1}{r}M^+\begin{bmatrix}\omega_{max}\\\omega_{max}\\\omega_{max}\\\omega_{max}\end{bmatrix}}_{u_{max}}$$
$$\Rightarrow u_{B_{min}} = u_{min} - u_F \leq u_B \leq u_{B_{max}} = u_{max} - u_F \tag{39}$$

Para deixar em função do vetor de incrementos de controle, basta deixar na forma da Equação (40).

$$\begin{bmatrix}T_M^{I_3}\\-T_M^{I_3}\end{bmatrix}\Delta\hat{u} \leq \begin{bmatrix}[u_{B_{max}} - u(k-1)]_M\\ [u(k-1) - u_{B_{min}}]_M\end{bmatrix},$$
$$T_M^{I_3} = \begin{bmatrix}I_3 & 0_{3\times3} & \cdots & 0_{3\times3}\\ I_3 & I_3 & \cdots & 0_{3\times3}\\ \vdots & \vdots & \ddots & \vdots\\ I_3 & I_3 & \cdots & I_3\end{bmatrix} \tag{40}$$

Agora, das Inequações (19) e (20), vem a Inequação (41). Como não foram impostas restrições a rotação, convencionou-se dizer que tais restrições são infinito.

$$\underbrace{\begin{bmatrix}-X_{max}\\-Y_{max}\\-\infty\end{bmatrix}}_{e_{min}} \leq \underbrace{\begin{bmatrix}e_1\\e_2\\e_3\end{bmatrix}}_{e} \leq \underbrace{\begin{bmatrix}X_{max}\\Y_{max}\\\infty\end{bmatrix}}_{e_{max}} \tag{41}$$

Colocando em termos do vetor de saídas $\hat{y}$ através das Equações (26) e (30), chega-se a Inequação (42).

$$[y_{min}]_N \leq \hat{y} \leq [y_{max}]_N \tag{42}$$

Combinando a Inequação (42) com a Equação (36), pode-se exprimir as restrições em termos do vetor de incrementos de controle, segundo a Inequação (43).

$$\begin{bmatrix}G\\-G\end{bmatrix}\Delta\hat{u} \leq \begin{bmatrix}[[y_{max}]_N - f]_M\\ [f - [y_{min}]_N]_M\end{bmatrix} \tag{43}$$

Combinando a Inequação (40) com a Inequação (43), chega-se a Inequação (44).

$$\underbrace{\begin{bmatrix}T_M^{I_3}\\-T_M^{I_3}\\G\\-G\end{bmatrix}}_{S}\Delta\hat{u} \leq \underbrace{\begin{bmatrix}[u_{B_{max}} - u(k-1)]_M\\ [u(k-1) - u_{B_{min}}]_M\\ [[y_{max}]_N - f]_M\\ [f - [y_{min}]_N]_M\end{bmatrix}}_{b} \tag{44}$$

Chegando-se por fim em uma equação da forma $S\Delta\hat{u} \leq b$, que descreve todas as restrições do sistema.

3.5 Calculando o controle ótimo

Após certa manipulação algébrica na Equação (31) pode-se chegar na Equação (45). Termos constantes foram removidos da equação por não influenciarem na otimização.

$$J(\Delta\hat{u}) = \frac{1}{2}\Delta\hat{u}^T H_{qp}\Delta\hat{u} + f_{qp}^T\Delta\hat{u},$$
$$\textit{sujeito a } S\Delta\hat{u} \leq b \tag{45}$$

Sendo os termos H_{qp} e f_{qp} definidos na Equação (46).

$$\begin{matrix}H_{qp} = 2(G^TQG + R)\\ f_{qp} = 2(QG)^T(f - ref)\end{matrix} \tag{46}$$

O controle ótimo pode então ser dado pelo problema de otimização de minimizar a função custo dada na Equação (45). Tal problema de otimização é conhecido como Programação Quadrática. Apesar de a cada instante toda a sequencia de controle ser calculada, em um dado instante k, o comando de controle é dado efetivamente por $u_B(k) = u_B(k-1) + [I_3\ 0_{3\times3(M-1)}]\Delta\hat{u}^*$.

4. SIMULAÇÕES

As simulações foram feitas com a utilização do *software* MATLAB/Simulink. Para resolver o problema de programação quadrática foi utilizada a função `quadprog` do MATLAB. As trajetórias de referência foram geradas utilizando curvas de Lissajous apenas para servir como exemplo, cujas equações paramétricas seguem as Equações (47) e (48).

$$X(t) = A_1 sin(\omega_1 t + \delta) \tag{47}$$

$$Y(t) = A_2 sin(\omega_2 t) \tag{48}$$

Sendo A_1, A_2, ω_1 e δ parâmetros.

Para todas as simulações foram usados os parâmetros da Tabela **??** que não correspondem com as restrições reais do robô, foram impostos apenas para testes e simulações.

Parâmetro	Valor
ω_{max}	3.5 rad/s
T	0.005 s
ω_1	1
ω_2	2
A_1	1
A_2	1.5
δ	$\pi/2$
M	20
N	15

Além disso, foi fixado $Q = diag(15, 20, 40)$ e $R = diag(0.05, 0.05, 0.05)$. Vale ressaltar que quanto maior a matriz Q, mais rapidamente a referência será atingida, no entanto, o ruído do sistema se amplifica, servindo a matriz R como um filtro que torna o sistema mais robusto a ruídos.

Para uma melhor comparação com a situação real, o movimento do robô foi simulado sobre um desenho de campo estilo *Small Size*.

4.1 Sem adição de perturbações e ruído

O resultado da simulação sem adição de perturbações e ruído pode ser verificado na Figura 5.

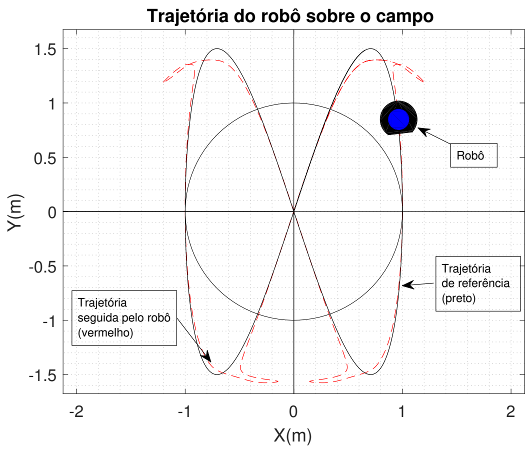


Figura 5. Gráfico do movimento do robô sobre a trajetória planejada. Em preto a trajetória de referência e em vermelho a trajetória realmente cumprida.

4.2 Resposta a pertubações e ruído

Para a simulação com ruído e pertubações foi adicionado um ruído de posição de covariância 0.001 e média zero e um deslocamento em degrau da posição do robô após 1 segundo de simulação. O resultado da simulação com adição de perturbações e ruído pode ser verificado na Figura 6.

Os resultados obtidos foram satisfatórios e mostraram que o robô é capaz de seguir a trajetória e se recuperar de perturbações externas. Levantamentos de requisitos para que os pesos das matrizes possam ser definidos e um modelo do ruído da câmera serão incluídos em trabalhos futuros.

4.3 Análise do respeito a restrições

É possível ainda observar a resposta do controlador a restrições impostas. A restrição de velocidade angular da roda imposta foi intencionalmente restritiva para que esse efeito pudesse ser verificado. Um gráfico comparativo entre a velocidade angular desenvolvida e a máxima permitida na simulação com ruído incluso pode ser verificado na Figura 7. Verifica-se que as restrições foram bem respeitadas pelo controlador.

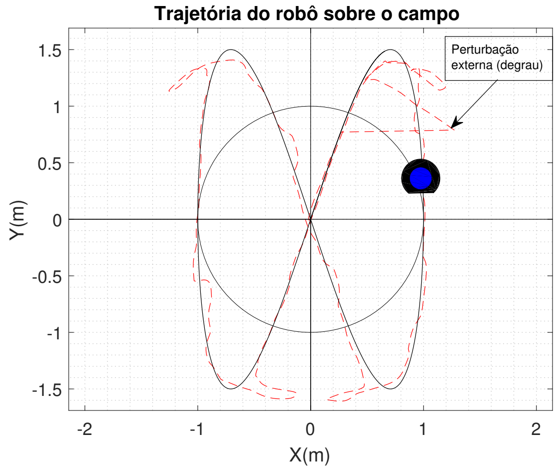


Figura 6. Gráfico do movimento do robô sobre a trajetória planejada. Em preto a trajetória de referência e em vermelho a trajetória realmente cumprida. Destaque para a resposta ao ruído e ao deslocamento após 1 segundo de simulação.

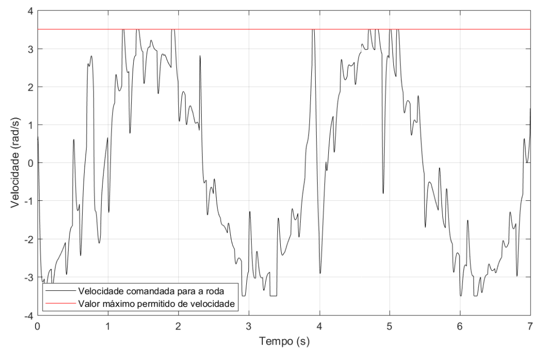


Figura 7. Gráfico comparando o valor máximo permitido para a velocidade angular da roda (em vermelho) e a velocidade comandada pelo controlador (em preto). Nota-se claramente que as restrições foram respeitadas.

5. CONCLUSÃO

Apesar dos desafios que o problema de seguir trajetórias na robótica possa representar, o controlador preditivo se mostrou capaz de, com maestria, contornar esse problema sem se desatentar as restrições que são impostas, conseguindo seguir a trajetória, corrigir perturbações e criar um filtro que barra o ruído sem descumprir as limitações do sistema.

Para trabalhos futuros, deseja-se fazer um levantamento de requisitos através de simulações e observações do robô, além de um levantamento de um modelo de ruído de posição do robô para que os pesos das matrizes Q e R possam ser melhor projetados. Além disso, deseja-se observar quantitativamente o efeito dos horizontes M e N no sistema. Por fim, deseja-se também implementar o controlador no robô e o observar na prática. Seria também interessante estabelecer as diferenças quantitativas entre o preditivo e outros controladores como o PID voltado para o *Small Size*.

6. AGRADECIMENTOS

O autor gostaria de agradecer a equipe ITAndroids, em especial ao prof. Dr Marcos Máximo, por todo o apoio e suporte no trabalho, e aos patrocinadores da equipe e ao ITA por tornarem tais pesquisas possíveis.

REFERÊNCIAS

Camacho, E.F. and Bordons, C.C. (2007). *Model Predictive Control*. Springer, London.

Klančar, G. and Škrjanc, I. (2007). Tracking-error model-based predictive control for mobile robots in real time. *Robotics and Autonomous Systems*, 55(6), 460–469. doi: 10.1016/J.ROBOT.2007.01.002.

Maximo, M.R.O.A. (2012). Model Predictive Controller for Trajetory Tracking by Diferential Drive Robot With Actuation Constraints. 6.

Maximo, M.R.O.A. (2018). ITAndroids Small Size Relatório do Modelamento da Dinâmica do Robô. Technical report.

RoboCup (2019). Rules of the RoboCup Small Size League 2019. 1–33.

Siegwart, R., Nourbakhsh, I.R., and Scaramuzza, D. (2011). *Introduction to autonomous mobile robots*.