VETOR

Resolução do Problema de Estacionamento Paralelo em Tempo Mínimo Utilizando um Novo Pacote de Otimização: *COPILOTS*☆

Solving the Parallel Parking Problem in Minimum Time Using a New Optimization Package: *COPILOTS*

Arthur Henrique Iasbeck[1†], Fran Sérgio Lobato[2], Pedro Augusto Queiroz de Assis[3]

[1]*Programa de Pós-Graduação em Engenharia Mecânica, Universidade Federal de Uberlândia, Uberlândia, Brasil*
[2]*Faculdade de Engenharia Química, Universidade Federal de Uberlândia, Uberlândia, Brasil*
[3]*Faculdade de Engenharia Mecânica, Universidade Federal de Uberlândia, Uberlândia, Brasil*
[†]**Autor correspondente:** arthuriasbeck@ufu.br

Resumo

Este trabalho objetiva apresentar o *COPILOTS* (*Basi**C** **OPt**I**ma**L** C**O**n**T**rol **S**olver*), um pacote proposto para resolução de Problemas de Controle Ótimo (PCOs) baseado nos Métodos Diretos, de fácil utilização e ideal para usuários com pouca ou nenhuma experiência na resolução de PCOs. Em linhas gerais, a ferramenta proposta realiza a discretização dos controles via colocação trapezoidal ou colocação Hermite-Simpson, ao passo que empregam-se polinômios cúbicos na interpolação dos estados. Com efeito, o problema original (algébrico-diferencial) é transformado em um puramente algébrico, que é resolvido considerando o SQP (*Sequential Quadratic Programming*). Para fins de validação, o *COPILOTS* foi empregado na determinação da trajetória de tempo mínimo a ser percorrida por um veículo na execução de uma manobra de estacionamento paralelo e um novo algoritmo para inicialização dos estados e controles foi proposto de forma a minimizar o esforço computacional despendido. Os resultados demonstram que o pacote proposto configura-se como uma alternativa para a resolução de PCOs.

Palavras-chave

COPILOTS • Estacionamento Paralelo • Problema de Controle Ótimo • Colocação Direta • Otimização

Abstract

This work aims to present *COPILOTS* (*Basi**C** **OPt**I**ma**L** C**O**n**T**rol **S**olver*), a proposed package for solving Optimal Control Problems (OCPs) based on Direct Methods, easy to use and ideal for users with little or no experience in solving PCOs. In general, the proposed tool performs the discretization of controls via trapezoidal colocation or Hermite-Simpson colocation, while cubic polynomials are used in the interpolation of the states. In effect, the original problem (algebraic-differential) is transformed into a purely algebraic problem, which is solved considering the SQP (Sequential Quadratic Programming). For validation purposes, *COPILOTS* was used to determine the minimum time path to be traveled by a vehicle when executing a parallel parking maneuver and a new algorithm for initializing states and controls was proposed in order to minimize the computational effort spent. The results demonstrate that the proposed package is configured as an alternative for the resolution of OCPs.

Keywords

COPILOTS • Parallel Parking • Optimal Control Problem • Direct Collocation • Optimization

☆Este artigo é uma versão estendida do trabalho apresentado no XXIII Encontro Nacional de Modelagem Computacional e XI Encontro de Ciência e Tecnologia de Materiais, realizado em Palmas/TO, 2020.

1 Introdução

O Controle Ótimo pode ser definido como um conjunto de métodos empregados na determinação das trajetórias de controle e, consequentemente, de estado em um sistema dinâmico via minimização de um índice de desempenho [1]. As origens do Controle Ótimo remontam ao fim do século XVII, quando, em 1697, Johann Bernoulli apresentou à comunidade científica a solução de um dos primeiros problemas de Controle Ótimo (PCOs), o problema da braquistócrona [2]. Além disso, o Controle Ótimo também possui raízes na Teoria do Controle Clássico, com destaque para a formulação do Filtro de Kalman e do Regulador Linear Quadrático [3]. Neste contexto, aplicações da teoria de Controle Ótimo podem ser verificadas nas áreas da robótica, engenharia aeroespacial, bioengenharia e engenharia química [1].

Dois tipos de abordagens são comumente empregados na resolução de PCOs: os Métodos Diretos e os Métodos Indiretos [4]. Os Métodos Indiretos consistem na transformação do problema original (otimização com restrições algébrico-diferenciais) em um equivalente de simulação algébrico-diferencial de valor no contorno via determinação das condições de otimalidade [3]. Já os Métodos Diretos são baseados na transformação do PCO original em um Problema de Programação Não Linear (PPNL) equivalente a partir da discretização das variáveis de estado e controle [1, 5]. Por fim, os perfis ótimos das variáveis de estado e controle são determinados a partir da resolução do PPNL obtido.

Tradicionalmente, os Métodos Diretos têm servido de base para o desenvolvimento de pacotes computacionais para resolução de PCOs, devido à qualidade dos otimizadores disponíveis e da aplicação direta deste tipo de abordagem [6]. Por exemplo, pacotes computacionais como BOCOP [7], FALCON [8], GEKKO [9], HamPath [10], OpenOCL [11], OptminTraj [12], OpenGoddard [13], Beluga [14], ICLOCS [15], e PSOPT [16] fazem uso dos Métodos Diretos para resolução de PCOs.

Este trabalho tem como objetivo introduzir um novo pacote para a resolução de PCOs via discretização das variáveis de estado e controle, o *COPILOTS* (*Basi**C** **OP**t**I**ma**L** C**O**n**T**rol **S**olver*). Este pacote é escrito em linguagem Matlab® para usuários com pouca ou nenhuma experiência na resolução de PCOs. Para isso, o *COPILOTS* possibilita que, por meio da execução de um único comando, sejam criados, e já parcialmente preenchidos, os *scripts* a serem utilizados na estruturação do PCO, guiando o usuário iniciante. Além disso, o *COPILOTS* apresenta uma sintaxe algébrica próxima à da formulação de Bolza, o que simplifica a implementação do PCO [17].

Fazendo uso de uma das métricas propostas em Febbo et al. [17] - o número de linhas de código - a implementação do problema do estacionamento paralelo, por exemplo, é mais simples no *COPILOTS* se comparada a outros pacotes similares, como indicam os dados introduzidos na Tabela 1. Vale ressaltar que o FALCON.m [18] e o ICLOCS (*Imperial College London Optimal Control Software*) [19] são pacotes escritos em linguagem Matlab®, enquanto o PSOPT (*Pseudospectral Optimal Control Solver*) [1] é uma biblioteca baseada em C++.

Tabela 1: Número de linhas de código utilizadas na implementação do problema do estacionamento paralelo.

Pacote computacional	Número de linhas de código
COPILOTS	149
FALCON.m	165
PSOPT	311
ICLOCS	362

Diferentemente de outros pacotes, como o GPOPS-II [20] e o PROPT [21], também baseados no Matlab®, o *COPILOTS* é gratuito e seu código fonte pode ser acessado e baixado em `https://bitbucket.org/iasbeck/copilots/src/master/`.

Para fins de aplicação, o referido pacote será empregado na resolução do problema do estacionamento paralelo, que consiste na determinação da trajetória a ser percorrida por um veículo para realização de uma manobra de baliza em tempo mínimo. Vale ressaltar que a complexidade desse estudo de caso se deve ao alto número de restrições associadas. Tais restrições impõem limites à velocidade e à aceleração do veículo, de forma a garantir o conforto dos passageiros e evitar o choque com outros veículos já estacionados ou com os limites da via [22]. Por fim, um novo algoritmo para inicialização dos estados e controles será proposto no intuito de minimizar o esforço computacional despendido na resolução do estudo de caso em análise.

Este trabalho está estruturado como segue. Na Seção 2 é introduzida a formulação matemática de um PCO genérico. O pacote *COPILOTS* é apresentado na Seção 3. O estudo de caso que será avaliado é introduzido na Seção 4. Os resultados obtidos são mostrados na Seção 5. Por fim, na última seção, são apresentadas as conclusões obtidas

e algumas sugestões para o desenvolvimento de trabalhos futuros.

2 Problema de Controle Ótimo

O PCO consiste na determinação da lei de controle que promove a minimização de uma função objetivo na presença de restrições algébrico-diferenciais. Matematicamente, este problema pode ser representado como segue [3]:

$$\min_{\mathbf{u}(t)} J = \varphi\big(\mathbf{x}(t_f), t_f\big) + \int_{t_0}^{t_f} L\big(\mathbf{x}(t), \mathbf{u}(t), t\big)\, dt \tag{1a}$$

$$\dot{\mathbf{x}}(t) = \mathbf{f}\big(\mathbf{x}(t), \mathbf{u}(t), t\big) \tag{1b}$$

$$\mathbf{x}(t_0) = \mathbf{x_0}\mathbf{c}(\mathbf{x}(t), \mathbf{u}(t), t) \leq \mathbf{0} \tag{1c}$$

$$\boldsymbol{\psi}(\mathbf{x}(t_f), t_f) \leq \mathbf{0} \tag{1d}$$

$$\mathbf{x_L} \leq \mathbf{x}(t) \leq \mathbf{x_U} \tag{1e}$$

$$\mathbf{u_L} \leq \mathbf{u}(t) \leq \mathbf{u_U} \tag{1f}$$

em que t_0 e t_f representam os instantes de tempo inicial e final, respectivamente, $\mathbf{x}$ é o vetor de variáveis de estado com condição inicial dada por $\mathbf{x_0}$, $\mathbf{u}$ o vetor de variáveis de controle, J é o índice de desempenho (ou função objetivo), formulado a partir das funções de custo de Mayer ϕ e de Lagrange L. $\mathbf{f}$ é o conjunto de equações que descreve a dinâmica do sistema no espaço de estados, $\mathbf{c}$ é o vetor de restrições de desigualdade e $\boldsymbol{\psi}$ é o vetor que contém as restrições terminais. Os limites inferiores e superiores para estados e controles são indicados pelos subscritos L e U.

3 O Pacote *COPILOTS*

O *COPILOTS* é um pacote desenvolvido para a resolução de PCOs descritos por restrições diferenciais de primeira ordem baseado no emprego de Métodos Diretos, também chamados Métodos de Colocação Direta. Neste caso, o usuário pode optar por utilizar a colocação trapezoidal ou a colocação Hermite-Simpson [23].

3.1 O Processo de Discretização

Os Métodos de Colocação Direta são baseados na discretização das variáveis de estado e controle. A resolução do PCO envolve a determinação dos valores ótimos atribuídos às variáveis em certos instantes discretos da trajetória, denominados nós (ou pontos) de colocação, conforme ilustrado na Figura 1. Tais nós costumam ser distribuídos uniformemente no domínio do tempo e o número de nós de colocação empregado na resolução de um dado estudo de caso é normalmente definido a partir de um processo de experimentação numérica.

Uma vez que os estados, os controles, a função objetivo e as restrições associadas à dinâmica do PCO tenham sido discretizados de forma que:

$$t \rightarrow t_0, \ldots, t_k, \ldots, t_M \tag{2a}$$

$$\mathbf{x}(t) \rightarrow \mathbf{x}_0, \ldots, \mathbf{x}_k, \ldots, \mathbf{x}_M \tag{2b}$$

$$\mathbf{u}(t) \rightarrow \mathbf{u}_0, \ldots, \mathbf{u}_k, \ldots, \mathbf{u}_M \tag{2c}$$

sendo ξ_k o valor atribuído à variável ξ no nó de colocação k, é possível que o PCO em análise seja tratado como um problema de otimização clássica, mais especificamente um PPNL. Cabe ressaltar que M é o índice atribuído ao último nó de colocação, localizado em $t = t_f$, e que, assumindo N como sendo o número de nós de colocação, tem-se $M = N - 1$. O processo de conversão de um PCO em um PPNL é conhecido como transcrição [23].

Uma vez que o PPNL tenha sido resolvido e os valores atribuídos a $\mathbf{x}(t)$ e $\mathbf{u}(t)$ nos nós de colocação, denotados respectivamente por $\mathbf{x}_k$ e $\mathbf{u}_k$, tenham sido determinados, é possível que os perfis dos estados e controles sejam construídos a partir interpolação de $\mathbf{x}_k$ e $\mathbf{u}_k$. Nos casos em que as colocações trapezoidal e Hermite-Simpson são empregadas, utilizam-se polinômios de primeiro e segundo grau, respectivamente, para interpolação dos controles, de forma que determinem-se os valores assumidos por esses entre os nós de colocação, e polinômios de terceiro grau para interpolação dos estados [4].

Em suma, a implementação da maioria dos métodos de transcrição é baseada nos cinco passos listados a seguir:

1. Discretização da integral associada à função objetivo;
2. Discretização das restrições dinâmicas;
3. Discretização das restrições de caminho, terminais e laterais;

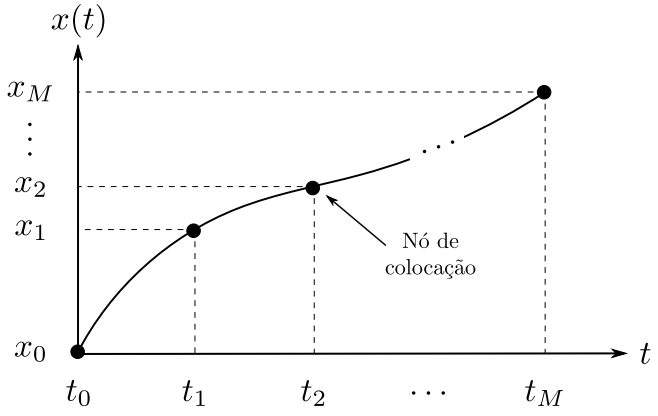

Figura 1: Representação do processo de discretização no qual baseiam-se os Métodos Diretos.

4. Solução do PPNL obtido a partir do processo de transcrição;
5. Elaboração das trajetórias de estados e controles com base em $\mathbf{x}_k$ e $\mathbf{u}_k$ assumindo-se $k = 0, 1, \ldots, M$.

A forma como cada uma dessas etapas será executada depende do método de transcrição adotado. Assim sendo, serão apresentados, nas próximas seções, os métodos que servem de base para a implementação do *COPILOTS*.

3.1.1 Colocação trapezoidal

A colocação trapezoidal é baseada na quadratura trapezoidal, empregada tanto na computação do integrando L associado à função objetivo, quanto na discretização das restrições dinâmicas. Desta forma, assumindo-se $h_k = t_{k+1} - t_k$, computa-se a integral de L conforme a seguinte relação [23]:

$$\int_{t_0}^{t_f} L\big(\mathbf{x}(t), \mathbf{u}(t), t\big)\, dt \approx \sum_{k=0}^{M-1} \frac{1}{2} h_k (L_k + L_{k+1}) \tag{3}$$

sendo $L_k = L\big(\mathbf{x}_k, \mathbf{u}_k, t_k\big)$.

Uma vez que os estados tenham sido discretizados, é possível que as restrições diferenciais associadas à dinâmica do PCO sejam representadas por um conjunto de restrições algébricas. Para tanto, é necessário que as restrições dinâmicas sejam reescritas na forma integral e que a quadratura trapezoidal seja empregada [23]:

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t), t) \tag{4a}$$

$$\int_{t_k}^{t_{k+1}} \dot{\mathbf{x}}(t)\, dt = \int_{t_k}^{t_{k+1}} \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t), t)\, dt \tag{4b}$$

$$\mathbf{x}_{k+1} - \mathbf{x}_k \approx \frac{1}{2} h_k (\mathbf{f}_k + \mathbf{f}_{k+1}) \tag{4c}$$

Assim sendo, M restrições de igualdade algébricas são formuladas a partir da Eq. (4c):

$$\mathbf{x}_{k+1} - \mathbf{x}_k - \frac{1}{2} h_k (\mathbf{f}_{k+1} + \mathbf{f}_k) = \mathbf{0}, \quad k = 0, \ldots, M-1 \tag{5}$$

sendo $\mathbf{f}_k = \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k, t_k)$. Vale ressaltar que $\mathbf{x}_k$ é uma variável de otimização, enquanto $\mathbf{f}_k$ é obtido via avaliação do k-ésimo nó de colocação [23].

As restrições laterais, terminais e de caminho são incorporadas à formulação do PPNL, sendo transformadas em restrições de igualdade e desigualdade. Tal transformação se dá pela imposição dessas restrições nos nós de colocação [23]. Assim sendo, as restrições associadas ao PCO podem ser transcritas da seguinte forma:

$$\mathbf{x_L} \leq \mathbf{x}(t) \leq \mathbf{x_U} \rightarrow \mathbf{x_L} \leq \mathbf{x}_k \leq \mathbf{x_U},\ k = 0, \ldots, M \tag{6a}$$

$$\mathbf{u_L} \leq \mathbf{u}(t) \leq \mathbf{u_U} \rightarrow \mathbf{u_L} \leq \mathbf{u}_k \leq \mathbf{u_U},\ k = 0, \ldots, M \tag{6b}$$

$$\mathbf{c}(\mathbf{x}(t), \mathbf{u}(t), t) \leq \mathbf{0} \rightarrow \mathbf{c}(\mathbf{x}_k, \mathbf{u}_k, t_k) \leq \mathbf{0},\ k = 0, \ldots, M \tag{6c}$$

$$\boldsymbol{\psi}(\mathbf{x}(t_f), t_f) \leq \mathbf{0} \rightarrow \boldsymbol{\psi}(\mathbf{x}_M, t_M) \leq \mathbf{0} \tag{6d}$$

em que a condição inicial $\mathbf{x_0}$ para o vetor de variáveis de estado é representada por:

$$\mathbf{x}_k = \mathbf{x_0}, \quad k = 0 \tag{7}$$

Por fim, o emprego da colocação trapezoidal na avaliação do PCO introduzido na Eq. (1) resulta no seguinte PPNL:

$$\min_{\mathbf{u}_k, \mathbf{x}_k} J = \varphi\big(\mathbf{x}_M, t_M\big) + \sum_{k=0}^{M-1} \frac{1}{2} h_k (L_k + L_{k+1}) \tag{8a}$$

$$\mathbf{x}_{k+1} - \mathbf{x}_k - \frac{1}{2}\, h_k \left(\mathbf{f}_{k+1} + \mathbf{f}_k\right) = \mathbf{0}, \quad k = 0, \ldots, M-1 \tag{8b}$$

$$\mathbf{x}(t_0) = \mathbf{x_0} \tag{8c}$$

$$\mathbf{x_L} \leq \mathbf{x}_k \leq \mathbf{x_U}, \quad k = 0, \ldots, M \tag{8d}$$

$$\mathbf{u_L} \leq \mathbf{u}_k \leq \mathbf{u_U}, \quad k = 0, \ldots, M \tag{8e}$$

$$\mathbf{c}(\mathbf{x}_k, \mathbf{u}_k, t_k) \leq \mathbf{0} \quad k = 0, \ldots, M \tag{8f}$$

$$\boldsymbol{\psi}(\mathbf{x}_M, t_M) \leq \mathbf{0} \tag{8g}$$

em que t_0 e t_f são conhecidos. Não sendo conhecidos, t_0 e t_f devem ser determinados de forma que as seguintes restrições sejam respeitadas:

$$t_{0L} \leq t_0 \leq t_{0U} \tag{9a}$$

$$t_{fL} \leq t_f \leq t_{fU} \tag{9b}$$

sendo os limites inferiores de t_0 e t_f denotados por t_{0L} e t_{fL} e os superiores por t_{0U} e t_{fU}, respectivamente.

3.1.2 Colocação de Hermite-Simpson

A colocação Hermite-Simpson é baseada na quadratura de Simpson, empregada tanto na computação do integrando L associado à função objetivo, quanto na discretização das restrições dinâmicas. Para que a quadratura de Simpson possa ser implementada é necessário que nós de colocação intermediários, posicionados entre os nós de colocação principais, sejam definidos [23]. As grandezas associadas aos nós intermediários são representadas utilizando-se uma barra. Por exemplo, os valores assumidos pelos controles nos nós intermediários são denotados por $\overline{\mathbf{u}}_k$, $k = 0, 1, \ldots, M-1$. Mais especificamente, o valor atribuído ao controle $u(t)$ no nó de colocação intermediário posicionado entre os nós k e $k+1$ é denotado por $\overline{u}_k$.

Assumindo-se $h_k = t_{k+1} - t_k$, computa-se a integral de L da seguinte forma:

$$\int_{t_0}^{t_f} L\big(\mathbf{x}(t), \mathbf{u}(t), t\big)\, dt \approx \sum_{k=0}^{M-1} \frac{1}{6} h_k (L_k + 4\,\overline{L}_k + L_{k+1}) \tag{10}$$

sendo $L_k = L\big(\mathbf{x}_k, \mathbf{u}_k, t_k\big)$ e $\overline{L}_k = L\big(\overline{\mathbf{x}}_k, \overline{\mathbf{u}}_k, \overline{t}_k\big)$ [23].

Enquanto $\overline{\mathbf{x}}_k$ pode ser computado a partir dos valores atribuídos a $\mathbf{x}_k$ e $\mathbf{x}_{k+1}$, como será mostrado adiante, e $\overline{t}_k = \dfrac{t_k + t_{k+1}}{2}$, $\overline{\mathbf{u}}_k$ deve ser uma variável de projeto assim como $\mathbf{u}_k$ e $\mathbf{x}_k$ [23]. Nota-se, portanto, que o PPNL formulado com base na colocação Hermite-Simpson possui mais variáveis de otimização que aquele obtido por meio da colocação trapezoidal, considerando-se que o mesmo número de nós de colocação seja utilizado em ambos os casos.

Após a discretização do vetor de variáveis de estados, é possível que as restrições diferenciais associadas à dinâmica do PCO sejam representadas por um conjunto de restrições algébricas. Para tanto, é necessário que as restrições dinâmicas sejam reescritas na forma integral e que a quadratura de Simpson seja empregada [23]:

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t), t) \tag{11a}$$

$$\int_{t_k}^{t_{k+1}} \dot{\mathbf{x}}(t)\, dt = \int_{t_k}^{t_{k+1}} \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t), t)\, dt \tag{11b}$$

$$\mathbf{x}_{k+1} - \mathbf{x}_k \approx \frac{1}{6} h_k (\mathbf{f}_k + 4\,\overline{\mathbf{f}}_k + \mathbf{f}_{k+1}) \tag{11c}$$

Assim sendo, M restrições de igualdade algébricas são formuladas com base nesta aproximação, isto é:

$$\mathbf{x}_{k+1} - \mathbf{x}_k - \frac{1}{6} h_k (\mathbf{f}_k + 4\,\overline{\mathbf{f}}_k + \mathbf{f}_{k+1}) = \mathbf{0}, \quad k = 0, \ldots, M-1 \tag{12}$$

sendo $\mathbf{f}_k = \mathbf{f}(\mathbf{x}_k, \mathbf{u}_k, t_k)$ e $\overline{\mathbf{f}}_k = \mathbf{f}(\overline{\mathbf{x}}_k, \overline{\mathbf{u}}_k, \overline{t}_k)$ obtidos a partir da computação da dinâmica do sistema.

Para que $\overline{\mathbf{f}}_k$ possa ser computado, é necessário antes que $\overline{\mathbf{x}}_k$ seja determinado empregando-se a interpolação de Hermite:

$$\overline{\mathbf{x}}_k = \frac{1}{2}(\mathbf{x}_k + \mathbf{x}_{k+1}) + \frac{h_k}{8}(\mathbf{f}_k - \mathbf{f}_{k+1}) \tag{13}$$

Assim sendo, é possível que $\overline{\mathbf{x}}_k$ seja computado diretamente, ou que $\overline{\mathbf{x}}_k$ seja considerado uma variável de otimização. Nesse último caso, é necessário que outras M restrições de igualdade algébricas, definidas a partir da Eq. (13), sejam acrescentadas ao PPNL:

$$\overline{\mathbf{x}}_k - \frac{1}{2}(\mathbf{x}_k + \mathbf{x}_{k+1}) - \frac{h_k}{8}(\mathbf{f}_k - \mathbf{f}_{k+1}) = \mathbf{0}, \quad k = 0, \ldots, M-1 \tag{14}$$

Analogamente ao que foi desenvolvido para a colocação trapezoidal, as restrições associadas ao PCO podem ser transcritas da seguinte forma:

$$\mathbf{x}_\mathbf{L} \leq \mathbf{x}(t) \leq \mathbf{x}_\mathbf{U} \rightarrow \begin{cases} \mathbf{x}_\mathbf{L} \leq \mathbf{x}_k \leq \mathbf{x}_\mathbf{U}, \ k = 0, \ldots, M \\ \mathbf{x}_\mathbf{L} \leq \overline{\mathbf{x}}_k \leq \mathbf{x}_\mathbf{U}, \ k = 0, \ldots, M-1 \end{cases} \tag{15a}$$

$$\mathbf{u}_\mathbf{L} \leq \mathbf{u}(t) \leq \mathbf{u}_\mathbf{U} \rightarrow \begin{cases} \mathbf{u}_\mathbf{L} \leq \mathbf{u}_k \leq \mathbf{u}_\mathbf{U}, \ k = 0, \ldots, M \\ \mathbf{u}_\mathbf{L} \leq \overline{\mathbf{u}}_k \leq \mathbf{u}_\mathbf{U}, \ k = 0, \ldots, M-1 \end{cases} \tag{15b}$$

$$\mathbf{c}(\mathbf{x}(t), \mathbf{u}(t), t) \leq \mathbf{0} \rightarrow \begin{cases} \mathbf{c}(\mathbf{x}_k, \mathbf{u}_k, t_k) \leq \mathbf{0}, \ k = 0, \ldots, M \\ \mathbf{c}(\overline{\mathbf{x}}_k, \overline{\mathbf{u}}_k, \overline{t}_k) \leq \mathbf{0}, \ k = 0, \ldots, M-1 \end{cases} \tag{15c}$$

$$\boldsymbol{\psi}(\mathbf{x}(t_f), t_f) \leq \mathbf{0} \rightarrow \boldsymbol{\psi}(\mathbf{x}_M, t_M) \leq \mathbf{0} \tag{15d}$$

em que a condição inicial é dada por:

$$\mathbf{x}_k = \mathbf{x}_\mathbf{0}, \quad k = 0 \tag{16}$$

Por fim, o emprego da colocação Hermite-Simpson na avaliação do PCO introduzido na Eq. (1) resulta no seguinte PPNL:

$$\min_{\overline{\mathbf{u}}_k, \mathbf{u}_k, \mathbf{x}_k} J = \varphi(\mathbf{x}_M, t_M) + \sum_{k=0}^{M-1} \frac{1}{6} h_k (L_k + 4\overline{L}_k + L_{k+1}) \tag{17a}$$

$$\mathbf{x}_{k+1} - \mathbf{x}_k - \frac{1}{6} h_k (\mathbf{f}_k + 4\overline{\mathbf{f}}_k + \mathbf{f}_{k+1}) = \mathbf{0}, \qquad k = 0, \ldots, M-1 \tag{17b}$$

$$\mathbf{x}(t_0) = \mathbf{x}_\mathbf{0} \tag{17c}$$

$$\mathbf{x}_\mathbf{L} \leq \mathbf{x}_k \leq \mathbf{x}_\mathbf{U}, \quad k = 0, \ldots, M \tag{17d}$$

$$\mathbf{x}_\mathbf{L} \leq \overline{\mathbf{x}}_k \leq \mathbf{x}_\mathbf{U}, \quad k = 0, \ldots, M-1 \tag{17e}$$

$$\mathbf{u}_\mathbf{L} \leq \mathbf{u}_k \leq \mathbf{u}_\mathbf{U}, \quad k = 0, \ldots, M \tag{17f}$$

$$\mathbf{u}_\mathbf{L} \leq \overline{\mathbf{u}}_k \leq \mathbf{u}_\mathbf{U} \quad k = 0, \ldots, M-1 \tag{17g}$$

$$\mathbf{c}(\mathbf{x}_k, \mathbf{u}_k, t_k) \leq \mathbf{0}, \quad k = 0, \ldots, M \tag{17h}$$

$$\mathbf{c}(\overline{\mathbf{x}}_k, \overline{\mathbf{u}}_k, \overline{t}_k) \leq \mathbf{0}, \quad k = 0, \ldots, M-1 \tag{17i}$$

$$\boldsymbol{\psi}(\mathbf{x}_M, t_M) \leq \mathbf{0} \tag{17j}$$

para t_0 e t_f conhecidos. Não sendo esse o caso, devem-se adotar t_0 e t_f como variáveis de projeto, e as restrições:

$$t_{0L} \leq t_0 \leq t_{0U} \tag{18a}$$

$$t_{fL} \leq t_f \leq t_{fU} \tag{18b}$$

em que t_{0L} e t_{fL} representam os limites inferiores e t_{0U} e t_{fU} os limites superiores.

3.2 Resolução do PPNL

Após a discretização do PCO, o problema original é reescrito como um equivalente puramente algébrico, sendo este resolvido empregando-se o SQP, que tem como premissa a simplificação do problema de otimização original a partir do emprego de aproximações quadráticas para a função objetivo e da linearização das restrições via série de Taylor. Com a resolução do NLP advindo da transcrição do PCO, determinam-se os valores ótimos referente aos estados e controles nos nós de colocação [23].

3.3 Instalação do Pacote *COPILOTS*

O processo de instalação do *COPILOTS* divide-se em duas etapas. Primeiramente, faz-se o *download* do código fonte em `https://bitbucket.org/iasbeck/copilots/src/master/` e adiciona-se ao *path* do Matlab® [24] a pasta em que o mesmo tenha sido armazenado. Em seguida, executa-se no Matlab® o comando `copilotsSetup`.

Uma vez instalado o pacote, a resolução de um PCO no *COPILOTS* também é um processo constituído de duas etapas, como indicado na Fig. 2. Primeiramente, executa-se o comando `copilotsNew` para criação da pasta *new*, que contém os *scripts* a serem preenchidos para implementação do PCO, sendo então definidas a função objetivo e as restrições algébrico-diferenciais associadas ao PCO. Uma vez realizado este preenchimento, executa-se o comando `copilots` dentro da pasta *new* para que se inicie o processo de obtenção da solução. Ao fim da execução, a pasta *results* será criada para armazenamento dos resultados obtidos (trajetórias ótimas de estados e controles e parâmetros associados à solução do PPNL). A interpolação e representação gráfica das trajetórias ótimas são geradas e armazenadas de forma automática, e são, por padrão, apresentadas ao final da execução.

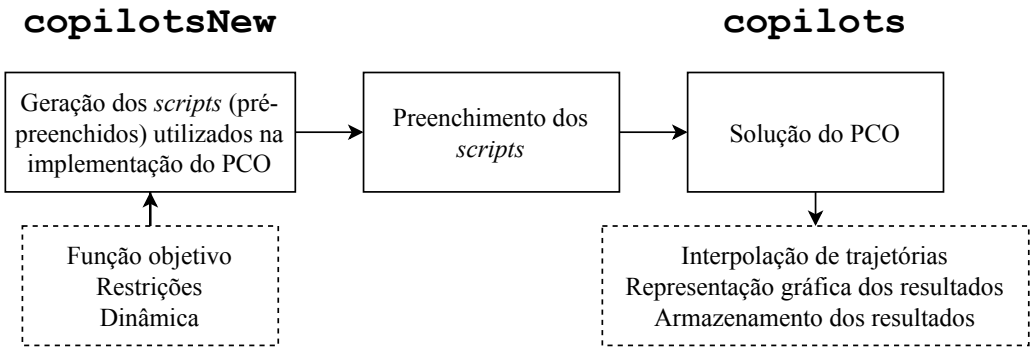

Figura 2: Etapas do processo de implementação de um PCO no *COPILOTS*.

Vale ressaltar que o usuário pode alterar o código fonte caso necessário, sendo possível que novas estratégias para a discretização dos estados e controles sejam incorporadas. Além disso, o pacote encontra-se organizado em módulos e foi desenvolvido com base no paradigma da Programação Orientada a Objetos, o que facilita a compreensão do código fonte e a realização de atualizações [25].

Por fim, vale ressaltar que tanto o código fonte do *COPILOTS* quanto os arquivos gerados após a execução do comando *copilotsNew* estão possuem diversos comentários que orientam a utilização do pacote. Mais ainda, acompanham o código fonte exemplos envolvendo diferentes aplicações que podem ser utilizadas como base para implementação de novos problemas.

4 Problema do Estacionamento Paralelo em Tempo Mínimo

Com o objetivo de verificar a qualidade das soluções obtidas pelo *COPILOTS*, considera-se o problema do estacionamento paralelo em tempo mínimo [26]. As variáveis empregadas na formulação desse problema, bem como as posições inicial e final do automóvel, são apresentadas nas Figs. 3(a)-(c).

O modelo que descreve o comportamento dinâmico do veículo é dado por [26]:

$$\dot{x}(t) = v(t)\cos\theta(t) \tag{19a}$$
$$\dot{y}(t) = v(t)\,\mathrm{sen}\,\theta(t) \tag{19b}$$
$$\dot{v}(t) = a(t) \tag{19c}$$
$$\dot{a}(t) = j(t) \tag{19d}$$
$$\dot{\theta}(t) = \frac{v(t)\tan\phi(t)}{l} \tag{19e}$$
$$\dot{\phi}(t) = \omega(t) \tag{19f}$$

em que l é a distância entre os eixos do automóvel, m e n representam os afastamentos entre as extremidades e os eixos traseiro e dianteiro, respectivamente, e b é metade da largura do veículo. A velocidade e a aceleração do ponto médio P do eixo traseiro são representadas por v e a, respectivamente. Os pontos A, B, C e D são as extremidades do retângulo que delimita o automóvel. O ângulo entre a velocidade do ponto médio do eixo dianteiro e a reta γ é dado

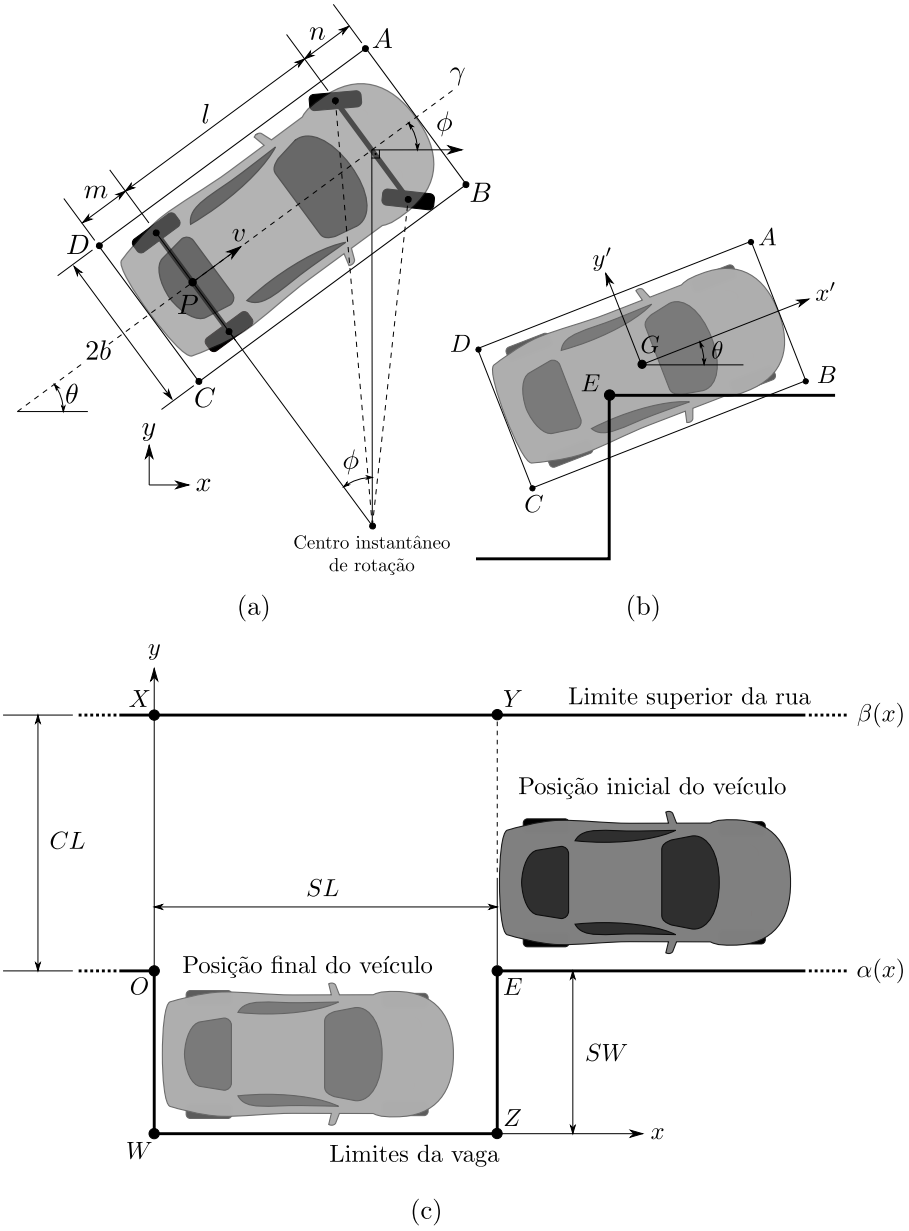

Figura 3: (a) Variáveis empregadas na formulação do problema do estacionamento paralelo. (b) Choque do automóvel com os limites da vaga. (c) Posições assumidas pelo veículo no início e no final da manobra. Adaptado de Li et al. [26].

por ϕ. A inclinação de γ em relação ao eixo x é dada por θ. A sobre-aceleração e a velocidade angular referente à ϕ são denotados respectivamente por j e ω.

Visando garantir o conforto dos passageiros e a redução do estresse sobre os atuadores, são impostas as seguintes restrições:

$$|a(t)| \leq 0{,}75 \text{ m/s}^2 \tag{20a}$$
$$|v(t)| \leq 2 \text{ m/s} \tag{20b}$$
$$|\phi(t)| \leq 0{,}58 \text{ rad} \tag{20c}$$
$$|j(t)| \leq 0{,}5 \text{ m/s}^3 \tag{20d}$$
$$|\kappa'(t)| \leq 0{,}6 \text{ 1/(m s)} \tag{20e}$$

em que $\kappa'(t) = \omega(t)/\left(l \cos^2 \phi(t)\right)$ é a derivada da curvatura instantânea, associada a limitações na velocidade angular $\omega(t)$ [26].

Todas as posições assumidas pelo veículo na execução da manobra devem estar contidas na região delimitada

pelas curvas $\alpha(x) = \big(H(x - SL) - H(x)\big)SW$ e $\beta(x) = CL$, sendo $H(x)$ a função degrau unitário, SL e SW o comprimento e largura da vaga, respectivamente, e CL a largura da via. Além disso, as relações entre a posição do veículo e os pontos A, B, C e D são dadas por:

$$A = (A_x, A_y) = \big(x + (l+n)\cos\theta - b\,\mathrm{sen}\,\theta,\ y + (l+n)\,\mathrm{sen}\,\theta + b\cos\theta\big) \tag{21a}$$

$$B = (B_x, B_y) = \big(x + (l+n)\cos\theta + b\,\mathrm{sen}\,\theta,\ y + (l+n)\,\mathrm{sen}\,\theta - b\cos\theta\big) \tag{21b}$$

$$C = (C_x, C_y) = \big(x - m\cos\theta + b\,\mathrm{sen}\,\theta,\ y - m\,\mathrm{sen}\,\theta - b\cos\theta\big) \tag{21c}$$

$$D = (D_x, D_y) = \big(x - m\cos\theta - b\,\mathrm{sen}\,\theta,\ y - m\,\mathrm{sen}\,\theta + b\cos\theta\big) \tag{21d}$$

Para garantir que o veículo não se choque com os limites da via ou da vaga são impostas as seguintes restrições:

$$A_y(t) \leq \beta(x) \tag{22a}$$

$$B_y(t) \leq \beta(x) \tag{22b}$$

$$C_y(t) \leq \beta(x) \tag{22c}$$

$$D_y(t) \leq \beta(x) \tag{22d}$$

$$A_y(t) \geq \alpha(A_x(t)) \tag{22e}$$

$$B_y(t) \geq \alpha(B_x(t)) \tag{22f}$$

$$C_y(t) \geq \alpha(C_x(t)) \tag{22g}$$

$$D_y(t) \geq \alpha(D_x(t)) \tag{22h}$$

No entanto, como indicado na Fig. 3(b), mesmo que as restrições descritas pelas Eqs. (22a)-(22h) sejam satisfeitas, ainda é possível que ocorram colisões, uma vez que tais restrições dependem apenas das posições das extremidades A, B, C e D do veículo. Assim sendo, considerando um novo sistema de eixos $(x'Gy')$ redefinem-se os pontos O e E da seguinte forma:

$$O_{x'Gy'} = (O'_x, O'_y) = \left(-x\cos(\theta) - y\,\mathrm{sen}(\theta) - \frac{l+n-m}{2},\ x\,\mathrm{sen}(\theta) - y\cos(\theta) \right) \tag{23a}$$

$$E_{x'Gy'} = (E'_x, E'_y) = \left(-x\cos(\theta) - y\,\mathrm{sen}(\theta) - \frac{l+n-m}{2} + SL\cos(\theta),\ x\,\mathrm{sen}(\theta) - y\cos(\theta) - SL\,\mathrm{sen}(\theta) \right) \tag{23b}$$

Então, para garantir que as laterais do automóvel não se choquem com os pontos O e E durante a execução da manobra, deve-se admitir que:

$$|O'_x| \geq (l+m+n)/2, \text{ quando } |O'_y| \leq b \tag{24a}$$

$$|E'_x| \geq (l+m+n)/2, \text{ quando } |E'_y| \leq b \tag{24b}$$

Consideram-se como condições iniciais: $x(0) = SL + m$, $y(0) = 1{,}5$ m, e $v(0) = a(0) = \theta(0) = \phi(0) = 0$. Como condições terminais têm-se: $m \leq x(t_f) \leq SL - (l+n)$, $-(SW - b) \leq y(t_f) \leq 0$, e $v(t_f) = a(t_f) = \theta(t_f) = 0$. Neste estudo de caso deseja-se determinar os perfis das variáveis de estado ($x(t)$, $y(t)$, $v(t)$, $a(t)$, $\theta(t)$ e $\phi(t)$) e de controle ($j(t)$ e $\omega(t)$) de forma a minimizar o tempo de execução da manobra ($J = t_f$). Para esta finalidade, os seguintes parâmetros foram adotados [26]: $l = 2{,}588$ m, $n = 0{,}839$ m, $m = 0{,}657$ m, $b = 0{,}8855$ m, $SL = 6$ m, $SW = 2$ m, e $CL = 3{,}5$ m.

A implementação completa do estudo de caso em análise pode ser verificada em `https://bitbucket.org/iasbeck/copilots/src/master/examples/estacionamento/`.

5 Resultados e Discussão

Para obtenção de uma trajetória factível, propõe-se em [26] a introdução de uma região crítica, delimitada pelos pontos X, Y, Z e W (ver a Fig. 3), e a resolução de uma série de N_{fe} PCOs semelhantes ao original. Neste contexto, na formulação do N_χ-ésimo PCO, consideram-se restrições que garantam que o veículo esteja contido na região crítica para $t \in [h \cdot N_\chi,\ tf]$, sendo $h = t_f/N_{fe}$ e $N_\chi = 1, ..., N_{fe}$. Então, a solução do primeiro PCO é utilizada como estimativa inicial para resolução do segundo, e assim sucessivamente, até que o PCO original seja resolvido para $N_\chi = N_{fe}$.

Visando minimizar o esforço computacional envolvido na inicialização do PPNL, uma abordagem distinta é aqui proposta. Neste caso, para a obtenção da solução são necessárias apenas duas execuções. Na primeira, desconsideram-se as restrições descritas pelas Eqs. (24a) e (24b), uma vez que estas são não convexas e dificultam a obtenção de

trajetórias viáveis [23]. Então, na segunda execução, as restrições inicialmente desconsideradas são reinseridas no problema e a solução obtida anteriormente é utilizada na inicialização do *COPILOTS*.

Cabe ressaltar que a resolução do PCO em análise pode ainda ser dificultada pela representação matemática dos limites da vaga, que possui descontinuidades em $x = 0$ e $x = SL$. Para evitar a divergência durante o processo de otimização, os limites da vaga são aqui representados pela soma de funções sigmoides [23]. Uma vez que as restrições introduzidas nas Eqs. (24a) e (24b) também são descontínuas, a mesma abordagem foi empregada na representação.

Um dos critérios comumente utilizados para avaliar o desempenho de pacotes numéricos é o tempo de processamento. No caso deste trabalho, assumiu-se que o tempo total de processamento (t_p) é a soma dos tempos despendidos pelas duas execuções necessárias para obtenção da solução. Para a determinação do tempo médio, considerou-se a média dos tempos despendidos na obtenção de cinco soluções. O *COPILOTS* foi executado, neste caso, em um computador com 8 GB de memória RAM e um processador Intel(R) Core(TM) i7-7500U CPU com 2.70 GHz. Além disso, foram adotados os parâmetros de execução *default* do *COPILOTS*.

Além do tempo de processamento médio t_p, dado em segundos, e do desvio padrão associado s_t, outros critérios foram também empregados na avaliação do desempenho do *COPILOTS*. Mais especificamente, o valor ótimo da função objetivo J^*, o número de avaliações n_{aval} do funcional J e a máxima violação das restrições Δr_{max} [27, 28, 29]. Na Tabela 2 são apresentados os resultados obtidos a partir da resolução do problema do estacionamento paralelo em tempo mínimo considerando-se o $COPILOTS_t$ (aproximação via colocação trapezoidal), o $COPILOTS_h$ (aproximação via colocação de Hermite-Simpson) e os reportados em Li et al. [26]. Nesta tabela pode-se observar que estão atribuídos ao *COPILOTS* valores de J^* bastante próximos aos reportados em Li et al. [26]. Todavia, os números de nós de colocação associados a este pacote foram menores que o utilizado em Li et al. [26]. Vale ressaltar que propõe-se em Li et al. [26] a utilização de polinômios de Lagrange de terceira ordem para interpolação dos estados e controles. Tais polinômios são computados a partir dos valores assumidos pelos estados e controles nos nós de colocação principais e intermediários, tendo sido utilizados nesse caso dois nós intermediários. Os números de nós de colocação requeridos pelo *COPILOTS* foram determinados a partir de testes preliminares, em que observou-se o número mínimo de nós que levava a um resultado próximo ao reportado na literatura. Em termos do tempo de processamento, também observa-se uma diferença significativa entre os resultados obtidos e o reportado em Li et al. [26]. A diferença entre os tempos de processamento está relacionada ao número de nós de colocação associado ao *COPILOTS*. Como era esperado, observa-se que o valor de n_{aval} atribuído ao $COPILOTS_t$ é menor que aquele associado ao $COPILOTS_h$, dadas as diferentes propriedades numéricas atribuídas a cada tipo de colocação. Finalmente, observa-se que os valores de Δr_{max} se mostraram bem próximos de zero.

Tabela 2: Resultados obtidos para o problema do estacionamento paralelo em tempo mínimo. Os dados não informados são representados por "–".

Método	N	J^* [s]	t_p [s]	s_t	n_{aval}	Δr_{max}
$COPILOTS_t$	20	7,581	6,607	0,011	165	$9{,}99\times10^{-16}$
$COPILOTS_h$	10	7,522	6,717	0,067	8836	$2{,}22\times10^{-15}$
Li et al. [26]	40	7,521	41,868	–	–	–

Na Figura 4 são apresentadas as posições assumidas pelo veículo durante a manobra de estacionamento. A curva que conecta os pontos P_0 e P_f representa a evolução da trajetória ótima do ponto $P(x, y)$. Já as trajetórias ótimas dos controles ($j(t)$ e $\omega(t)$) e estados $v(t)$, $a(t)$, $\theta(t)$ e $\phi(t)$ são apresentadas na Fig. 5. De modo geral, os resultados apresentados demonstram que os perfis associados ao *COPILOTS* são bastante próximos aos reportados em Li et al. [26]. No entanto, é importante destacar que os perfis de controle apresentados em Li et al. [26] contém oscilações próximas ao zero do eixo das ordenadas, que ocorrem devido a variações bruscas nos valores de a e ϕ respectivamente, dadas as relações entre j, a, ω e ϕ introduzidas nas Eqs. (19d) e (19f).

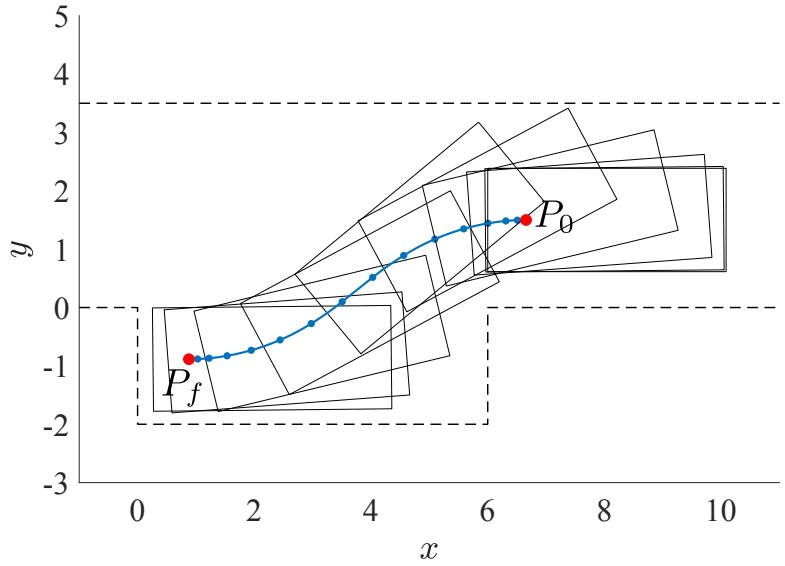

Figura 4: Posições ótimas do veículo durante a execução da manobra de estacionamento. A trajetória do ponto $P(x, y)$ é representada pela curva que conecta os pontos P_0 a P_f.

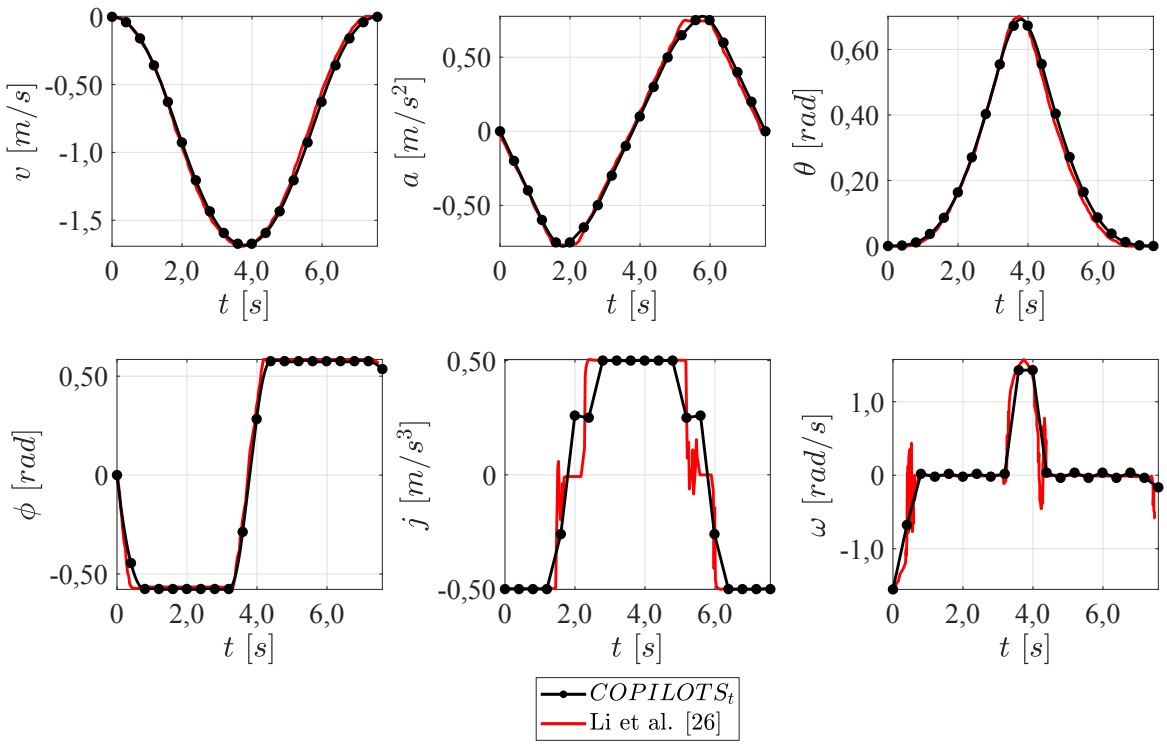

Figura 5: Comparação entre os resultados obtidos por meio do *COPILOTS*, considerando-se o emprego da colocação trapezoidal, e aqueles reportados em Li et al. [26].

É importante enfatizar que durante a avaliação da qualidade das soluções obtidas pelo *COPILOTS* é necessário que se leve em conta que a abordagem aqui proposta para inicialização do problema é consideravelmente mais simples que aquela empregada em Li et al. [26], já que envolve apenas duas execuções.

6 Conclusões

O presente trabalho teve como objetivo apresentar um novo pacote para a resolução de PCOs, o *COPILOTS*. Em linhas gerais, este pacote promove a discretização das variáveis de estado e controle a partir do emprego das colocações trapezoidal e de Hermite-Simpson. Foi apresentada uma visão geral sobre as funcionalidades do pacote e sobre sua utilização. Para fins de validação, o *COPILOTS* foi empregado para determinação da trajetória a ser percorrida por um automóvel durante uma manobra de estacionamento paralelo realizada em tempo mínimo. Os resultados obtidos demonstraram que, empregando a ferramenta proposta, obtém-se boas estimativas para os perfis de estado, de controle e para o valor da função objetivo.

Dentre as principais vantagens do *COPILOTS* destacam-se a simplicidade e a geração automática dos *scripts* utilizados na implementação dos PCOs. Neste caso, é possível que usuários com pouca ou nenhuma experiência na resolução de PCOs possam utilizar esse pacote como um ponto de partida.

Como propostas de trabalho futuro, pretende-se oferecer a possibilidade de utilizar outros otimizadores não nativos do Matlab®, como, por exemplo, o IPOPT (*Interior Point OPTimizer*) e empregar ferramentas de cálculo simbólico do Matlab® na obtenção de derivadas analíticas (da função objetivo e das restrições), o que pode resultar na redução do tempo de processamento e do número de avaliações da função objetivo.

Referências

[1] V. M. Becerra, *PSOPT Optimal Control Solver User Manual*, 1ª ed., 2019. Disponível em: https://github.com/PSOPT/psopt/releases/tag/V4.0.0

[2] H. J. Sussmann e J. C. Willems, "300 years of optimal control: From the brachystochrone to the maximum principle," *IEEE Control Systems Magazine*, vol. 3, no. 17, pp. 32–44, 1997. Disponível em: https://doi.org/10.1109/37.588098

[3] A. E. Bryson, "Optimal control-1950 to 1985," *IEEE Control Systems*, vol. 3, no. 16, pp. 26–33, 1996. Disponível em: https://doi.org/10.1109/37.506395

[4] J. T. Betts, *Practical Methods for Optimal Control and Estimation Using Nonlinear Programming*, 2ª ed. Advances in Design and Control: Society for Industrial and Applied Mathematics, 2010. Disponível em: https://doi.org/10.1137/1.9780898718577

[5] F. Biral, E. Bertolazzi, e P. Bosetti, "Notes on numerical methods for solving optimal control problems," *IEEJ Journal of Industry Applications*, vol. 5, no. 1, pp. 154–166, 2016. Disponível em: https://doi.org/10.1541/ieejjia.5.154

[6] A. Wächter e L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Mathematical Programming*, vol. 1, no. 106, pp. 25–57, 2006. Disponível em: https://doi.org/10.1007/s10107-004-0559-y

[7] I. Saclay, "Bocop: an open source toolbox for optimal control," 2017. Disponível em: http://bocop.org

[8] M. Rieck, M. Bittner, B. Grüter, J. Diepolder, e P. Piprek, "FALCON.m User Guide," 2020. Disponível em: https://www.fsd.lrg.tum.de/software/wp-content/uploads/UserGuideMain.pdf

[9] L. Beal, D. Hill, R. Martin, e J. Hedengren, "GEKKO Optimization Suite," *Processes*, vol. 6, no. 8, p. 106, Jul. 2018. Disponível em: https://doi.org/10.3390/pr6080106

[10] J.-B. Caillau, O. Cots, e J. Gergaud, "Differential continuation for regular optimal control problems," *Optimization Methods and Software*, vol. 27, no. 2, pp. 177–196, 2012, publisher: Taylor & Francis. Disponível em: https://doi.org/10.1080/10556788.2011.593625

[11] J. Koenemann, G. Licitra, M. Alp, e M. Diehl, "OpenOCL–Open Optimal Control Library," 2017. Disponível em: https://openocl.github.io/

[12] M. Kelly, "OptimTraj Users Guide," 2018. Disponível em: https://github.com/MatthewPeterKelly/OptimTraj/blob/master/docs/UsersGuide/OptimTraj_UsersGuide.pdf

[13] "OpenGoddard," Dec. 2017. Disponível em: https://github.com/istellartech/OpenGoddard

[14] "Beluga," Jan. 2018. Disponível em: https://github.com/Rapid-Design-of-Systems-Laboratory/beluga

[15] P. Falugi, E. Kerrigan, e E. van Wyk, "ICLOCS2: A MATLAB Toolbox for Optimization Based Control - Downloads," 2018. Disponível em: http://www.ee.ic.ac.uk/ICLOCS/Downloads.html

[16] V. M. Becerra, "PSOPT Optimal Control Solver User Manual," p. 437, 2019.

[17] H. Febbo, P. Jayakumar, J. L. Stein, e T. Ersal. (2020) NLOptControl: A modeling language for solving optimal control problems. arXiv:2003.00142. Disponível em: https://arxiv.org/abs/2003.00142

[18] M. Rieck, M. Bittner, B. Grüter, J. Diepolder, e P. Piprek., *FALCON.m: User Guide*, 1ª ed., Institute of Flight System Dynamics, Technical University of Munich, 2020. Disponível em: https://www.fsd.lrg.tum.de/software/wp-content/uploads/UserGuideMain.pdf

[19] Y. Nie, O. Faqir, e E. Kerrigan, *ICLOCS2 - Imperial College London Optimal Control Software*, 1ª ed., 2018. Disponível em: http://www.ee.ic.ac.uk/ICLOCS/

[20] M. A. Patterson e A. V. Rao, "GPOPS-II: A MATLAB software for solving multiple-phase optimal control problems using hp-adaptive gaussian quadrature collocation methods and sparse nonlinear programming," *ACM Transactions on Mathematical Software*, vol. 41, no. 1, pp. 1–12, 2014. Disponível em: https://doi.org/10.1145/2558904

[21] P. E. Rutquist e M. M. Edvall, *PROPT - Matlab Optimal Control Software*, 1ª ed., 2010. Disponível em: https://tomopt.com/docs/TOMLAB_PROPT.pdf

[22] I. E. Paromtchik e C. Laugier, "Autonomous parallel parking of a nonholonomic vehicle," em *Proceedings of Conference on Intelligent Vehicles*, V. Piper, Ed., 1996, pp. 1–10. Disponível em: https://doi.org/10.1109/IVS.1996.566343

[23] M. Kelly, "An introduction to trajectory optimization: How to do your own direct collocation," *SIAM Review*, vol. 59, no. 4, pp. 849–904, 2017. Disponível em: https://doi.org/10.1137/16M1062569

[24] MathWorks, "Change Folders on the Search Path - MATLAB & Simulink," 2020. Disponível em: https://www.mathworks.com/help/matlab/matlab_env/add-remove-or-reorder-folders-on-the-search-path.html

[25] J. A. Parejo, A. Ruiz-Cortés, S. Lozano, e P. Fernandez, "Metaheuristic optimization frameworks: A survey and benchmarking," *Soft Computing*, vol. 1, no. 16, pp. 527–561, 2012. Disponível em: https://doi.org/10.1007/s00500-011-0754-8

[26] B. Li, K. Wang, e Z. Shao, "Time-optimal maneuver planning in automatic parallel parking using a simultaneous dynamic optimization approach," *IEEE Transactions on Intelligent Transportation Systems*, vol. 11, no. 17, pp. 3263–3274, 2016. Disponível em: https://doi.org/10.1109/TITS.2016.2546386

[27] O. Qx, I. Bongartz, A. Conn, N. Gould, e M. Saunders, "A numerical comparison between the LANCELOT and MINOS packages for large-scale constrained optimization," *Council for the Central Laboratory of The Research Councils*, vol. 1, no. 1, pp. 1–9, 1997. Disponível em: https://cds.cern.ch/record/338797/files/SCAN-9711063.pdf

[28] C. L. Darby, W. W. Hager, e A. V. Rao, "An hp-adaptive pseudospectral method for solving optimal control problems," *Optimal Control Applications and Methods*, vol. 32, no. 4, pp. 476–502, 2011. Disponível em: https://doi.org/10.1002/oca.957

[29] E. D. Dolan, J. J. Moré, e T. S. Munson, *Benchmarking Optimization Software With COPS 3.0*, 1ª ed., 2018. Disponível em: https://doi.org/10.2172/834714